9.2

IBM MQ Vývoj odkazů na aplikace

Poznámka

Než začnete používat tyto informace a produkt, který podporují, přečtěte si informace, které uvádí “Poznámky” na stránce 2179.

Toto vydání se vztahuje k verzi 9 vydání 2 produktu IBM® MQ a ke všem následujícím vydáním a modifikacím, dokud nebude v nových vydáních uvedeno jinak.

Obsah

Odkaz na vývoj aplikací

Prostřednictvím odkazů uvedených v této části můžete vyvíjet aplikace produktu IBM MQ .

- “Odkaz na aplikace MQI” na stránce 7
- IBM i “IBM i Application Programming Reference (ILE/RPG)” na stránce 986
- IBM i “Převod dat v systému IBM i” na stránce 1421
- “Uživatelské procedury, uživatelské procedury rozhraní API a odkazy na instalovatelné služby” na stránce 1440
- “Třídy a rozhraní produktu IBM MQ .NET” na stránce 1708
- “IBM MQ Třídy C++” na stránce 1770
- “Vlastnosti objektů IBM MQ classes for JMS” na stránce 1879
- “Odkaz systému zpráv REST API” na stránce 2148

Související úlohy

Vývoj aplikací

Související odkazy

Třídy IBM MQ pro knihovny Java

Třídy IBM MQ pro JMS

Odkaz na aplikace MQI

Prostřednictvím odkazů uvedených v této části můžete usnadnit vývoj aplikací rozhraní MQI (Message Queue Interface).

- “Příklady kódu” na stránce 8
- “Konstanty” na stránce 60
- “Datové typy použité v rozhraní MQI” na stránce 235
- “Volání funkcí” na stránce 620
- “Atributy objektů” na stránce 791
- “Návratové kódy” na stránce 866
- “Pravidla pro ověření platnosti voleb MQI” na stránce 866
- “Kódování počítače” na stránce 890
- “Volby sestav a příznaky zpráv” na stránce 893
- “Uživatelská procedura konverze dat” na stránce 897
- “Vlastnosti zadané jako prvky MQRFH2 .” na stránce 920
- “Převod kódové stránky” na stránce 929

Související pojmy

“Uživatelské procedury, uživatelské procedury rozhraní API a odkazy na instalovatelné služby” na stránce 1440

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

Související úlohy

Vývoj aplikací

Související odkazy

“Třídy a rozhraní produktu IBM MQ .NET” na stránce 1708

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

“IBM MQ Třídy C++” na stránce 1770
Třídy jazyka C++ produktu IBM MQ zapouzdřují rozhraní MQI (Message Queue Interface) produktu IBM MQ . K dispozici je jeden soubor záhlaví C + +, **imqi.hpp**, který pokrývá všechny tyto třídy.

Třídy IBM MQ pro knihovny produktu Java

IBM MQ Třídy pro JMS

Příklady kódu

Referenční informace v této sekci slouží k provedení úloh, které řeší vaše obchodní potřeby.

Příklady jazyků C

Tato kolekce témat je většinou převzata z ukázkových aplikací produktu IBM MQ for z/OS . Jsou použitelné na všechny platformy, kromě případů, kdy je to uvedeno.

Připojování ke správci front

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

Tento extrakt je převzat z ukázkové aplikace Procházet (program CSQ4BCA1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, 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 “Připojování ke správci front” na stránce 8. Tento extrakt je převzat z ukázkové aplikace Procházet (program CSQ4BCA1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tento extrakt je převzat z ukázkové aplikace Mail Manager (program CSQ4TCD1) dodávanou s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQOPEN k otevření fronty, která již byla definována.

Tento extrakt je převzat z ukázkové aplikace Procházet (program CSQ4BCA1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQCLOSE k uzavření fronty.

Tento extrakt je převzat z ukázkové aplikace Procházet (program CSQ4BCA1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQPUT k vložení zprávy do fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných s produktem IBM MQ. Názvy a umístění ukázkových aplikací naleznete v tématu Ukázkové procedurální programy (platformy kromě z/OS) z/OS a Ukázkové programy pro produkt 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 příkazu MQPUT1

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

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditu (program CSQ4CCB5) dodávanou s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQGET k odebrání zprávy z fronty.

Tento extrakt je převzat z ukázkové aplikace Procházet (program CSQ4BCA1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak používat volbu čekání na volání MQGET.

Tento kód přijímá oříznuté zprávy. Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditu (program CSQ4CCB5) dodávanou s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 IBM MQ for z/OS .

Tento příklad ukazuje, jak použít volání MQGET k nastavení signálu tak, abyste byli upozorněni, když přijde vhodná zpráva do fronty. Tento extrakt se nevezme z ukázkových aplikací dodaných 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;
}
```

Inquaktování o atributech objektu

Tento příklad ukazuje, jak použít volání MQINQ k dotazům na atributy fronty.

Tento extrakt je převzat z ukázkové aplikace Atributy fronty (program CSQ4CCC1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQSET ke změně atributů fronty.

Tento extrakt je převzat z ukázkové aplikace Atributy fronty (program CSQ4CCC1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 COBOL

Tato kolekce témat je převzata z ukázkových aplikací produktu IBM MQ for z/OS . Jsou použitelné na všechny platformy, kromě případů, kdy je to uvedeno.

Připojování ke správci front

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

Tento extrakt je odebrán z ukázkové aplikace Procházet (program CSQ4BVA1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, 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 “Připojování ke správci front” na stránce 23. Tento extrakt je odebrán z ukázkové aplikace Procházet (program CSQ4BVA1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQOPEN k otevření existující fronty.

Tento extrakt je odebrán z ukázkové aplikace Procházet (program CSQ4BVA1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 “Připojování ke správci front” na stránce 23. Tento extrakt je odebrán z ukázkové aplikace Procházet (program CSQ4BVA1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak používat volání MQPUT pomocí kontextu.

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 příkazu MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 .

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB5) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQGET k odebrání zprávy z fronty.

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB1) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQGET s volbou wait a přijetí oříznutých zpráv.

Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB5) dodávaného s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak používat volání MQGET se signalizací. Tento extrakt je převzat z ukázkové aplikace pro kontrolu kreditů (program CSQ4CVB2) dodávaného s produktem IBM MQ for z/OS.

Signalizace je k dispozici pouze s 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
*
     ⋮
```

Inquaktování o atributech objektu

Tento příklad ukazuje, jak použít volání MQINQ k dotazům na atributy fronty.

Tento extrakt je převzat z ukázkové aplikace Atributy fronty (program CSQ4CVC1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 ukazuje, jak použít volání MQSET ke změně atributů fronty.

Tento extrakt je převzat z ukázkové aplikace Atributy fronty (program CSQ4CVC1) dodávané s produktem IBM MQ for z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové procedurální programy (platformy kromě 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 produktu IBM MQ for z/OS .

Připojování ke správci front

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

Tento extrakt je převzat z ukázkového programu Procházet (CSQ4BAA1) dodaného s IBM MQ for z/OS.

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

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

```
*
         LA    R0,MQCC_OK       Expected compcode
         C     R0,COMPCODE      As expected?
         BER   R6               Yes .. return to caller
*
         MVC   INF4_TYP,=CL10'CONNECT   '
         BAL   R7,ERRCODE             Translate error
         LA    R0,8                   Set exit code
         ST    R0,EXITCODE            to 8
         B     ENDPROG                End the program
*
```

Odpojení od správce front

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

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQOPEN k otevření fronty, která již byla definována.

Ukazuje, jak uvést dvě volby. Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQCLOSE k uzavření fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQPUT k vložení zprávy do fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 příkazu MQPUT1

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

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQGET k odebrání zprávy z fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak používat volbu čekání na volání MQGET.

Tento kód přijímá oříznuté zprávy. Tento extrakt se nevezme z ukázkových aplikací dodaných 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 tak, abyste byli upozorněni, když přijde vhodná zpráva do fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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
```

Inkviování a nastavení atributů fronty

Tento příklad ukazuje, jak použít volání MQINQ k zjištění atributů fronty a použití volání MQSET pro změnu atributů fronty.

Tento extrakt je převzat z ukázkové aplikace Atributy fronty (program CSQ4CAC1) dodávané 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 jazyka PL/I je podporováno pouze produktem z/OS . Tato kolekce témat demonstruje techniky používající příklady PL/I.

Připojování ke správci front

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

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQDISC k odpojení programu od správce front v dávce z/OS .

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQOPEN k vytvoření dynamické fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQOPEN k otevření existující fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak používat volání MQPUT pomocí kontextu.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 příkazu MQPUT1

Tento příklad ukazuje, jak použít volání MQPUT1 .

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQGET k odebrání zprávy z fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQGET s volbou wait a přijetí oříznutých zpráv.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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

Extrakt kódu, který ukazuje, jak používat volání MQGET se signalizací.

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

Tento extrakt se nevezme z ukázkových aplikací dodaných 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;
```

Inquaktování o atributech objektu

Tento příklad ukazuje, jak použít volání MQINQ k dotazům na atributy fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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 ukazuje, jak použít volání MQSET ke změně atributů fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných 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

Referenční informace v této sekci slouží k provedení úloh, které řeší vaše obchodní potřeby.

Soubory IBM MQ COPY, header, include a module

Tyto informace jsou obecné informace o programovém rozhraní.

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

Soubory záhlaví C

Jsou poskytnuty soubory záhlaví, které vám pomohou s napsáním aplikačních programů jazyka C, které používají rozhraní MQI.

Soubory záhlaví C jsou shrnuty v následující tabulce:

Tabulka 1. Soubory hlaviček C-prototypy volání, datové typy, návratové kódy, konstanty a struktury

Název souboru	Popis	IBM i	Systémy AIX and Linux®	Windows	z/OS
Volat prototypy, datové typy, návratové kódy, konstanty a struktury					
CMQC	Definice 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
CMQPSCCOMMENT	Definice publikování/odběru	C	C	C	C
CMQXC	Definice kanálů a vyplutí	C	C	C	C
CMQZC	Definice instalovatelných služeb	C	C	C	
Klíč: C= Soubory k dispozici					

Soubory COBOL COPY

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

Tabulka 2. Soubory kopie jazyka COBOL-návratové kódy, konstanty a struktury

Název souboru	Popis	IBM i	AIX and Linux	Windows	z/OS
Návratové kódy a konstanty					
CMQx	Definice MQI	V	V	V	V
CMQCFx	Definice PCF	V	V	V	V
CMQPSx	Definice publikování/odběru	V	V	V	V
CMQXx	Definice kanálů a vyplutí	V	V	V	V
Struktury					
CMQAIRx	MQAIR-záznam ověřovacích informací		V L	V L	
CMQBOx	MQBO-Začátek voleb	V L	V L	V L	
CMQCDx	MQCD-Definice kanálu	V L	V L	V L	V L
CMQCFBFx	MQCFBF-parametr filtru bajtových řetězců PCF	V L	V L	V L	V L
CMQCFBSx	MQCFBS-parametr bajtového řetězce PCF	V L	V L	V L	V L

Tabulka 2. Soubory kopie jazyka COBOL-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQCFGRx	MQCFGR-parametr skupiny PCF	V L	V L	V L	V L
CMQCFHx	Záhlaví MQCFH-PCF	V L	V L	V L	V L
CMQCFIFx	MQCFIF-parametr filtru celých čísel 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	MQCFIN-Celočíselný parametr PCF	V L	V L	V L	V L
CMQCFSFx	MQCFSF-parametr filtru řetězce PCF	V L	V L	V L	V L
CMQCFSLx	MQCFSL-parametr seznamu řetězců PCF	V L	V L	V L	V L
CMQCFSTx	MQCFST-parametr řetězce PCF	V L	V L	V L	V L
CMQCFXLx	MQCFIL64 -64bitový seznam celočíselných hodnot PCF	V L	V L	V L	V L
CMQCFXNx	MQCFIN64 -64bitový parametr PCF 64bitů	V L	V L	V L	V L
CMQCHRVx	MQCHARV-Řetězec proměnné délky	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	MQDH-záhlaví distribuce	V L	V L	V L	V L
CMQDLHx	Záhlaví MQDLH-Dead-letter	V L	V L	V L	V L
CMQDXPx	MQDXP-Parametry uživatelské procedury pro převod dat	V L		V L	
CMQEPHx	MQEPH-záhlaví vloženého PCF	V L	V L	V L	V L
CMQGMOx	MQGMO-Získat volby zpráv	V L	V L	V L	V L
CMQIIHx	Záhlaví informací MQIIH- IMS	V L	V L	V L	V L
CMQMDx	MQMD-deskriptor zprávy	V L	V L	V L	V L
CMQMD1x	MQMD1 -Popisovač zprávy verze 1	V L	V L	V L	V L
CMQMD2x	MQMD2 -Verze deskriptoru zpráv 2	V L	V L	V L	V L
CMQMDEx	MQMDE-Rozšířený deskriptor zprávy	V L	V L	V L	V L
CMQODx	MQOD-Deskriptor objektu	V L	V L	V L	V L
CMQORx	MQOR-Záznam objektu	V L	V L	V L	V L
CMQPMOx	MQPMO-Vložit volby zprávy	V L	V L	V L	V L

Tabulka 2. Soubory kopie jazyka COBOL-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQRFHx	MQRFH-Pravidla a záhlaví formátování	V L	V L	V L	V L
CMQRFH2x	MQRFH2 -Pravidla a formátovací záhlaví 2	V L	V L	V L	V L
CMQRMHx	MQRMH-záhlaví zprávy odkazu	V L	V L	V L	V L
CMQRRx	MQRR-záznam odpovědi	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 -Znak zprávy spouštěče 2.	V L	V L	V L	V L
CMQWIHx	MQWIH-Záhlaví informací o práci	V L	V L	V L	V L
CMQXQHx	MQXQH-Hlavička přenosové fronty	V L	V L	V L	V L
Klíč: • Soubory s výchozími hodnotami uvedenými, x = V • Soubory bez počátečních hodnot poskytnutých, x = L					

z/OS *Soubory začlenění PL/I*

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

Tabulka 3. PL/I zahrnout 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 MQI				P
CMQCFP	Definice PCF				P
CMQEPP	Definice vstupního bodu				P
CMQPSP	Definice publikování/odběru				P
CMQXP	Definice kanálů a vyplutí				P
Klíč: P= Poskytnutý soubor					

IBM i *Soubory kopií RPG*

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

Tabulka 4. Soubory kopie RPG-návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
Návratové kódy a konstanty					

Tabulka 4. Soubory kopie RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQx	Definice MQI	G R			
CMQCFx	Definice PCF	G			
CMQPSx	Definice publikování/odběru	G			
CMQXx	Definice kanálů a vyplutí	G R			
Struktury					
CMQBOx	MQBO-Začátek voleb	G H			
CMQCDx	MQCD-Definice kanálu	G H R			
CMQCFBFx	MQCFBF-parametr filtru bajtových řetězců PCF	G H			
CMQCFBSx	MQCFBS-parametr bajtového řetězce PCF	G H			
CMQCFGRx	MQCFGR-parametr skupiny PCF	G H			
CMQCFHx	Záhlaví MQCFH-PCF	G H			
CMQCFIFx	MQCFIF-parametr filtru celých čísel PCF	G H			
CMQCFILx	MQCFIL-parametr seznamu celých čísel PCF	G H			
CMQCFINx	MQCFIN-Celočíselný parametr PCF	G H			
CMQCFSFx	MQCFSF-parametr filtru řetězce PCF	G H			
CMQCFSLx	MQCFSL-parametr seznamu řetězců PCF	G H			
CMQCFSTx	MQCFST-parametr řetězce PCF	G H			
CMQCFXLx	MQCFIL64 -64bitový seznam celočíselných hodnot PCF	G H			
CMQCFXNx	MQCFIN64 -64bitový parametr PCF 64bitů	G H			
CMQCHARVx	MQCHARV-Řetězec proměnné délky	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	MQDH-záhlaví distribuce	G H R			
CMQDLHx	Záhlaví MQDLH-Dead-letter	G H R			
CMQDXPx	MQDXP-Parametry uživatelské procedury pro převod dat	G H R			
CMQEPHx	MQEPH-záhlaví vloženého PCF	G H			

Tabulka 4. Soubory kopie RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQGMOx	MQGMO-Získat volby zpráv	G H R			
CMQIIHx	Záhlaví informací MQIIH- IMS	G H R			
CMQMDx	MQMD-deskriptor zprávy	G H R			
CMQMD1x	MQMD1 -Popisovač zprávy verze 1	G H R			
CMQMD2x	MQMD2 -Verze deskriptoru zpráv 2	G H			
CMQMDEx	MQMDE-Rozšířený deskriptor zprávy	G H R			
CMQODx	MQOD-Deskriptor objektu	G H R			
CMQORx	MQOR-Záznam objektu	G H R			
CMQPMOx	MQPMO-Vložit volby zprávy	G H R			
CMQPXPx	MQPXP-Parametry uživatelské procedury směrování publikování/ odběru	G H			
CMQRFHx	MQRFH-Pravidla a záhlaví formátování	G H			
CMQRFH2x	MQRFH2 -Pravidla a formátovací záhlaví 2	G H			
CMQRMHx	MQRMH-záhlaví zprávy odkazu	G H R			
CMQRRx	MQRR-záznam odpovědi	G H R			
CMQTMx	MQTM-Zpráva spouštěče	G H R			
CMQTMCx	MQTMC-Znak zprávy spouštěče	G H R			
CMQTMC2x	MQTMC2 -Znak zprávy spouštěče 2.	G H R			
CMQWIHx	MQWIH-Záhlaví informací o práci	G H			
CMQXQHx	MQXQH-Hlavička přenosové fronty	G H R			
Klíč: • Soubor pro statické sestavení, inicializovaný, poskytnutý x = G • Soubor pro statické sestavení, neinicializovaný, poskytnutý x = H • Soubor pro dynamické sestavení, inicializovaný, poskytnutý, x = R					

Windows *Soubory modulu Visual Basic*

Jsou poskytnuty soubory záhlaví (nebo formuláře), které vám pomohou s napsáním aplikačních programů jazyka Visual Basic, které používají rozhraní MQI. Tyto soubory záhlaví jsou dodávány pouze ve 32bitové verzi.

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					

Tabulka 5. Soubory modulu Visual Basic-deklarace volání, datové typy, návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**Systémy AIX and Linux**	**Windows**	**z/OS**
CMQB	Definice MQI			B	
CMQBB	definice MQAI			B	
CMQCFB	Definice PCF			B	
CMQXB	Definice kanálů a vyplutí			B	
Klíč: B= Poskytnutý soubor					

z/OS *z/OS Soubory COPY Assembler*

K dispozici jsou různé soubory COPY, které vám pomohou napsat aplikační programy produktu z/OS Assembler, které používají rozhraní MQI.

Tabulka 6. z/OS Soubory kopií asembler-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 MQI				A
CMQCFA	Definice PCF				A
CMQPSA	Definice publikování/odběru				A
CMQVERA	Řízení verze struktury				A
CMQXA	Definice kanálů a vyplutí				A
Struktury					
CMQCDA	MQCD-Definice kanálu				
CMQCFBFA	MQCFBF-parametr filtru bajtových řetězců PCF				
CMQCFBSA.	MQCFBS-parametr bajtového řetězce PCF				A
CMQCFGRA	MQCFGR-parametr skupiny PCF				A
CMQCFHA	Záhlaví MQCFH-PCF				A
CMQCFIFIA	MQCFIF-parametr filtru celých čísel PCF				A
CMQCFILA	MQCFIL-parametr seznamu celých čísel PCF				A
CMQCFINA	MQCFIN-Celočíselný parametr PCF				A
CMQCFSFA	MQCFSF-parametr filtru řetězce PCF				A
CMQCFSLA	MQCFSL-parametr seznamu řetězců PCF				A
CMQCFSTA	MQCFST-parametr řetězce PCF				A

Tabulka 6. z/OS Soubory kopií asembler-datové typy, návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQCFXLA	MQCFIL64 -64bitový seznam celočíselných hodnot PCF				A
CMQCFXNA	MQCFIN64 -64bitový parametr PCF 64bitů				A
CMQCHARVA	MQCHARV-Řetězec proměnné délky				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	MQDH-záhlaví distribuce				A
CMQDLHA	Záhlaví MQDLH-Dead-letter				A
CMQDXPA	MQDXP-Parametry uživatelské procedury pro převod dat				A
CMQEPHA	MQEPH-záhlaví vloženého PCF				A
CMQGMOA	MQGMO-Získat volby zpráv				A
CMQIIHA	Záhlaví informací MQIIH- IMS				A
CMQMDA	MQMD-deskriptor zprávy				A
CMQMD1A	MQMD1 -Popisovač zprávy verze 1				A
CMQMD2A	MQMD2 -Verze deskriptoru zpráv 2				A
CMQMDEA	MQMDE-Rozšířený deskriptor zprávy				A
CMQODA	MQOD-Deskriptor objektu				A
CMQORA	MQOR-Záznam objektu				A
CMQPMOA	MQPMO-Vložit volby zprávy				A
CMQRFHA	MQRFH-Pravidla a záhlaví formátování				A
CMQRFH2A	MQRFH2 -Pravidla a formátovací záhlaví 2				A
CMQRMHA	MQRMH-záhlaví zprávy odkazu				A
CMQTMA	MQTM-Zpráva spouštěče				A
CMQTMC2A	MQTMC2 -Znak zprávy spouštěče 2.				A
CMQWCRA	MQWCR-Záznam klastru pracovní zátěže klastru				A
CMQWDRA	MQWDR-Cílový záznam pracovní zátěže klastru				A

Tabulka 6. z/OS Soubory kopií asembler-datové typy, návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**AIX and Linux**	**Windows**	**z/OS**
CMQWDR1A	MQWDR1 -Záznam místa určení pracovní zátěže klastru verze 1				A
CMQWDR2A	MQWDR2 -Cílová verze záznamu pracovní zátěže klastru 2				A
CMQWIHA	MQWIH-Záhlaví informací o práci				A
CMQWQRAQ.	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 uživatelské procedury pracovní zátěže klastru				A
CMQWXP1A	MQWXP1 -Parametry uživatelské procedury pracovní zátěže klastru verze 1				A
CMQWXP2A	MQWXP2 -Parametry uživatelské procedury pracovní zátěže klastru verze 2				A
CMQWXP3A	MQWXP3 -Parametry uživatelské procedury pracovní zátěže klastru verze 3				A
CMQXPA	MQXP- CICS Parametry ukončení překřížení rozhraní API				A
CMQXQHA	MQXQH-Hlavička přenosové fronty				A
CMQXWDA	MQXWD-Ukončení deskriptoru čekání				A
Klíč: A= Poskytnutý soubor					

MQ_ * (Délky řetězce)

Tabulka 7. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_ABEND_CODE_LENGTH	4	X'00000004'
MQ_ACCOUNTING_TOKEN_LENGTH	32	X'00000020'
DÉLKA_FUNKCE_FUNKCE_MQ_APPL_	10	X'0000000A'
DÉLKA MQ_APPL_IDENTITY_IDENTITY_DATA_	32	X'00000020'
DÉLKA_APL_KQ_MQ	28	X'0000001C'
MQ_APPL_ORIGIN_DATA_LENGTH	4	X'00000004'
MQ_APL_TAG_LENGTH	28	X'0000001C'
DÉLKA PARAMETRU MQ_ARM_SUFFIX_LENGTH	2	X'00000002'

Tabulka 7. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
DÉLKA_MQ_ATTENTION_ID_	4	X'00000004'
MQ_AUTH_INFO_CONN_NAME_LENGTH	264	X'00000108'
MQ_AUTH_INFO_ODESC_DÉLKA	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_BRIGE_NAME_LENGTH	24	X'00000018'
MQ_CELKOVÝ_KÓDOVÁ_DÉLKA	4	X'00000004'
MQ_CF_STRUČNÁ_DÉLKY	64	X'00000040'
DÉLKA_STRUČNÝ_NÁZEV_ROZHRANÍ_MQ_CF	12	X'0000000C'
MQ_CHANNEL_DATE_LENGTH	12	X'0000000C'
DÉLKA_KANÁLŮ_MQ_CHANNEL_LENGTH	64	X'00000040'
DÉLKA_KANÁLU_MQ_KANÁLU	20	X'00000014'
ČASOVÁ_DÉLKA MQ_CHANNEL_TIME_LENGTH	8	X'00000008'
MQ_CHINIT_SERVICE_PARM_DÉLKA	32	X'00000020'
DÉLKA_NÁZVU_NÁZVU_MQ_SOUBORU	8	X'00000008'
ID_KLIENTA_KLIENTA_SPRÁVY	23	X'00000017'
DÉLKA_KLASTRU MQ_CLUSTER_LENGTH	48	X'00000030'
DÉLKA_NÁZVU_MQ_SERVERU	264	X'00000108'
MQ_CONN_TAG_LENGTH	128	X'00000080'
DÉLKA_PŘIPOJENÍ_MQ_ID_PŘIPOJENÍ	24	X'00000018'
MQ_CORRELA_ID_LENGTH	24	X'00000018'
MQ_CREATION_DATE_LENGTH	12	X'0000000C'
MQ_DOBA_VYTVOŘENÍ_ČASU	8	X'00000008'
MQ_DATUM_DÉLKA	12	X'0000000C'
MQ_DISTINGUISHED_NAME_LENGTH	1024	X'00000400'
MQ_DNS_NÁZEV_SKUPINY_NA_DÉLKU	18	X'00000012'
MQ_EXIT_DATA_LENGTH	32	X'00000020'
NÁZEV MQ_EXIT_INFO_NAME_LENGTH	48	X'00000030'
DÉLKA_NÁZVU_MQ_EXIT_LENGTH	(value differs by platform or version)	
MQ_EXIT_PD_AREA_LENGTH	48	X'00000030'
MQ_EXIT_USER_AREA_LENGTH	16	X'00000010'
MQ_FACILITA_DÉLKA	8	X'00000008'
MQ_FACILITY_LIKE_LENGTH	4	X'00000004'
DÉLKA_FORMÁTU MQ_FORMÁTU	8	X'00000008'
DÉLKA_FUNKCE_MQ_FUNKCE	4	X'00000004'

Tabulka 7. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQ_GROUP_ID_DÉLKA	24	X'00000018'
MQ_LDAP_PASSWORD_LENGTH	32	X'00000020'
DÉLKA_NÁZVU_MQ_LISTENER	48	X'00000030'
DÉLKA_LISTENER_MQ_LISTENERA	64	X'00000040'
DÉLKA_LOKÁLNÍ_ADRESA_TIP	48	X'00000030'
MQ_LTERM_OVERRIDE_LENGTH	8	X'00000008'
DÉLKA_NÁZVU_MQL	8	X'00000008'
MQ_LUWID_LENGTH	16	X'00000010'
MQ_MAX_EXIT_NAME_LENGTH	128	X'00000080'
MQ_MAX_MCA_USER_ID_DÉLKA	64	X'00000040'
MQ_MAX_DÉLKA_VLASTNOSTI_VLASTNOSTI	4095	X'00000FFF'
MQ_MAX_ID_UŽIV_ID_UŽIVATELE	64	X'00000040'
DÉLKA_ÚLOHY_MQ_MCA_JM_ÚLOHY	28	X'0000001C'
DÉLKA MQ_MCA_NAME_LENGTH	20	X'00000014'
MQ_MCA_USER_DATA_LENGTH	32	X'00000020'
DÉLKA MQ_MCA_USER_ID_LENGTH	(value differs by platform or version)	(value differs by platform or version)
DÉLKA_MAPOVÁNÍ_MFS_MFS_MAP	8	X'00000008'
DÉLKA_MODELU_MQ_REŽIMU	8	X'00000008'
DÉLKA_HLAVNÍHO_ZÁHLAVÍ MQ_MSG_	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'
DÉLKA_NÁZVU_INSTANCE_MQ_NHA_	48	X'00000030'
DÉLKA_INSTANCE_MQ_OBJEKTU	24	X'00000018'
MQ_OBJECT_NAME_LENGTH	48	X'00000030'
DÉLKA_TIKETU_APLIK MQ_PASS_APPL_	8	X'00000008'
DÉLKA_MQ_HESLA	12	X'0000000C'
MQ_PROCESS_APPL_ID_LENGTH	256	X'00000100'
DÉLKA_PROCESS_PROCESU MQ_PROCESS_	64	X'00000040'
MQ_PROCESS_ENV_DATA_LENGTH	128	X'00000080'
DÉLKA_PROCESU_MQ_PROCESU	48	X'00000030'
MQ_PROCESS_USER_DATA_LENGTH	128	X'00000080'
DÉLKA_NÁZVU_MQ_PROGRAMU	20	X'00000014'
DÉLKA_PLNÝ_NÁZEV_APL_SYSTÉMU	28	X'0000001C'
MQ_PUT_DATE_LENGTH	8	X'00000008'
MQ_PUT_TIME_LENGTH	8	X'00000008'
MQ_Q_ODESC_DÉLKA	64	X'00000040'
MQ_MGR_ODESC_DÉLKA	64	X'00000040'

Tabulka 7. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
DÉLKA MQ_Q_MGR_IDENTIFIER_LENGTH	48	X'00000030'
DÉLKA_MGR_NÁZVU_MQ_QM	48	X'00000030'
DÉLKA MQ_Q_NAME_LENGTH	48	X'00000030'
MQ_QSG_NAME_LENGTH	4	X'00000004'
MQ_REMOTE_SYS_ID_LENGTH	4	X'00000004'
MQ_ID_SADY_ZABEZPEČENÍ	40	X'00000028'
MQ_SELECTOR_LENGTH	10240	X'00002800'
DÉLKA MQ_SERVICE_ARGS_LENGTH	255	X'000000FF'
DÉLKA_PŘÍKAZU MQ_SERVICE_	255	X'000000FF'
DÉLKA_SLUŽBY MQ_SERVICE_	64	X'00000040'
DÉLKA_NÁZVU_SLUŽBY_MQ	32	X'00000020'
MQ_SERVICE_PATH_LENGTH	255	X'000000FF'
MQ_SERVICE_STEP_LENGTH	8	X'00000008'
DÉLKA_NÁZVU_V_DÉLCE MQ_SHORT_NA_	20	X'00000014'
MQ_SHORT_DNAME_LENGTH	256	X'00000100'
DÉLKA_ŠIFRY MQ_SSL_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'
DÉLKA_POČÁTEČNÍ_KÓD_MQ_	4	X'00000004'
DÉLKA_AGENTA_MQ_STORAGE_	64	X'00000040'
MQ_STORAGE_TŘÍDA_TŘÍDY	8	X'00000008'
MQ_SUB_IDENTITY_IDENTIFIKÁTORU	128	X'00000080'
DÉLKA_MEZIHO_BODU_MQ_BOD	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'
DÉLKA_NÁZVU_MQ_SERVERU	8	X'00000008'
MQ_TIME_LENGTH	8	X'00000008'
MQ_TOPIC_DESC_LENGTH	64	X'00000040'
DÉLKA_NÁZVU_TÉMAT_MC	48	X'00000030'
MQ_TOPIC_STR_LENGTH	10240	X'00002800'
MQ_TOTAL_EXIT_DATA_LENGTH	999	X'000003E7'
CELKOVÁ DÉLKA MQ_TOTAL_EXIT_NAME_LENGTH	999	X'000003E7'

Tabulka 7. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
DÉLKA MQ_TP_NAME_LENGTH	64	X'00000040'
DÉLKA_NÁZVU_FRONTY_MQ_TPIPE	8	X'00000008'
MQ_TRAN_INSTANCE_ID_DÉLKA	16	X'00000010'
MQ_TRANSACTION_ID_DÉLKA	4	X'00000004'
DÉLKA MQ_TRIGGER_DATA_LENGTH	64	X'00000040'
DÉLKA MQ_TRIGGER_PROGRAM_NAME_LENGTH	8	X'00000008'
DÉLKA_ID_SPOUŠTĚČE MQ_TRIGGER	4	X'00000004'
DÉLKA_TRANSAKCE_MQ_TRIGGERUCOMMENT	4	X'00000004'
DÉLKA MQ_USER_ID_LENGTH	12	X'0000000C'
DÉLKA_VERZE_MQ_VERZE	8	X'00000008'
MQ_NÁZEV_SKUPIN_NEBO_NÁZVU_SKUPINY_SKUPINY	8	X'00000008'
MQ_NÁZEV_ČLENA_ČLENA_ČLENA_ČLENA_ČLENU	16	X'00000010'

*MQ_ * (řetězce příkazového řetězce Délky)*

Tabulka 8. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_ARCHIVE_PFX_LENGTH	36	X'00000024'
MQ_ARCHIV_JEDNOTKA_JEDNOTKY_JEDNOTKY	8	X'00000008'
MQ_ASID_LENGTH	4	X'00000004'
DÉLKA_NÁZVU_PROFILU_MQ_AUTH_PROFILE	48	X'00000030'
MQ_CF_LEID_LENGTH	12	X'0000000C'
MQ_COMMAND_MQSC_LENGTH	32768	X'00008000'
DÉLKA OBJEKTU MQ_DATA_SET_NAME_LENGTH	44	X'0000002C'
MQ_DB2_NAME_LENGTH	4	X'00000004'
MQ_DSG_NAME_LENGTH	8	X'00000008'
DÉLKA_ENTITY_MQ_ENTITY_	1024	X'00000400'
DÉLKA_V_INFOM_MQ_ENV_O_	96	X'00000060'
DÉLKA MQ_IP_ADDRESS_LENGTH	48	X'00000030'
MQ_LOG_CORRELA_ID_LENGTH	8	X'00000008'
MQ_LOG_EXTENT_NAME_LENGTH	24	X'00000018'
MQ_LOG_PATH_LENGTH	1024	X'00000400'
DÉLKA MQ_LRSNA_LENGTH	12	X'0000000C'
DÉLKA_PŮVODNÍHO_NÁZVU	8	X'00000008'
MQ_PSB_NÁZVU_DÉLKY	8	X'00000008'
MQ_PST_ID_DÉLKA	8	X'00000008'
DÉLKA MQ_Q_MGR_CPF_LENGTH	4	X'00000004'
DÉLKA_ID_ODEZVY MQ_ID_	24	X'00000018'
DÉLKA MQ_RBA_LENGTH	16	X'00000010'
MQ_SECURITY_PROFILE_LENGTH	40	X'00000028'
MQ_SERVICE_COMPONENT_LENGTH	48	X'00000030'

Tabulka 8. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
DÉLKA_DÍLČÍ_FRONTY	10240	X'00002800'
MQ_SYSP_SERVICE_LENGTH	32	X'00000020'
DÉLKA_NÁZVU_SYSTÉMU_MQ_SYSTÉMU	8	X'00000008'
DÉLKA_ÚLOHY_MQ_TASK	8	X'00000008'
MQ_TPIPE_PFX_LENGTH	4	X'00000004'
MQ_UOW_ID_DÉLKA	256	X'00000100'
MQ_USER_DATA_LENGTH	10240	X'00002800'
MQ_VOLSER_DÉLKA	6	X'00000006'

MQACH_ * (struktura záhlaví oblasti řetězu uživatelských procedur rozhraní API)

Tabulka 9. Konstrukce konstant	
Název	**Struktura**
MQACH_STRUCTION_ID	"ACH¬"
POLE 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'
AKTUÁLNÍ_VERZE MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	(value differs by platform or version)
AKTUÁLNÍ_DÉLKA MQACH_LENGTH	(value differs by platform or version)	(value differs by platform or version)

MQACT_ * (evidenční token)

Tabulka 11. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQACT_NONE	X'00...00' (32 nulových hodnot)
MQACCT_NON_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'
PROTOKOL MQACCT_ADVANCE_LOG	2	X'00000002'
STATISTIKA KOLEKCÍ MQACT_COLLECT_	3	X'00000003'
MQACT_PUBSUB	4	X'00000004'

MQACTP_ * (Akce)

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

MQACTT_ * (typy účetních tokenů)

Tabulka 14. Hodnoty konstant	
Název	**Hexadecimální hodnota**
MQACTT_UNKNOWN	X'00'
MQACCT_CICS_LUOW_ID	X'01'
MQACTT_OS2_DEFAULT	X'04'
MQACTT_DOS_DEFAULT	X'05'
MQACTT_UNIX_NUMERIC_ID	X'06'
MQACTT_OS400_ACCOUNT_TOKEN	X'08'
MQACTC_WINDOWS_DEFAULT	X'09'
MQACTT_NT_SECURITY_ID	X'0B'
UŽIVATEL MQACTT_USER	X'19'

MQADOPT_ * (Převzetí nového agenta MCA-kontroly a převzetí nových typů MCA)

Převzetí nových kontrol 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'

Převzetí nových typů MCA

Tabulka 16. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQADOPT_TYPE_NO	0	X'00000000'
MQADOPT_TYPY_VŠE	1	X'00000001'
MQADOPT_TYPY_SVR	2	X'00000002'
MQADOPT_TYPY_SDR	4	X'00000004'
MQADOPT_TYPY_RCVR	8	X'00000008'
MQADOPT_TYPY_CLUSIVR	16	X'00000010'

MQAIR_ * (Struktura záznamu ověřovacích informací)

Tabulka 17. Konstrukce konstant

Název	Struktura
ID_STRUKTURY MQIR_CONSTRUCT	"AIR¬"
POLE MQIR_STRUC_ID_ARRAY	'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'
AKTUÁLNÍ_VERZE MQIR_CURRENT_VERSION	2	X'00000002'

MQAIT_ * (typ ověřovacích informací)

Tabulka 19. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIT_VŠE	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQACY_OCSP	2	X'00000002'
MQAIT_IDPW_OS	3	X'00000003'
MQITOM_IDPW_LDAP	4	X'00000004'

MQAS_ * (Asynchronní stavové hodnoty ve formátu příkazu)

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_SUSPENDED	4	X'00000004'
MQAS_SUSPENDED_TEMPORARY	5	X'00000005'
MQAS_ACTIVE	6	X'00000006'
MQAS_INACTIVE	7	X'00000007'

MQAT_ * (Put Application Types)

Tabulka 21. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQAT_UNKNOWN	-1	X'FFFFFFFF'
MQAT_NO_CONTEXT	0	X'00000000'
MQAT_CICS	1	X'00000001'
MQAT_MVS	2	X'00000002'
MQAT_OS390	2	X'00000002'

Tabulka 21. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQAT_ZOS	2	X'00000002'
MQAT_IMS	3	X'00000003'
MQAT_OS2	4	X'00000004'
MQAT_DOS	5	X'00000005'
MQAT_AIX	6	X'00000006'
MQAT_UNIX	6	X'00000006'
MQAT_QMGR	7	X'00000007'
MQAT_OS400	8	X'00000008'
MQAT_WINDOWS	9	X'00000009'
MQAT_CICS_VSE	10	X'0000000A'
POČ MQAT_WINDOWS_NT	11	X'0000000B'
MQAT_VMS	12	X'0000000C'
MQAT_GUARDIAN	13	X'0000000D'
MQAT_NSK	13	X'0000000D'
MQAT_VOS	14	X'0000000E'
MQAT_OPEN_TP1	15	X'0000000F'
MQAT_VM	18	X'00000012'
MOST MQAT_IMS_BRIDGE	19	X'00000013'
MQAT_XCF	20	X'00000014'
MOST MQAT_CICS_BRIDGE	21	X'00000015'
MQAT_NOTES_AGENT	22	X'00000016'
MQAT_TPF	23	X'00000017'
UŽIVATEL 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'
INICIALIZÁTOR 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 HODNOTA MQAT_DEFAULT	(value differs by platform or version)	(value differs by platform or version)
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

MQAUTH_ * (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	3	X'00000003'
MQAUTH_CLEAR	4	X'00000004'
MQAUTH_CONNECT	5	X'00000005'
VYTVOŘIT_MQAUTH_CREATE	6	X'00000006'
ODSTRANIT MQAUTH_DELETE	7	X'00000007'
MQAUTH_DISPLAY	8	X'00000008'
VSTUP MQAUTH_INPUT	9	X'00000009'
MQAUTH_INQUIRE	10	X'0000000A'
VÝSTUP MQAUTH_OUTPUT	11	X'0000000B'
MQAUTH_PASS_ALL_CONTEXT, KONTEXT	12	X'0000000C'
KONTEXT MQAUTH_PASS_IDENTITY_CONTEXT	13	X'0000000D'
MQAUTH_SET	14	X'0000000E'
MQAUTH_SET_ALL_CONTEXT,	15	X'0000000F'
KONTEXT MQAUTH_SET_IDENTITY_CONTEXT	16	X'00000010'
OVLADAČ MQAUTH_CONTROL	17	X'00000011'
FUNKCE MQAUTH_CONTROL_EXTENDED	18	X'00000012'
MQAUTH_PUBLISH	19	X'00000013'
MQAUTH_SUBSCRIBE	20	X'00000014'
MQAUTH_RESUME	21	X'00000015'
SYSTÉM MQAUTH_SYSTEM	22	X'00000016'

MQAUTHOPT_ * (Volby oprávnění formátu příkazu)

Tabulka 23. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQAUTHOPT_SOUČTOVÉ	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. Konstrukce konstant

Název	Struktura
MQAXC_STRUC_ID	"AXC¬"

Tabulka 24. Konstrukce konstant (pokračování)	
Název	**Struktura**
MQAXC_STRUC_ID_POLE	'A','X','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

MQAXP_ * (struktura výstupního parametru rozhraní API)

Tabulka 26. Konstrukce konstant	
Název	**Struktura**
MQAXP_STRUCTION_ID	"AXP¬"
POLE 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'
AKTUÁLNÍ_VERZE MQAXP_	2	X'00000002'

MQBA_ * (Selektory bajtového atributu)

Tabulka 28. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQBA_FIRST	6001	X'00001771'
MQBA_LAST	8000	X'00001F40'

MQBACF_ * (typy bajtových parametrů příkazového formátu)

Tabulka 29. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBAKF_PRVNÍ	7001	X'00001B59'
MQBAKF_EVENT_ACCOUNTING_TOKEN	7001	X'00001B59'
MQBAKF_EVENT_SECURITY_ID	7002	X'00001B5A'
MQBACF_RESPONSE_SET	7003	X'00001B5B'
MQBAKF_RESPONSE_ID	7004	X'00001B5C'
MQBAC_EXTERNAL_UOW_ID	7005	X'00001B5D'
MQBAKF_CONNECTION_ID	7006	X'00001B5E'
MQBAKF_GENERICKÝ_CONNECTION_ID	7007	X'00001B5F'
MQBAF_ORIGIN_UOW_ID	7008	X'00001B60'
MQBAKF_Q_MGR_UOW_ID	7009	X'00001B61'

Tabulka 29. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBAKF_ACCOUNTING_TOKEN	7010	X'00001B62'
MQBAKF_CORRELACE_ID	7011	X'00001B63'
MQBAKF_GROUP_ID	7012	X'00001B64'
MQBAKF_MSG_ID	7013	X'00001B65'
MQBAKF_CF_LEID	7014	X'00001B66'
MQBAC_DESTINATION_CORREL_ID	7015	X'00001B67'
DÍLČÍ _ID MQBACF_SUB_ID	7016	X'00001B68'
MQBAF_LAST_POUŽITO	7016	X'00001B68'

MQBL_ * (Délka vyrovnávací paměti pro řetězec mqAddString a mqSetString)

Tabulka 30. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBL_NULL_TERMINATED	-1	X'FFFFFFFF'

MQBMHO_ * (vyrovnávací paměť pro volby a strukturu vyrovnávací paměti)

Struktura voleb popisovače zpráv do vyrovnávací paměti

Tabulka 31. Konstrukce konstant	
Název	**Struktura**
MQBMHO_STRUCTION_ID	"BMHO"
MQBMHO_STRUC_ID_POLE	'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'
AKTUÁLNÍ_VERZE MQBMHO_CURRENT_VERSION	1	X'00000001'

Volby popisovače zprávy do vyrovnávací paměti

Tabulka 33. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBMHO_NONE	0	X'00000000'
VLASTNOSTI 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'
SKUPINA MQBND_BIND_ON_GROUP	2	X'00000002'

MQBO_ * (počáteční volby a struktura)

Začátek struktury voleb

Tabulka 35. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY OBJEKTU MQBO_STRUCT	"BO¬¬"
MQBO_STRUC_ID_POLE	'B','O','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

Volby začátku

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

MQBT_ * (Typy mostů příkazového formátu)

Tabulka 38. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
FUNKCE 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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_CF_STRUC_NAME	2039	X'000007F7'
MQCA_CHANNEL_AUTO_DEF_EXIT	2026	X'000007EA'
MQCA_CHILD	2101	X'00000835'
MQCA_CHINIT_SERVICE_PARM	2076	X'0000081C'
MQCA_CICS_FILE_NAME	2060	X'0000080C'
MQCA_CLUS_CHL_NAME	2124	X'0000084C'
MQCA_CLUSTER_DATE	2037	X'000007F5'
MQCA_CLUSTER_NAME	2029	X'000007ED'
MQCA_CLUSTER_NAMELIST	2030	X'000007EE'
MQCA_CLUSTER_Q_MGR_NAME	2031	X'000007EF'
MQCA_CLUSTER_TIME	2038	X'000007F6'
MQCA_CLUSTER_WORKLOAD_DATA	2034	X'000007F2'
MQCA_CLUSTER_WORKLOAD_EXIT	2033	X'000007F1'
MQCA_COMMAND_INPUT_Q_NAME	2003	X'000007D3'
MQCA_COMMAND_REPLY_Q_NAME	2067	X'00000813'
MQCA_CREATION_DATE	2004	X'000007D4'
MQCA_CREATION_TIME	2005	X'000007D5'
MQCA_DEAD_LETTER_Q_NAME	2006	X'000007D6'
MQCA_DEF_XMIT_Q_NAME	2025	X'000007E9'
MQCA_DNS_GROUP	2071	X'00000817'
MQCA_ENV_DATA	2007	X'000007D7'
MQCA_FIRST	2001	X'000007D1'
MQCA_IGQ_USER_ID	2041	X'000007F9'
MQCA_INITIATION_Q_NAME	2008	X'000007D8'
MQCA_LAST	4000	X'00000FA0'
MQCA_LAST_USED	2109	X'0000083D'
MQCA_LDAP_PASSWORD	2048	X'00000800'
MQCA_LDAP_USER_NAME	2047	X'000007FF'
MQCA_LU_GROUP_NAME	2072	X'00000818'
MQCA_LU_NAME	2073	X'00000819'
MQCA_LU62_ARM_SUFFIX	2074	X'0000081A'
MQCA_MODEL_DURABLE_Q	2096	X'00000830'
MQCA_MODEL_NON_DURABLE_Q	2097	X'00000831'
MQCA_MONITOR_Q_NAME	2066	X'00000812'
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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_PROCESS_DESC	2011	X'000007DB'
MQCA_PROCESS_NAME	2012	X'000007DC'
MQCA_Q_DESC	2013	X'000007DD'
MQCA_Q_MGR_DESC	2014	X'000007DE'
MQCA_Q_MGR_IDENTIFIER	2032	X'000007F0'
MQCA_Q_MGR_NAME	2015	X'000007DF'
MQCA_Q_NAME	2016	X'000007E0'
MQCA_QSG_NAME	2040	X'000007F8'
MQCA_REMOTE_Q_MGR_NAME	2017	X'000007E1'
MQCA_REMOTE_Q_NAME	2018	X'000007E2'
MQCA_REPOSITORY_NAME	2035	X'000007F3'
MQCA_REPOSITORY_NAMELIST	2036	X'000007F4'
MQCA_RESUME_DATE	2098	X'00000832'
MQCA_RESUME_TIME	2099	X'00000833'
MQCA_SERVICE_DESC	2078	X'0000081E'
MQCA_SERVICE_NAME	2077	X'0000081D'
MQCA_SERVICE_START_ARGS	2080	X'00000820'
MQCA_SERVICE_START_COMMAND	2079	X'0000081F'
MQCA_SERVICE_STOP_ARGS	2082	X'00000822'
MQCA_SERVICE_STOP_COMMAND	2081	X'00000821'
MQCA_STDERR_DESTINATION	2084	X'00000824'
MQCA_STDOUT_DESTINATION	2083	X'00000823'
MQCA_SSL_CRL_NAMELIST	2050	X'00000802'
MQCA_SSL_CRYPTO_HARDWARE	2051	X'00000803'
MQCA_SSL_KEY_LIBRARY	2069	X'00000815'
MQCA_SSL_KEY_MEMBER	2070	X'00000816'
MQCA_SSL_KEY_REPOSITORY	2049	X'00000801'
MQCA_STORAGE_CLASS	2022	X'000007E6'
MQCA_STORAGE_CLASS_DESC	2042	X'000007FA'
MQCA_SYSTEM_LOG_Q_NAME	2065	X'00000811'
MQCA_TCP_NAME	2075	X'0000081B'
MQCA_TOPIC_DESC	2093	X'0000082D'
MQCA_TOPIC_NAME	2092	X'0000082C'
MQCA_TOPIC_STRING_FILTER	2108	X'0000083C'
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'

Tabulka 39. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (Typy znakových znakových parametrů příkazu)

Tabulka 40. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQCACF_	3001	X'00000BB9'
MQCAF_FROM_Q_NAME	3001	X'00000BB9'
NÁZEV OBJEKTU MQCACF_TO_Q_NAME	3002	X'00000BBA'
MQCACF_FROM_NÁZEV_PROCESU	3003	X'00000BBB'
NÁZEV PODPROCESU MQCACF_TO_PROCESS_NAME	3004	X'00000BBC'
MQCACFF_FROM_NAMELIST_NAME	3005	X'00000BBD'
NÁZEV MQCACFF_TO_NAMELIST_NAME	3006	X'00000BBE'
NÁZEV MQCACFF_FROM_CHANNELS	3007	X'00000BBF'
MQCAF_TO_CHANNELS	3008	X'00000BC0'
MQCACF_FROM_AUTH_INFO_NAME	3009	X'00000BC1'
MQCAF_TO_AUTH_INFO_NAME	3010	X'00000BC2'
NÁZVY QCACFF_Q_NAMES	3011	X'00000BC3'
NÁZVY PROCESS_MQCACF_PROCESS_	3012	X'00000BC4'
NÁZVY MQCACF_NAMELIST_NAMES	3013	X'00000BC5'
ESCAP_ESCAPE_TEXT_SOUBORU	3014	X'00000BC6'
NÁZVY MQCACFF_LOCAL_Q_NAMES	3015	X'00000BC7'
MQCACF_MODEL_Q_NAMES	3016	X'00000BC8'
MAQCAF_ALIAS_Q_NAMES	3017	X'00000BC9'
NÁZEV SOUBORU MQCACF_REMOTE_Q_NAMES	3018	X'00000BCA'
NÁZVY MQCACFF_SENDER_CHANNEL_NAMES	3019	X'00000BCB'
NÁZVY KANÁLŮ MQCACF_SERVER_CHANNEL_NAMES	3020	X'00000BCC'
NÁZVY MQCACFF_REKTESTER_CHANNEL_NAMES	3021	X'00000BCD'
NÁZVY_ZÁSOBNÍKŮ_MQCACF_RECEIVER_CHANNEL	3022	X'00000BCE'
MQCACF_NÁZEV_OBJEKTU_Q_MGR_NAME	3023	X'00000BCF'
NÁZEV_APLIK. MQCACF_	3024	X'00000BD0'
IDENTIFIKÁTOR UŽIVATELE MQCACFF_	3025	X'00000BD1'
MQCACF_AUX_ERROR_DATA_STR_1	3026	X'00000BD2'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_AUX_ERROR_DATA_STR_2	3027	X'00000BD3'
MQCACF_AUX_ERROR_DATA_STR_3	3028	X'00000BD4'
NÁZEV MQCACF_BRIDGE_NAME	3029	X'00000BD5'
MQCACF_STREAM_NAME	3030	X'00000BD6'
TÉMA MQCACF_	3031	X'00000BD7'
MQCACF_PARENT_QM_MGR_NAME	3032	X'00000BD8'
MQCAF_KOREKTORID.	3033	X'00000BD9'
ČASOVÉ_RAZÍTKO_PUBLIKAČNÍHO_SYSTÉMU	3034	X'00000BDA'
MQCAF_STRING_DATA	3035	X'00000BDB'
MQCACF_SUPPORTED_STREAM_NAME	3036	X'00000BDC'
MQCACF_REG_TOPIC-TÉMA	3037	X'00000BDD'
MQCAF_REG_ČAS	3038	X'00000BDE'
FUNKCE MQCACFF_REG_USER_ID	3039	X'00000BDF'
MQCACF_CHILD_Q_MGR_NAME	3040	X'00000BE0'
NÁZEV PROUDU MQCACF_REG_STREAM_NAME	3041	X'00000BE1'
MQCAF_REG_Q_MGR_NAME	3042	X'00000BE2'
NÁZEV QCACF_REG_QNAME	3043	X'00000BE3'
MQCAF_REG_CORRELATION_ID	3044	X'00000BE4'
ID_UŽIVATELE_MQCAF_EVENT_USER_ID	3045	X'00000BE5'
OBJEKT MQCACF_OBJECT_NAME	3046	X'00000BE6'
MQCAF_EVENT_Q_MGR	3047	X'00000BE7'
MQCACF_AUTH_INFO_NAMES	3048	X'00000BE8'
IDENTITA MQCAF_EVENT_APPL_IDENTITY	3049	X'00000BE9'
NÁZEV_APL_OBJEKTU MQCACF_APPL_NAME	3050	X'00000BEA'
PŮVOD MQCACF_EVENT_APPL_ORIGIN	3051	X'00000BEB'
MQCACF_SUBSCRIPTION_NAME	3052	X'00000BEC'
NÁZEV MQCACF_REG_SUB_NAME	3053	X'00000BED'
MQCACF_SUBSCRIPTION_IDENTITY	3054	X'00000BEE'
MQCAF_REG_SUB_IDENTITY	3055	X'00000BEF'
MQCACF_SUBSCRIPTION_USER_DATA	3056	X'00000BF0'
MQCACF_REG_SUB_USER_DATA	3057	X'00000BF1'
MQCAF_APPL_TAG	3058	X'00000BF2'
MQCACF_DATA_SET_NAME	3059	X'00000BF3'
DATUM ZAHÁJENÍ MQCACF_UOW_START_DATE	3060	X'00000BF4'
DOBA SPUŠTĚNÍ MQCACF_UOW_START_TIME	3061	X'00000BF5'
DATUM ZAHÁJENÍ ÚLOHY MQCACFF_UOW_LOG_START_DATE	3062	X'00000BF6'
ČAS ZAHÁJENÍ ÚLOHY MQCACFF_UOW_LOG_START_TIME	3063	X'00000BF7'
MQCACF_UOW_LOG_NÁZEV_ROZŠÍŘENÍ	3064	X'00000BF8'
NÁZVY MQCACFF_PRINCIPAL_ENTITY_NAMES	3065	X'00000BF9'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
NÁZVY FQCACF_GROUP_ENTITY_NAMES	3066	X'00000BFA'
MQCACF_JMÉNO_PROFILU	3067	X'00000BFB'
MQCACF_ENTITY_NAME	3068	X'00000BFC'
KOMPONENTA MQCACF_SERVICE_	3069	X'00000BFD'
MQCAF_RESPONSE_Q_MGR_NAME	3070	X'00000BFE'
MQCACF_AKTUÁLNÍ_NÁZEV_AKTUÁLNÍHO_PROTOKOLU	3071	X'00000BFF'
NÁZEV SOUBORU MQCACF_RESTART_LOG_EXTENT_NAME	3072	X'00000C00'
MQCACF_MEDIA_LOG_NÁZEV_ROZŠÍŘENÍ	3073	X'00000C01'
MQCAF_LOG_CESTA	3074	X'00000C02'
MQCAF_COMMAND_MQSC	3075	X'00000C03'
MQCACF_Q_MGR_CPF	3076	X'00000C04'
MQCACF_USAGE_LOG_RBA	3078	X'00000C06'
MQCACFF_USAGE_LOG_LRSN	3079	X'00000C07'
MQCACF_COMMAND_SCOPE	3080	X'00000C08'
ID OKAKTUS	3081	X'00000C09'
MQCACF_PSB_NÁZEV	3082	X'00000C0A'
MQCAF_PST_ID	3083	X'00000C0B'
MQCACF_TASK_NUMBER	3084	X'00000C0C'
ID_TRANSAKCE_MQCQU	3085	X'00000C0D'
MQCACF_Q_MGR_UOW_ID	3086	X'00000C0E'
MQCACFF_NÁZEV_PŮVODNÍ_JMÉNO	3088	X'00000C10'
INFORMACE O PROSTŘEDÍ MQCACF_ENV_	3089	X'00000C11'
MQCACF_SECURITY_PROFILE	3090	X'00000C12'
DATUM_KONFIGURACE_MQCCFF_	3091	X'00000C13'
DOBA KONFIGURACE_MQCACF_KONFIGURACE	3092	X'00000C14'
MQCACF_FROM_CF_STRUC_NAME	3093	X'00000C15'
NÁZEV OBJEKTU MQCACFF_TO_CF_STRUCT	3094	X'00000C16'
MQCAF_CF_STRUCKY_NÁZVY	3095	X'00000C17'
MQCAF_FAIL_DATE	3096	X'00000C18'
MQCAF_FAIL_TIME	3097	X'00000C19'
MQCAF_BACKUP_DATUM	3098	X'00000C1A'
ČAS ZÁLOHOVÁNÍ MQCACF_BACKUP_TIME	3099	X'00000C1B'
MQCACF_NÁZEV_SYSTÉMU	3100	X'00000C1C'
MQCACF_CF_STRUC_BACKUP_START	3101	X'00000C1D'
MQCAF_CF_STRUC_BACKUP_END	3102	X'00000C1E'
MQCACF_CF_STRUC_LOG_Q_MGRS	3103	X'00000C1F'
TŘÍDA MQCACFF_FROM_STORAGE_CLASS	3104	X'00000C20'
TŘÍDA MQCAF_TO_STORAGE_CLASS	3105	X'00000C21'
MQCACF_STORAGE_CLASS_NAMES,	3106	X'00000C22'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_DSG_NAME	3108	X'00000C24'
MQCACF_DB2_NAME	3109	X'00000C25'
MQCACF_SYSP_CMD_ID_UŽIVATELE	3110	X'00000C26'
MQCACF_SYSP_OTMA_SKUPINA	3111	X'00000C27'
MQCACF_SYSP_OTMA_MEMBER, ČLEN	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_CORRELAC_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'
SLUŽBA MQCACF_SYSP_SERVICE	3123	X'00000C33'
MQCACF_FROM_LISTENER_NAME	3124	X'00000C34'
MQCAF_TO_LISTENER_NAME	3125	X'00000C35'
MQCACF_FROM_SERVICE_NAME	3126	X'00000C36'
SLUŽBA MQCACF_TO_SERVICE_NAME	3127	X'00000C37'
MQCACFF_POSLEDNÍ_DATUM_ČAS	3128	X'00000C38'
MQCAF_POSLEDNÍ_ČAS_SPL.	3129	X'00000C39'
MQCAF_POSLEDNÍ_DATUM_GET_DATUM	3130	X'00000C3A'
MQCAF_GET_TIME (ČAS)	3131	X'00000C3B'
DATUM OPERACE MQCACF_OPERATION_DATE	3132	X'00000C3C'
ČAS OPERACE MQCACF_OPERATION_TIME	3133	X'00000C3D'
SOUBOR MQCACFF_ACTIVITY_DESC	3134	X'00000C3E'
DATA OBJEKTU MQCACFF_APPL_IDENTITY_DATA	3135	X'00000C3F'
MQCACF_APPL_ORIGIN_DATA	3136	X'00000C40'
MQCAF_PUT_DATE	3137	X'00000C41'
MQCAF_PUT_ČAS	3138	X'00000C42'
MQCAF_REPLY_TO_Q	3139	X'00000C43'
FUNKCE MQCACF_REPLY_TO_Q_MGR	3140	X'00000C44'
NÁZEV QCACF_RESOLVED_Q_NAME	3141	X'00000C45'
MQCAF_STRUCT_ID	3142	X'00000C46'
NÁZEV MQCACF_VALUE_NAME	3143	X'00000C47'
DATUM ZAHÁJENÍ SLUŽBY MQCACF_SERVICE_	3144	X'00000C48'
ČAS SPUŠTĚNÍ SLUŽBY MQCACF_SERVICE_	3145	X'00000C49'
MQCACF_SYSP_OFFLINE_RBA	3146	X'00000C4A'

Tabulka 40. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCACF_SYSP_ARCHIVE_PFX2	3147	X'00000C4B'
MQCACF_SYSP_ARCHIVE_UNIT2	3148	X'00000C4C'
NÁZEV SOUBORU MQCAF_TO_TOPIC_NAME	3149	X'00000C4D'
MQCACF_FROM_TOPIC_NAME	3150	X'00000C4E'
NÁZVY MQCACFF_TOPIC_NAMES	3151	X'00000C4F'
SUB_NAME MQCACF_SUB_NAME	3152	X'00000C50'
MQCACF_DESTINATION_Q_MGR	3153	X'00000C51'
CÍL 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_POSLEDNÍ_DATUM_OBL. DATUM	3161	X'00000C59'
MQCACFF_POSLEDNÍ_ČASOVÉ_PÁSMO	3162	X'00000C5A'
MQCACF_FROM_SUB_NAME	3163	X'00000C5B'
NÁZEV MQCACF_TO_SUB_NAME	3164	X'00000C5C'
ČAS MQCACFF_LAST_MSG_TIME	3167	X'00000C5F'
DATUM MQCACF_LAST_MSG_DATE	3168	X'00000C60'
PODBOD MQCACF_SUBSCRIPTION_POINT	3169	X'00000C61'
FILTR MQCACF_FILTER	3170	X'00000C62'
MQCAF_NONE	3171	X'00000C63'
NÁZVY ADMINISTRATIVNÍCH_TÉMAŮ_MQCACFF_	3172	X'00000C64'
MQCACFF_ROUTING_FINGER_PRINT	3173	X'00000C65'
POPIS MQCACFF_APPL_DES	3174	X'00000C66'
POČÁTEČNÍ_DATUM MQCACF_Q_MGR_START_DATE	3175	X'00000C67'
DOBA SPUŠTĚNÍ MQCACFF_Q_MGR_START_TIME	3176	X'00000C68'
FUNKCE MQCACFF_FROM_COMM_INFO_NAME	3177	X'00000C69'
MQCACF_TO_COMM_INFO_INFO	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'
MQCAF_CF_SMDS	3183	X'00000C6F'
DATUM_OBNOVY MQCACF_FM	3184	X'00000C70'
DOBA OBNOVY_MQCACF_OVERTIME	3185	X'00000C71'
MQCACF_CF_SMDSCONN	3186	X'00000C72'
NÁZEV OBJEKTU MQCACF_CF_STRUC_NAME	3187	X'00000C73'
MQCACF_ALTERNATE_USERID.	3188	X'00000C74'
MQCACF_CHAR_ATTRS	3189	X'00000C75'
OBJEKT MQCACF_DYNAMIC_Q_NAME	3190	X'00000C76'

Tabulka 40. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCACF_HOST_NAME	3191	X'00000C77'
MQCACF_MQCB_NAME	3192	X'00000C78'
MQCACF_OBJECT_STRING	3193	X'00000C79'
MQCAF_RESOLV_LOCAL_Q_MGR	3194	X'00000C7A'
NÁZEV SOUBORU MQCACFF_RESOLV_LOCAL_Q_NAME	3195	X'00000C7B'
SOUBOR MQCACFF_RESOLVED_OBJECT_STRING	3196	X'00000C7C'
FUNKCE MQCACF_RESOLVED_Q_MGR	3197	X'00000C7D'
ŘETĚZEC_VÝBĚRU_MQCACF\	3198	X'00000C7E'
MQCACF_XA_INFO	3199	X'00000C7F'
FUNKCE MQCACF_APPL_FUNCTION	3200	X'00000C80'
MQCACFXQHL_VZDÁLENÝ_NÁZEV_VZDÁLENÉ_FRONTY	3201	X'00000C81'
SUBRUTINA MQCACF_XQH_REMOTE_Q_MGR	3202	X'00000C82'
MQCACFXQQ_PUT_ČAS	3203	X'00000C83'
MQCACFXQQ_PUT_DATE	3204	X'00000C84'
ZPRÁVY MQCACFF_EXCL_OPERATOR_MESSAGES	3205	X'00000C85'
IDENTIFIKÁTOR MQCACF_CSP_USER_IDENTIFIER	3206	X'00000C86'
MQCAF_AMQP_CLIENT_ID	3207	X'00000C87'
NÁZEV_EXTENT_PROTOKOLU_MQCACF_ARCHIVE_LOG_	3208	X'00000C88'
DATUM A ČAS MQCACF_APPL_IMMOVABLE_DATE	3209	X'00000C89'
ČAS MQCACF_APPL_IMMOVABLE_TIME	3210	X'00000C8A'
MQCACF_NHA_NÁZEV_INSTANCE	3211	X'00000C8B'
MQCAF_LIST_USED	3211	X'00000C8B'

MQCACH_ * (Typy parametrů kanálu znaků v příkazovém kanálu)

Tabulka 41. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQCACH_FIRST	3501	X'00000DAD'
NÁZEV_KANÁLU_MQCACHE_NAME	3501	X'00000DAD'
MQCACH_DESC	3502	X'00000DAE'
NÁZEV_MODELU_MQCACHE_NAME	3503	X'00000DAF'
NÁZEV OBJEKTU MQCACH_TP_NAME	3504	X'00000DB0'
MQCACH_ XMIT_Q_NÁZEV	3505	X'00000DB1'
NÁZEV PŘIPOJENÍ 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'
NÁZEV MQCACHE_RCV_EXIT_NAME	3511	X'00000DB7'
NÁZVY KANÁLŮ MQCACH_CHANNEL_NAMES	3512	X'00000DB8'

Tabulka 41. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
ID_UŽIVATELE_MQCACH_	3517	X'00000DBD'
HESLO MQCACH_PASSWORD	3518	X'00000DBE'
LOKÁLNÍ ADRESA MQCACHE_LOCAL_ADDRESS	3520	X'00000DC0'
LOKÁLNÍ NÁZEV MQCACH_LOCAL_NAME	3521	X'00000DC1'
ČAS MQCACH_LAST_MSG_TIME	3524	X'00000DC4'
MQCACH_LAST_MSG_DATE	3525	X'00000DC5'
MQCACH_MCA_USER_ID	3527	X'00000DC7'
DOBA POČÁTKU MQCACH_CHANNEL_START_TIME	3528	X'00000DC8'
POČÁTEČNÍ_DATUM_ZAHÁJENÍ MQCACH_CHANNELY	3529	X'00000DC9'
NÁZEV ÚLOHY MQCACH_MCA_JOB_NAME	3530	X'00000DCA'
MQCACH_LAST_LUWID	3531	X'00000DCB'
MQCACHE_AKTUÁLNÍ_IDENTIFIKÁTOR-LUW	3532	X'00000DCC'
NÁZEV_FORMÁTU_MQCACHE_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_ID_UŽIVATELE	3549	X'00000DDD'
MQCACH_SSL_CERT_ISSUER_NAME	3550	X'00000DDE'
MQCACH_ -LUM_NÁZEV	3551	X'00000DDF'
MACK_ADRESA_IP_SERVERU	3552	X'00000DE0'
MQCACH_TCP_NAME	3553	X'00000DE1'
NÁZEV_MODULU_LISTENER MQCACH_LISTENER	3554	X'00000DE2'
POPIS_NASLOUCHÁNÍ MQCACHE_LISTS	3555	X'00000DE3'
DATUM_ZAHÁJENÍ PŘÍJMU MQCACH_LISTENER_	3556	X'00000DE4'
DOBA SPUŠTĚNÍ PŘÍKAZU MQCACH_LISTENER_TIME	3557	X'00000DE5'
MQCACH_SSL_KEY_RESET_DATE	3558	X'00000DE6'
MQCACH_SSL_KEY_RESET_TIME	3559	X'00000DE7'
POUŽITÁ MQCACH_LAST_USED	3559	X'00000DE7'

MQCADSD_ * (CICS informační záhlaví ADS Descriptors)

Tabulka 42. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCADSD_NONE	0	X'00000000'
MQCADSD_SEND	1	X'00000001'
MQCADSD_RECV	16	X'00000010'
FORMÁT ZPRÁVY MQCADSD_MSGFORMAT	256	X'00000100'

MQCAFTY_ * (Hodnoty afinity připojení)

Tabulka 43. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCAFTY_NONE	0	X'00000000'
PREFEROVANÉ MQCAFTY_	1	X'00000001'

MQCAMO_ * (Typy parametrů monitorování znaků pro formát příkazů)

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'
MQCAMO_START_DATE	2711	X'00000A97'
ČAS SPUŠTĚNÍ MQCAMO_START_TIME	2712	X'00000A98'
MQCAMO_LAST_USED	2712	X'00000A98'

MQCBC_ * (struktura konstant MQCBC)

Tabulka 45. Konstrukce konstant

Název	Struktura
ID_KONSTRUKCE_MQCBC_	"CBC¬"
POLE 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'

Tabulka 46. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
AKTUÁLNÍ_VERZE 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_ * (MQCBC constants Callback type)

Tabulka 48. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBCT_START_CALL	1	X'00000001'
MQCBCT_STOP_CALL	2	X'00000002'
VOLÁNÍ MQCBCT_REGISTER_CALL	3	X'00000003'
VOLÁNÍ MQCBCT_DEREGISTER_CALL	4	X'00000004'
VOLÁNÍ MQCBCT_EVENT_CALL	5	X'00000005'
MQCBCT_MSG_REMOVED	6	X'00000006'
MQCBCT_MSG_NOV_REMOVED	7	X'00000007'

MQCBD_ * (struktura konstant MQCBD)

Tabulka 49. Konstrukce konstant	
Název	**Struktura**
MQCBD_STRUCTION_ID	"CBD¬"
POLE 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'
AKTUÁLNÍ_VERZE MQCBD_	1	X'00000001'

MQCBDO_ * (operace MQCBD constants Callback Options)

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

MQCBO_ * (Vytvoření-Volby balíku pro objekt 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_BLOKOVÁNO	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 Tj. 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'
VAROVÁNÍ MQCC_WARNING	1	X'00000001'
SELHÁNÍ MQCC_FAILED	2	X'00000002'
NEZNÁMÉ MQCC_UNKNOWN	-1	X'FFFFFFFF'

MQCCSI_ * (identifikátory kódované znakové sady)

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

MQCCT_ * (CICS -volby konverzačních úloh záhlaví informací)

Tabulka 56. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCKT_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 V 9.2.0 MQCD_VERSION_12	12	X'0000000C'
z/OS V 9.2.0 MQCD_CURRENT_VERSION	12	X'0000000C'
MQCD_LENGTH_4	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_5	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_6	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_7	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_8	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_9	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_10	(value differs by platform or version)	(value differs by platform or version)
MQCD_LENGTH_11	(value differs by platform or version)	(value differs by platform or version)
z/OS V 9.2.0 MQCD_LENGTH_12	(value differs by platform or version)	(value differs by platform or version)
AKTUÁLNÍ_DÉLKA_MQCD_	(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
KONVERZE MQCDC_SENDER_CONVERSION	1	X'00000001'
KONVERZE MQCDC_NO_SENDER_CONVERSION	0	X'00000000'

MQCERT_ * (typ zásady ověření platnosti 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_ * (parametry schopností)

Tabulka 59. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCF_NONE	0	X'00000000'
MQCF_DIST_LISTS	1	X'00000001'

MQCFAC_ * (CICS -prostředek záhlaví informací)

Tabulka 60. Konstantní názvy a hodnoty

Název	Hexadecimální hodnota
MQCFAC_NONE	X'00...00' (8 nul)
MQCFAC_NON_ARRAY	'\0','\0',... (8 nul)

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 parametrů bajtového řetězce formátu příkazu)

Tabulka 62. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFC_ * (Volby ovládacího prvku záhlaví příkazu format)

Tabulka 63. Hodnoty konstant

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

MQCFGR_ * (struktura parametrů skupiny příkazů)

Tabulka 64. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFGR_STRUST_LENGTH	16	X'00000010'

MQCFH_ * (struktura záhlaví příkazového formátu)

Tabulka 65. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
DÉLKA OBJEKTU MQCFH_STRU_LENGTH	36	X'00000024'
MQCFH_VERSION_1	1	X'00000001'
MQCFH_VERSION_2	2	X'00000002'
MQCFH_VERSION_3	3	X'00000003'
AKTUÁLNÍ_VERZE MQCFH_AKTUÁLNÍ_VERZE	3	X'00000003'

MQCFIF_ * (struktura parametrů filtru celého čísla formátu příkazu)

Tabulka 66. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
DÉLKA OBJEKTU MQCFIF_STRU_LENGTH	20	X'00000014'

MQCFIL_ * (Struktura konfiguračního parametru celého čísla formátu příkazu)

Tabulka 67. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
PEVNÁ DÉLKA_OBJEKTU_MQCFIL_FIX_FIXED	16	X'00000010'

MQCFIL64_* (Struktura konfiguračního parametru 64bitového celočíselného seznamu příkazů)

Tabulka 68. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIN_ * (struktura parametrů celého čísla příkazu)

Tabulka 69. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIN_STRUST_LENGTH	16	X'00000010'

MQCFIN64_* (Struktura parametrů 64bitového celočíselného formátu příkazů)

Tabulka 70. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIN64_STRUC_LENGTH	24	X'00000018'

MQCFO_ * (Volby úložiště a volby příkazu Odebrat fronty pro formát příkazů pro formát příkazů a volby Odebrat fronty)

Volby úložiště pro aktualizaci formátu příkazů

Tabulka 71. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFO_REFRESH_REPOSITORY_YES	1	X'00000001'
MQCFO_REFRESH_REPOSITORY_NO	0	X'00000000'

Volby příkazu pro odebrání 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_GREATER	4	X'00000004'
MQCFOP_NOT_LESS	6	X'00000006'
MQCFOP_NOT_EQUAL	5	X'00000005'
MQCFOP_NOT_GREATER	3	X'00000003'
MQCFOP_LIKE	18	X'00000012'
MQCFOP_NOT_LIKE	21	X'00000015'
MQCFOP_CONTAINS	10	X'0000000A'
MQCFOP_EXCLUDES	13	X'0000000D'
MQCFOP_CONTAINS_GEN	26	X'0000001A'
MQCFOP_EXCLUDES_GEN	29	X'0000001D'

MQCFR_ * (zotavení prostředku CF)

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

MQCFSF_ * (Struktura parametru filtru řetězce příkazového řetězce)

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

MQCFSL_ * (Struktura parametrů řetězce formátu příkazu)

Tabulka 76. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
PEVNÁ DÉLKA_STRUKTURA_MQCFSL_STRUCT	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_ * (Stav prostředku CF příkazu format)

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'
SELHÁNÍ MQCFSTATUS_FAILED	4	X'00000004'
MQCFSTATUS_NONE	5	X'00000005'
MQCFSTATUS_UNKNOWN	6	X'00000006'
FUNKCE MQCFSTATUS_ADMIN_NECOMPLETE	20	X'00000014'
MQCFSTATUS_NEVER_USED	21	X'00000015'
MQCFSTATUS_NO_BACKUP	22	X'00000016'
SELHÁNÍ MQCFSTATUS_NOT_FAILED	23	X'00000017'
MQCFSTATUS_NOT_RECOVERABLE	24	X'00000018'
CHYBA MQCFSTATUS_XES_ERROR	25	X'00000019'

MQCFT_ * (Typy formátu příkazů pro strukturu)

Tabulka 79. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFT_NONE	0	X'00000000'
PŘÍKAZ MQCFT_COMMAND	1	X'00000001'
ODEZVA MQCFT_RESPONSE	2	X'00000002'
MQCFT_INTEGER	3	X'00000003'
ŘETĚZEC MQCFT_STRING	4	X'00000004'
MQCFT_INTEGER_LIST	5	X'00000005'
SEZNAM_NÁZVŮ_MQCFT_LIST	6	X'00000006'
UDÁLOST MQCFT_EVENT	7	X'00000007'
UŽIVATEL MQCFT_USER	8	X'00000008'
MQCFT_BYTE_STRING	9	X'00000009'
MQCFT_TRACE_ROUTE	10	X'0000000A'

Tabulka 79. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SESTAVA MQCFT_REPORT	12	X'0000000C'
MQCFT_INTEGER_FILTER	13	X'0000000D'
FILTR MQCFT_STRING_FILTER	14	X'0000000E'
MQCFT_BYTE_ŘETĚZEC_FILTRU	15	X'0000000F'
MQCFT_COMMAND_XR	16	X'00000010'
ZPRÁVA MQCFT_XR_MSG	17	X'00000011'
POLOŽKA MQCFT_XR_ITEM	18	X'00000012'
SOUHRN MQCFT_XR_SUMMARY	19	X'00000013'
SKUPINA MQCFT_GROUP	20	X'00000014'
STATISTIKA MQCFT_STATISTICS	21	X'00000015'
ÚČETNÍ_ÚČT_VYR.	22	X'00000016'
MQCFT_INTEGER64	23	X'00000017'
MQCFT_INTEGER64_LIST	25	X'00000019'

MQCFTYPE_ * (Typy CF příkazového formátu)

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

MQCFUNC_ * (funkce záhlaví informací produktu CICS)

Tabulka 81. Konstrukce konstant	
Název	**Struktura**
MQCFUNC_MQCONN	"CONN"
FUNKCE MQCFUNC_MQGET	"GET¬"
MQCFUNC_MQINQ	"INQ¬"
MQCFUNC_MQOPEN	"OPEN"
MQCFUNC_MQPUT	"PUT¬"
MQCFUNC_MQPUT1	"PUT1"
MQCFUNC_NONE	"¬¬¬¬"
MQCFUNKCE_MQCONN_ARRAY	'C','O','N','N'
MQCFUNC_MQGET_ARRAY	'G','E','T','¬'
MQCFUNC_MQINQ_ARRAY	'I','N','Q','¬'
MQCFUNC_MQOPEN_ARRAY	'O','P','E','N'
MQCFUNC_MQPUT_ARRAY	'P','U','T','¬'
MQCFUNC_MQPUT1_ARRAY	'P','U','T','1'
MQCFUNC_NON_ARRAY	'¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQCGWI_ * (CICS záhlaví informací Get Wait Interval)

Tabulka 82. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
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_ * (Nejistý formát příkazu)

Tabulka 84. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCHIDS_NOT_NEOVĚŘENÝ	0	X'00000000'
NEJISTÉ MQCHIDS_NEOVĚŘENÝ	1	X'00000001'

MQCHLD_ * (pozice kanálu příkazového řádku)

Tabulka 85. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCHLD_ALL	-1	X'FFFFFFFF'
VÝCHOZÍ HODNOTA MQCHLD_DEFAULT	1	X'00000001'
MQCHLD_SHARED	2	X'00000002'
MQCHLD_PRIVATE	4	X'00000004'
SDÍLENOU MQCHLD_FIXSHARED	5	X'00000005'

MQCHS_ * (Stav kanálu příkazového řádku)

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 příkazového kanálu sdíleného restartování)

Tabulka 87. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCHSH_RESTART_NO	0	X'00000000'
MQCHSH_RESTART_YES	1	X'00000001'

MQCHSR_ * (Volby ukončení kanálu příkazového kanálu)

Tabulka 88. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCHSR_STOP_NOT_REQUESTED	0	X'00000000'
MQCHSR_STOP_REQUESTED	1	X'00000001'

MQCHSSTATE_ * (Substavy kanálů příkazu format)

Tabulka 89. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCHSSTATE_OTHER	0	X'00000000'
MQCHSSTATE_END_OF_BATCH	100	X'00000064'
MQCHSSTATE_ODESÍLÁNÍ	200	X'000000C8'
MQCHSSTATE_RECEIVING	300	X'0000012C'
MQCHSSTATE_SERIALIZACE	400	X'00000190'
MQCHSSTATE_RESYNCHING	500	X'000001F4'
MQCHSSTATE_HEARTBEAT	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'
SERVER 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	1	X'00000001'
SERVER MQCHT_SERVER	2	X'00000002'

Tabulka 90. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PŘÍJEMCE MQCHT_RECEIVER	3	X'00000003'
MQCHT_REQUESTER	4	X'00000004'
MQCHT_ALL	5	X'00000005'
MQCHT_CLNTCONN	6	X'00000006'
FUNKCE MQCHT_SVRCONN	7	X'00000007'
SOUBOR MQCHT_CLURCVR	8	X'00000008'
MQCHT_CLUSDR	9	X'00000009'

MQCHTAB_ * (Typy tabulek kanálu s formátem příkazu)

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. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCI_NONE	X'00...00' (24 nul)
MQCI_NON_ARRAY	'\0','\0',... (24 nul)
MQCI_NEW_SESSION	X'414D5121...'
MQCI_NEW_SESSION_ARRAY	'\x41','\x4D','\51','\x21',...

MQCIH_ * (struktura a příznaky záhlaví informačního obsahu produktu CICS)

Struktura záhlaví informací produktu CICS

Tabulka 93. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQCIH_	"CIH¬"
POLE 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'
AKTUÁLNÍ_VERZE MQCIH_CURRENT_VERSION	2	X'00000002'
MQCIH_LENGTH_1	164	X'000000A4'
MQCIH_LENGTH_2	180	X'000000B4'
AKTUÁLNÍ_DÉLKA MQCIH_CURRENT_LENGTH	180	X'000000B4'

Parametry záhlaví informací produktu 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_REPLYY_WITHUT_NULL	2	X'00000002'
MQCIH_REPLY_WITH_NULL	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_ * (Výmaz formátu příkazu-obor názvů témat)

Tabulka 97. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCLRS_LOCAL	1	X'00000001'
MQCLRS_GLOBAL	2	X'00000002'

MQCLRT_ * (typ příkazu Vymazat typ řetězce tématu)

Tabulka 98. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCLRT_RETAINED	1	X'00000001'

MQCLT_ * (CICS -typy odkazů záhlaví informací)

Tabulka 99. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCLT_PROGRAM	1	X'00000001'
TRANSAKCE MQCLT_TRANSACTION	2	X'00000002'

MQCLWL_ * (Vytížení 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

“DefClusterXmitQueueType (MQLONG)” na stránce 808
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-Dotaz na atributy objektu” na stránce 699
Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

MQCMD_ * (Příkazové kódy)

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'
PROCES MQCMD_CHANGE_PROCESS	3	X'00000003'
PROCES MQCMD_COPY_PROCESS	4	X'00000004'
PROCES OBJEKTU MQCMD_CREATE_PROCESS	5	X'00000005'
MQCMD_DELETE_PROCESS	6	X'00000006'
ZPRACOVÁNÍ PŘÍKAZU 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'
FUNKCE MQCMD_RESET_Q_STATS	17	X'00000011'
NÁZVY MQCMD_INQUIRE_Q_NAMES	18	X'00000012'
NÁZVY MQCMD_INQUIRE_PROCESS_NAMES	19	X'00000013'
NÁZVY MQCMD_INQUIRE_CHANNEL_NAMES	20	X'00000014'
MQCMD_CHANGE_KANÁL	21	X'00000015'
MQCMD_COPY_CHANNEL	22	X'00000016'
MQCMD_CREATE_CHANNEL	23	X'00000017'

Tabulka 102. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCMD_DELETE_CHANNEL	24	X'00000018'
MQCMD_INQUIRE_CHANNEL	25	X'00000019'
KANÁL 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'
MODUL LISTENER MQCMD_START_CHANNEL_LISTENER	31	X'0000001F'
MQCMD_CHANGE_SEZNAM NÁZVŮ	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'
NÁZVY MQCMD_INQUIRE_NAMELIST_NAMES	37	X'00000025'
MQCMD_ESCAPE	38	X'00000026'
MQCMD_RESOLVE_CHANNEL	39	X'00000027'
MQCMD_PING_Q_MGR	40	X'00000028'
STAV MQCMD_INQUIRE_Q_STATUS	41	X'00000029'
STAV 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'
UDÁLOST MQCMD_CHANNEL_EVENT	46	X'0000002E'
MQCMD_DELETE_PUBLICATION	60	X'0000003C'
MQCMD_DEREGISTER_PUBLISHER	61	X'0000003D'
ODBĚRATEL MQCMD_DEREGISTER_ODBĚRATEL	62	X'0000003E'
MQCMD_PUBLISH	63	X'0000003F'
MQCMD_REGISTER_PUBLISHER	64	X'00000040'
MQCMD_REGISTER_SUBSCRIBER	65	X'00000041'
MQCMD_REQUEST_UPDATE	66	X'00000042'
MQCMD_BROKER_INTERNAL	67	X'00000043'
ZPRÁVA MQCMD_ACTIVITY_MSG	69	X'00000045'
MQCMD_INQUIRE_CLUSTER_Q_MGR	70	X'00000046'
KLASTR MQCMD_RESUME_Q_MGR_CLUSTER	71	X'00000047'
MQCMD_SUSPEND_Q_MGR_CLUSTER	72	X'00000048'
KLASTR MQCMD_REFRESH_CLUSTER	73	X'00000049'
KLASTR MQCMD_RESET_CLUSTER	74	X'0000004A'
MQCMD_TRACE_ROUTE	75	X'0000004B'
ZABEZPEČENÍ MQCMD_REFRESH_SECURITY	78	X'0000004E'

Tabulka 102. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
NÁZVY MQCMD_INQUIRE_AUTH_INFO_NAMES	84	X'00000054'
MQCMD_INQUIRE_CONNECTION	85	X'00000055'
MQCMD_STOP_CONNECTION	86	X'00000056'
MQCMD_INQUIRE_AUTH_RECS	87	X'00000057'
MQCMD_INQUIRE_ENTITY_AUTH	88	X'00000058'
MQCMD_DELETE_AUTH_REC	89	X'00000059'
MQCMD_SET_AUTH_REC	90	X'0000005A'
UDÁLOST MQCMD_LOGGER_EVENT	91	X'0000005B'
FUNKCE MQCMD_RESET_Q_MGR	92	X'0000005C'
MQCMD_CHANGE_LISTENER	93	X'0000005D'
MQCMD_COPY_LISTENER	94	X'0000005E'
MODUL MQCMD_CREATE_LISTENER	95	X'0000005F'
MQCMD_DELETE_LISTENER	96	X'00000060'
MQCMD_INQUIRE_LISTENER	97	X'00000061'
STAV OBJEKTU MQCMD_INQUIRE_LISTENER_STATUS	98	X'00000062'
MQCMD_COMMAND_EVENT	99	X'00000063'
MQCMD_CHANGE_ZABEZPEČENÍ	100	X'00000064'
MQCMD_CHANGE_CF_STRUKTURY	101	X'00000065'
TŘÍDA MQCMD_CHANGE_STG_CLASS	102	X'00000066'
TRASOVÁNÍ MQCMD_CHANGE_TRACE	103	X'00000067'
PROTOKOL MQCMD_ARCHIVE_LOG	104	X'00000068'
MQCMD_BACKUP_CF_STRUKTURY	105	X'00000069'
OBLAST MQCMD_CREATE_BUFFER_POOL	106	X'0000006A'
MQCMD_CREATE_PAGE_SET	107	X'0000006B'
MQCMED_CREATE_CF_STRUKTURY	108	X'0000006C'
TŘÍDA MQCMD_CREATE_STG_CLASS	109	X'0000006D'
MQCMD_COPY_CF_STRUKTURY	110	X'0000006E'
TŘÍDA MQCMD_COPY_STG_CLASS	111	X'0000006F'
MQCMD_DELETE_CF_STRUKTURY	112	X'00000070'
TŘÍDA MQCMD_DELETE_STG_CLASS	113	X'00000071'
MQCMD_INQUIRE_ARCHIVE	114	X'00000072'
MQCMD_INQUIRE_CF_STRUKTURY	115	X'00000073'
STAV_ROZHRANÍ_MQCMD_INQUIRE_CF_STRU_STATUS	116	X'00000074'
MQCMD_INQUIRE_CMD_SERVER	117	X'00000075'

Tabulka 102. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
TŘÍDA MQCMD_INQUIRE_STG_CLASS	122	X'0000007A'
MQCMD_INQUIRE_SYSTEM	123	X'0000007B'
MQCMD_INQUIRE_THREAD	124	X'0000007C'
TRASOVÁNÍ MQCMD_INQUIRE_TRACE	125	X'0000007D'
MQCMD_INQUIRE_USAGE	126	X'0000007E'
MQCMD_MOVE_Q	127	X'0000007F'
VYPRAVA_OBNOVY_MQCMD_BSD	128	X'00000080'
POUŽITÁ_OBNOVY_PROSTŘEDÍ_MQCMD_	129	X'00000081'
MQCMD_RESET_TPIPE	130	X'00000082'
MQCMD_RESOLVE_NEOVĚŘENÝ	131	X'00000083'
MQCMD_RESUME_Q_MGR	132	X'00000084'
ZABEZPEČENÍ MQCMD_REVERIFY_SECURITY	133	X'00000085'
MQCMD_SET_ARCHIVE	134	X'00000086'
PROTOKOL MQCMD_SET_LOG	136	X'00000088'
SYSTÉM MQCMD_SET_SYSTEM	137	X'00000089'
MQCMD_START_CMD_SERVER	138	X'0000008A'
MQCMD_START_Q_MGR	139	X'0000008B'
TRASOVÁNÍ MQCMD_START_TRACE	140	X'0000008C'
INICIALIZAČNÍ KANÁL MQCMD_STOP_CHANNEL_INIT	141	X'0000008D'
MODUL LISTENER MQCMD_STOP_CHANNEL_LISTENER	142	X'0000008E'
SERVER MQCMD_STOP_CMD_SERVER	143	X'0000008F'
MQCMD_STOP_Q_MGR	144	X'00000090'
TRASOVÁNÍ MQCMD_STOP_TRACE	145	X'00000091'
MQCMD_SUSPEND_Q_MGR	146	X'00000092'
NÁZVY OBJEKTŮ MQCMD_INQUIRE_CF_STRUC_STRU_NÁZVY	147	X'00000093'
MQCMD_INQUIRE_STG_CLASS_NAMES	148	X'00000094'
SLUŽBA MQCMD_CHANGE_SERVICE	149	X'00000095'
MQCMD_COPY_SERVICE	150	X'00000096'
SLUŽBA MQCMD_CREATE_SERVICE	151	X'00000097'
SLUŽBA MQCMD_DELETE_SERVICE	152	X'00000098'
SLUŽBA MQCMD_INQUIRE_SERVICE	153	X'00000099'
STAV MQCMD_INQUIRE_SERVICE_STATUS	154	X'0000009A'
SLUŽBA MQCMD_START_SERVICE	155	X'0000009B'
SLUŽBA MQCMD_STOP_SERVICE	156	X'0000009C'
FOND VYROVNÁVACÍCH PAMĚTÍ MQCMD_DELETE_BUFFER_NAME	157	X'0000009D'

Tabulka 102. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMD_DELETE_PAGE_SET	158	X'0000009E'
MQCMD_CHANGE_BUFFER_VYROVNÁVACÍ_PAMĚŤ	159	X'0000009F'
MQCMD_CHANGE_STRÁNKA_STRÁNKY_SADY	160	X'000000A0'
STAV MQCMD_INQUIRE_Q_MGR_STATUS	161	X'000000A1'
PROTOKOL MQCMD_CREATE_LOG	162	X'000000A2'
MQCMD_STATISTICS_MQI	164	X'000000A4'
MQCMD_STATISTICS_Q	165	X'000000A5'
KANÁL 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'
TÉMA MQCMD_CREATE_TOPIC	172	X'000000AC'
MQCMD_DELETE_TOPIC, TÉMA	173	X'000000AD'
MQCMD_INQUIRE_TOPIC	174	X'000000AE'
NÁZVY MQCMD_INQUIRE_TOPIC_NAMES	175	X'000000AF'
MQCMD_INQUIRE_SUBSCRIPTION	176	X'000000B0'
MQCMD_CREATE_SUBSCRIPTION	177	X'000000B1'
MQCMD_CHANGE_ODBĚR	178	X'000000B2'
ODBĚR MQCMD_DELETE_SUBSCRIPTION	179	X'000000B3'
MQCMD_COPY_SUBSCRIPTION	181	X'000000B5'
STAV MQCMD_INQUIRE_SUB_STATUS	182	X'000000B6'
STAV MQCMD_INQUIRE_TOPIC_STATUS	183	X'000000B7'
ŘETĚZEC MQCMD_CLEAR_TOPIC_STRING	184	X'000000B8'
STAV MQCMD_INQUIRE_PUBSUB_STATUS	185	X'000000B9'
MQCMD_PURGE_CHANNEL	195	X'000000C3'

MQCMDI_ * (Hodnoty příkazu Command Format Command Information Values)

Tabulka 103. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMDI_CMDSCOPY_ACCEPTED	1	X'00000001'
MQCMDI_CMDSCOPY_GENERATED	2	X'00000002'
MQCMDI_CMDSCOPY_COMPLETED	3	X'00000003'
MQCMDI_QSG_DISP_COMPLETED	4	X'00000004'
MQCMDI_COMMAND_ACCEPTED	5	X'00000005'
MQCMDI_CLUSTER_REQUEST_QUEUED	6	X'00000006'
MQCMDI_CHANNEL_INIT_STARTED	7	X'00000007'
MQCMDI_RECOVER_STARTED	11	X'0000000B'

Tabulka 103. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMDI_BACKUP_STARTED	12	X'0000000C'
MQCMDI_RECOVER_COMPLETED	13	X'0000000D'
MQCMDI_SEC_TIMER_ZERO	14	X'0000000E'
KONFIGURACE MQCMDI_REFRESH_CONFIGURATION	16	X'00000010'
CHYBA PŘI CHYBĚ MQCMDI_SEC_SIGNOFF_ERROR	17	X'00000011'
MQCMDI_IMS_BRIDGE_SUSPENDED	18	X'00000012'
MQCMDI_DB2_SUSPENDED	19	X'00000013'
MQCMDI_DB2_OBSOLETE_MSGS	20	X'00000014'
MQCMDI_SEC_UPPERCASE	21	X'00000015'
MQCMDI_SEC_MIXEDCASE	22	X'00000016'

MQCMDL_ * (Úrovně příkazů)

Tabulka 104. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCMDL_LEVEL_800	800
MQCMDL_LEVEL_801	801
MQCMDL_LEVEL_802	802
MQCMDL_LEVEL_900	900
MQCMDL_LEVEL_901	901
MQCMDL_LEVEL_902	902
MQCMDL_LEVEL_903	903
MQCMDL_LEVEL_904	904
MQCMDL_LEVEL_905	905
MQCMDL_LEVEL_910	910
MQCMDL_LEVEL_912	912
MQCMDL_LEVEL_913	913
MQCMDL_LEVEL_914	914
MQCMDL_LEVEL_915	915
MQCMDL_LEVEL_920	920
MQCMDL_LEVEL_921	921
MQCMDL_LEVEL_922	922
MQCMDL_LEVEL_923	923
MQCMDL_LEVEL_924	924
MQCMDL_LEVEL_925	925

MQCMHO_ * (Vytvoření voleb a struktury manipulátoru zprávy)

Vytvořit strukturu voleb zpracování zpráv

Tabulka 105. Konstrukce konstant	
Název	**Struktura**
MQCMHO_STRUCTION_ID	"CMHO"
MQCMHO_STRUC_ID_POLE	'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'
AKTUÁLNÍ_VERZE MQCMHO_AKTUÁLNÍ_VERZE	1	X'00000001'

Volby vytvoření popisovače zpráv

Tabulka 107. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMHO_DEFAULT_VALIDATION	0	X'00000000'
MQCMHO_NO_VALIDATION	1	X'00000001'
MQCMHO_VALIDATE	2	X'00000002'
MQCMHO_NONE	0	X'00000000'

MQCNO_ * (Volby připojení a struktura)

Struktura voleb připojení

Tabulka 108. Konstrukce konstant	
Název	**Struktura**
MQCNO_STRUCTION_ID	"CNO¬"
MQCNO_STRUC_ID_POLE	'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'
AKTUÁLNÍ_VERZE MQCNO_CURRENT_VERSION	5	X'00000005'

Volby připojení

Tabulka 110. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VAZBA MQCNO_STANDARD_BINDING	0	X'00000000'
VAZBA 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'
CQCNO_SHARED_BINDING	256	X'00000100'
VAZBA MQCNO_ISOLATED_BINDING	512	X'00000200'
MQCNO_LOCAL_BINDING.	1024	X'00000400'
VÁZÁNÍ 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'
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'
VÝBĚR MQCNO_USE_CD_SELECTION	1048576	X'00100000'
MQCNO_RECONNECT	16777216	X'01000000'
MQCNO_RECONNECT_AS_DEF	0	X'00000000'
FUNKCE MQCNO_RECONNECT_DISABLED	33554432	X'02000000'
FUNKCE 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'

Tabulka 111. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCO_QUIESCE	32	X'00000020'

MQCODL_ * (CICS -délka výstupních dat záhlaví informací)

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

MQCOMPRESS_* (Komprese kanálu)

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'
SYSTÉM MQCOMPRESS_SYSTEM	8	X'00000008'
MQCOMPRESS_ANY	268435455	X'0FFFFFFF'

MQCONNID_ * (Identifikátor připojení)

Tabulka 114. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCONNID_NONE	X'00...00' (24 nul)
MQCONNID_NO_ARRAY	'\0','\0',... (24 nul)

MQCOPY_ * (Volby kopie vlastnosti)

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'

Tabulka 116. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCQT_REMOTE_Q	3	X'00000003'
ALIAS MQCQT_Q_MGR_ALIAS	4	X'00000004'

MQCRC_ * (Návratové kódy záhlaví informačního záhlaví produktu CICS)

Tabulka 117. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCRC_OK	0	X'00000000'
CHYBA MQCRC_CICS_EXEC_ERROR	1	X'00000001'
CHYBA MQCRC_MQ_API_ERROR	2	X'00000002'
CHYBA MQCRC_BRIDGE_ERROR	3	X'00000003'
MQCRC_BRIDGE_ABEND	4	X'00000004'
UKONČENÍ MQCRC_APPLICATION_ABEND	5	X'00000005'
MQCRC_SECURITY_ERROR	6	X'00000006'
MQCRC_PROGRAM_NOT_AVAILABLE	7	X'00000007'
MQCRC_BRIDGE_TIMEOUT	8	X'00000008'
MQCRC_TRANSID_NOT_AVAILABLE	9	X'00000009'

MQCS_ * (MQCBC constants Consumer State)

Tabulka 118. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCS_NONE	0	X'00000000'
DOČASNÉ DOČASNÉ OBJEKTY MQCS_SUSPENDED_TEMPORARY	1	X'00000001'
AKCE MQCS_SUSPENDED_USER_ACTION	2	X'00000002'
POZASTAVENÉ MQCS_	3	X'00000003'
ZASTAVENÉ MQCS_	4	X'00000004'

MQCSC_ * (počáteční kódy záhlaví informačního záhlaví produktu CICS)

Tabulka 119. Konstrukce konstant	
Název	**Struktura**
MQCSC_START	"S¬¬¬"
POČÁTEČNÍ_DATA MQCSC_STARTDATA	"SD¬¬"
MQCSC_TERMINPUT	"TD¬¬"
MQCSC_NONE	"¬¬¬¬"
POLE MQCSC_START_ARRAY	'S','¬','¬','¬'
MQCSC_STARTDATA_ARRAY	'S','D','¬','¬'
MQCSC_TERMINPUT_ARRAY	'T','D','¬','¬'
MQCSC_NON_ARRAY	'¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQCSP_ * (Struktura parametrů zabezpečení připojení a typy ověřování)

Struktura parametrů zabezpečení připojení

Tabulka 120. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQCSP_STRUCT	"CSP¬"
MQCSP_STRUKRE_ID_POLE	'C','S','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 121. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCSP_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQCSP_CURRENT_VERSION	1	X'00000001'

Typy ověřování parametrů zabezpečení připojení

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

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. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCT_NONE	X'00...00' (128 nulových hodnot)
MQCT_NON_ARRAY	'\0','\0',... (128 nulových hodnot)

MQCTES_ * (CICS -konec úlohy záhlaví úlohy)

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

MQCTLO_ * (struktura voleb MQCTL a volby kontroly spotřebitele)

Struktura voleb MQCTL

Tabulka 126. Konstrukce konstant	
Název	**Struktura**
MQCTLO_STRUCTION_ID	"CTLO"
POLE 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'
AKTUÁLNÍ VERZE MQCTLO_VERSION	1	X'00000001'

Volby kontroly spotřebitele voleb MQCTL

Tabulka 128. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCTLO_NONE	0	X'00000000'
MQCTLO_THREAD_AFFINITY	1	X'00000001'
UVÁDĚNÍ MQCTLO_FAIL_IF_QUIESCING	8192	X'00002000'

MQCUOWC_ * (Řízení informační jednotky záhlaví produktu CICS -řízení práce)

Tabulka 129. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POUZE MQCUOWC_ONLY	273	X'00000111'
MQCUOWC_CONTINUE	65536	X'00010000'
NEJPRVE 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 výstupního parametru kanálu)

Tabulka 130. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQCXP_STRUCTION_ID	"CXP¬"
POLE 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'
AKTUÁLNÍ_VERZE MQCXP_CURRENT_VERSION	9	X'00000009'

MQDC_ * (Cílová třída)

Tabulka 132. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SPRAVOVANÝ MQDC_	1	X'00000001'
POSKYTNUTÝ MQDC_	2	X'00000002'

MQDCC_ * (Možnosti převodu a Masky a faktory)

Volby převodu

Tabulka 133. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PŘEVOD MQDCC_DEFAULT_VERSION	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'

Volby masky a faktory konverze

Tabulka 134. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MAQDCC_ZDROJOVÁ_MASKA	240	X'000000F0'

Tabulka 134. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MAQDCC_CÍLOVÁ_MASKA	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. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQDH_	"DH¬¬"
POLE MQDH_STRUC_ID_ARRAY	'D','H','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

MQDHF_ * (parametry záhlaví distribuce)

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

MQDISCONNECT_ * (typ příkazu Disconnect Types)

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

MQDL_ * (distribuční seznamy)

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

MQDLH_ * (struktura záhlaví nedoručených zpráv)

Tabulka 141. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQDLH_STRUCTION_ID	"DLH¬"
POLE MQDLH_STRUC_ID_ARRAY	'D','L','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 142. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDLH_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQDLH_CURRENT_VERSION	1	X'00000001'

MQDLV_ * (Persistit/Non-persistent Message Delivery)

Tabulka 143. 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_ * (Odstranění voleb a struktury zpracování zpráv)

Odstranit strukturu voleb obsluhy zprávy

Tabulka 144. Konstrukce konstant	
Název	**Struktura**
MQDMHO_STRUCTION_ID	"DMHO"
MQDMHO_STRUC_ID_POLE	'D','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 145. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMHO_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQDMHO_CURRENT_VERSION	1	X'00000001'

Volby odstranění popisovače zpráv

Tabulka 146. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMHO_NONE	0	X'00000000'

MQDMPO_ * (Odstranění vlastností a struktury vlastností zprávy)

Odstranit strukturu voleb vlastností zprávy

Tabulka 147. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQDMPO_	"DMPO"
POLE MQDMPO_STRUC_ID_ARRAY	'D','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 148. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDMPO_VERSION_1	1	X'00000001'
MQDMPO_AKTUÁLNÍ_VERZE	1	X'00000001'

Volby odstranění vlastností zprávy

Tabulka 149. 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_* (DNS WLM)

Tabulka 150. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDNSWLM_NO	0	X'00000000'
MQDNSWLM_YES	1	X'00000001'

MQDT_ * (cílové typy)

Tabulka 151. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDT_APPL	1	X'00000001'
MQDT_BROKER	2	X'00000002'

MQDXP_ * (struktura výstupního parametru převodu)

Tabulka 152. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQDXP_STRUCTION_ID	"DXP¬"
POLE MQDXP_STRUC_ID_ARRAY	'D','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 153. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDXP_VERSION_1	1	X'00000001'
MQDXP_VERSION_2	2	X'00000002'
AKTUÁLNÍ_VERZE MQDXP_CURRENT_VERSION	2	X'00000002'

MQEC_ * (signálové hodnoty)

Tabulka 154. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEC_MSG_ARRIVED	2	X'00000002'
MQEC_WAIT_INTERVAL_EXPIRED	3	X'00000003'
ČEKÁNÍ MQEC_WAIT_CANCELED	4	X'00000004'
UVÁDĚNÍ MQEC_Q_MGR_QUIESCING	5	X'00000005'
FUNKCE MQEC_CONNECTION_QUIESCING	6	X'00000006'

MQEI_ * (Vypršení platnosti)

Tabulka 155. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEI_UNLIMITED	-1	X'FFFFFFFF'

MQENC_ * (Kódování)

MQENC_ * (Kódování)

Tabulka 156. Hodnoty konstant podle platformy			
Název	**Platforma**	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_NATIVE	IBM i	273	X'00000111 '.
	Linux	546	X'00000222 '.
	Linux na SPARC	273	X'00000111 '.
	Linux na x86	546	X'00000222 '.
	AIX and Linux	273	X'00000111 '.
	Windows	546	X'00000222 '.
	Micro fokus COBOL na Windows	17	X'00000011 '
	z/OS	785	X'00000311 '

MQENC_ * (zakódování Masky)

Tabulka 157. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_INTEGER_MASK	15	X'0000000F'
MQENC_DECIMAL_MASK	240	X'000000F0'
MQENC_FLOAT_MASK.	3840	X'00000F00'
MAQ_REZERVOVANÁ_MASKA	-4096	X'FFFFF000'

MQENC_ * (kódování pro binární celé číslo)

Tabulka 158. 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 desítková čísla komprimovaného desetinného čísla)

Tabulka 159. 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 čísel s pohyblivou řádovou čárkou)

Tabulka 160. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_FLOAT_UNDEFINED	0	X'00000000'
MQENC_FLOAT_IEEEE_NORMAL	256	X'00000100'
MQENC_FLOAT_IEEE_OBRÁCENÝ	512	X'00000200'
MQENC_FLOAT_S390	768	X'00000300'
MQENC_FLOAT_TNS,	1024	X'00000400'

MQEPH_ * (Struktura záhlaví vloženého příkazu a příznaky)

Struktura záhlaví vestavěného příkazového formátu

Tabulka 161. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQEPH_	"EPH¬"
POLE MQEPH_STRUC_ID_ARRAY	'E','P','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 162. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SPRAVA_STRUKTUROVANÉ_STRUKTUROVANÉHO_SYSTÉMU	68	X'00000044'
MQEPH_VERSION_1	1	X'00000001'
VERZE AKTUÁLNÍ_VERZE	1	X'00000001'

Parametry záhlaví vloženého příkazového formátu

Tabulka 163. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEPH_NONE	0	X'00000000'

Tabulka 163. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VLOŽKA MQEPH_CCSID_EMBEDDED	1	X'00000001'

MQET_ * (Typy znaků změny formátu příkazu)

Tabulka 164. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQET_MQSC	1	X'00000001'

MQEVO_ * (Standardní formát událostí událostí)

Tabulka 165. 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'
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNAL	5	X'00000005'
MQEVO_MQSUB	6	X'00000006'
ZPRÁVA MQEVO_CTLMSG	7	X'00000007'
MQEVO_REST	8	X'00000008'

MQEVR_ * (záznam událostí ve formátu příkazu)

Tabulka 166. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEV_DISABLED	0	X'00000000'
POVOLENÝ MQEVR_	1	X'00000001'
VÝJIMKA MQEVR_EXCEPTION	2	X'00000002'
MQEVR_NO_DISPLAY	3	X'00000003'

MQEXPI_ * (Interval skenování vypršení platnosti)

Tabulka 167. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEXPI_OFF	0	X'00000000'

MQFB_ * (hodnoty zpětné vazby)

Tabulka 168. 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'

Tabulka 168. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_COMPLETED	262	X'00000106'
MQFB_CHANNEL_FAIL_RETRY	263	X'00000107'
MQFB_CHANNEL_FAIL	264	X'00000108'
MQFB_APPL_CANNOT_BE_STARTED	265	X'00000109'
CHYBA MQFB_TM_ERROR	266	X'0000010A'
CHYBA MQFB_APPL_TYPE_ERROR	267	X'0000010B'
MQFB_STOPPED_BY_MSG_EXIT	268	X'0000010C'
AKTIVITA MQFB_ACTIVITY	269	X'0000010D'
CHYBA MQFB_XMIT_Q_MSG_ERROR	271	X'0000010F'
MQFB_PAN	275	X'00000113'
MQFB_NAN	276	X'00000114'
MQFB_STOPPED_BY_CHAD_EXIT	277	X'00000115'
MQFB_STOPPED_BY_PUBSUB_EXIT	279	X'00000117'
MQFB_NOT_A_REPOSITORY_MSG	280	X'00000118'
MQFB_BIND_OPEN_CLURCVR_DEL	281	X'00000119'
MQFB_MAX_AKTIVIT	282	X'0000011A'
MQFB_NOT_FORWARDED	283	X'0000011B'
MQFB_NOT_DELIVERED	284	X'0000011C'
MQFB_UNSUPPORTED_FORWARDING	285	X'0000011D'
MQFB_UNSUPPORTED_DELIVERY	286	X'0000011E'
MQFB_DATA_LENGTH_ZERO	291	X'00000123'
MQFB_DATA_LENGTH_NEGATIVE	292	X'00000124'
MQFB_DATA_LENGTH_TOO_BIG	293	X'00000125'
PŘETEČENÍ MQFFB_BUFFER_OVERFLOW	294	X'00000126'
MQFB_LENGTH_OFF_BY_ONE	295	X'00000127'
CHYBA MQFB_IIH_ERROR	296	X'00000128'
MQFB_NOT_AUTHORIZED_FOR_IMS	298	X'0000012A'
CHYBA 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'
SELHÁNÍ MQFB_CICS_BRIDGE_FAILURE	403	X'00000193'
CHYBA MQFB_CICS_CORREL_ID_ERROR	404	X'00000194'
CHYBA MQFB_CICS_CCSID_ERROR	405	X'00000195'
CHYBA MQFB_CICS_ENCODING_ERROR	406	X'00000196'
MQFB_CICS_CIH_ERROR	407	X'00000197'

Tabulka 168. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
CHYBA MQFB_CICS_UOW_ERROR	408	X'00000198'
CHYBA MQFB_CICS_COMMAGA_ERROR	409	X'00000199'
MQFB_CICS_APPL_NOT_STARTED	410	X'0000019A'
MQFB_CICS_APPL_ABENDED	411	X'0000019B'
CHYBA MQFB_CICS_DLQ_ERROR	412	X'0000019C'
MQFB_CICS_UOW_BACKED_OUT	413	X'0000019D'
POŽADAVEK MQFB_PUBLICATIONS_ON_REQUEST	501	X'000001F5'
MQFB_SUBSCRIBER_IS_PUBLISHER	502	X'000001F6'
NESROVNALOST MQFB_MSG_SCOPE_MATCH	503	X'000001F7'
MQFB_SELECTOR_MISMATCH	504	X'000001F8'
MQFB_IMS_NACK_1A_REASON_FIRST	600	X'00000258'
MQFB_IMS_NACK_1A_REASON_LAST	855	X'00000357'
MQFB_SYSTEM_LAST	65535	X'0000FFFF'
MQFB_APPL_FIRST	65536	X'00010000'
MQFB_APPL_LAST	999999999	X'3B9AC9FF'

MQFC_ * (Volby vynucení formátu příkazu)

Tabulka 169. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFC_YES	1	X'00000001'
MQFC_NO	0	X'00000000'

MQFMT_ * (Formáty)

Tabulka 170. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQFMT_NONE	"¬¬¬¬¬¬¬¬"
MQFMT_ADMIN	"MQADMIN¬"
MQFMT_CHANNEL_COMPLETED	"MQCHCOM¬"
MQFMT_CICS	"MQCICS¬¬"
MQFMT_COMMAND_1	"MQCMD1¬¬"
MQFMT_COMMAND_2	"MQCMD2¬¬"
HLAVIČKA MQFMT_DEAD_LETTER_HEADER	"MQDEAD¬¬"
ZÁHLAVÍ MQFMT_DICT_HEADER	"MQHDIST¬"
MQFMT_EMBEDDED_PCF	"MQHEPCF¬"
UDÁLOST MQFMT_EVENT	"MQEVENT¬"
MQFMT_IMS	"MQIMS¬¬¬"
MQFMT_IMS_VAR_STRING	"MQIMSVS¬"
ROZŠÍŘENÍ MQFMT_MD_EXTENSION	"MQHMDE¬¬"
MQFMT_PCF	"MQPCF¬¬¬"

Tabulka 170. Konstantní názvy a hodnoty (pokračování)	
Název	**Hodnota**
MQFMT_REF_MSG_HEADER	"MQHREF¬¬"
ZÁHLAVÍ MQFMT_RF_HEADER	"MQHRF¬¬¬"
MQFMT_RF_HEADER_1	"MQHRF¬¬¬"
MQFMT_RF_HEADER_2	"MQHRF2¬¬"
ŘETĚZEC MQFMT_STRING	"MQSTR¬¬¬"
SPOUŠTĚČ MQFMT_TRIGGER	"MQTRIG¬¬"
MQFMT_WORK_INFO_HEADER	"MQHWIH¬¬"
ZÁHLAVÍ MQFMT_XMIT_Q_HEADER	"MQXMIT¬¬"
MQFMT_NO_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'
MQFMT_ADMIN_ARRAY	'M','Q','A','D','M','I','N','¬'
MQFMT_CHANNEL_COMPLETED_ARRAY	'M','Q','C','H','C','O','M','¬'
MQFMT_CICS_ARRAY	'M','Q','C','I','C','S','¬','¬'
MQFMT_COMMAND_1_ARRAY	'M','Q','C','M','D','1','¬','¬'
MQFMT_COMMAND_2_ARRAY	'M','Q','C','M','D','2','¬','¬'
MQFMT_DEAD_LETTER_HEADER_ARRAY	'M','Q','D','E','A','D','¬','¬'
POLE MQFMT_DIFT_HEADER_ARRAY	'M','Q','H','D','I','S','T','¬'
MQFMT_EMBEDDED_PCF_ARRAY	'M','Q','H','E','P','C','F','¬'
POLE 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','¬','¬'
POLE MQFMT_RF_HEADER_ARRAY	'M','Q','H','R','F','¬','¬','¬'
MQFMT_RF_HEADER_1_ARRAY	'M','Q','H','R','F','¬','¬','¬'
MQFMT_RF_HEADER_2_ARRAY	'M','Q','H','R','F','2','¬','¬'
MQFMT_STRING_ARRAY	'M','Q','S','T','R','¬','¬','¬'
POLE MQFMT_TRIGGER_ARRAY	'M','Q','T','R','I','G','¬','¬'
POLE MQFMT_WORK_INFO_HEADER_ARRAY	'M','Q','H','W','I','H','¬','¬'
POLE MQFMT_XMIT_Q_HEADER_ARRAY	'M','Q','X','M','I','T','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQFUN_ * (Typy aplikačních funkcí)

Tabulka 171. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEZNÁMÝ TYP MQFUN_TYPE_UNKNOWN	0	X'00000000'
MQFUN_TYPY_JVM	1	X'00000001'
MQFUN_TYPE_PROGRAM	2	X'00000002'
PROCEDURA MQFUN_TYPE_PROCEDURE	3	X'00000003'

Tabulka 171. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFUN_TYPE_USERDEF.	4	X'00000004'
PŘÍKAZ MQFUN_TYPE_COMMAND	5	X'00000005'

MQGA_ * (Selektory atributu skupiny)

Tabulka 172. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQGA_FIRST	8001	X'00001F41'
MQGA_LAST	9000	X'00002328'

MQGACF_ * (Typy parametrů skupiny parametrů příkazu)

Tabulka 173. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQGAF_FIRST	8001	X'00001F41'
MQGACF_COMMAND_CONTEXT	8001	X'00001F41'
MQGACF_COMMAND_DATA	8002	X'00001F42'
MQGACF_TRACE_TRAUTE	8003	X'00001F43'
OPERACE MQGAF_OPERATION	8004	X'00001F44'
AKTIVITA MQGACF_ACTIVITY	8005	X'00001F45'
MQGACF_EMBEDDED_MQMD.	8006	X'00001F46'
ZPRÁVA MQGACF_MESSAGE	8007	X'00001F47'
MQGACF_MQMD	8008	X'00001F48'
POJMENOVÁNÍ MQGACF_VALUE_NAME_VALUE	8009	X'00001F49'
MQGAC_Q_ACCOUNTING_DATA	8010	X'00001F4A'
MQGACF_Q_STATISTICS_DATA	8011	X'00001F4B'
DATA MQGACF_CHL_STATISTICS_DATA	8012	X'00001F4C'
MQGAC_LAST_USED	8012	X'00001F4C'

MQGI_ * (Identifikátor skupiny)

Tabulka 174. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQGI_NONE	X'00...00' (24 nul)
MQGI_NON_ARRAY	'\0','\0',... (24 nul)

MQGMO_ * (Získat volby a strukturu zprávy)

Získat strukturu voleb zprávy

Tabulka 175. Konstrukce konstant	
Název	**Struktura**
MQGMO_STRUCTION_ID	"GMO¬"
MQGMO_STRUC_ID_POLE	'G','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 176. 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'
AKTUÁLNÍ_VERZE MQGMO_VERSION	4	X'00000004'

Volby získání zprávy

Tabulka 177. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQGMO_WAIT	1	X'00000001'
MQGMO_NO_WAIT	0	X'00000000'
SIGNÁL MQGMO_SET_DATA	8	X'00000008'
FUNKCE 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'
NEJPRVE MQGMO_BROWSE_FIRST	16	X'00000010'
PŘÍŠTĚ MQGMO_BROWSE_NEXT	32	X'00000020'
MQGMO_BROWSE_MSG_UNDER_CURSOR	2048	X'00000800'
POPISOVAČ MQGMO_BROWSE_HANDLE	17825808	X'01100010'
MQGMO_BROWSE_CO_OP	18874384	X'01200010'
MQGMO_MSG_UNDER_CURSOR	256	X'00000100'
MQGMOVÝ_ZÁMEK	512	X'00000200'
MQGMO_ODEMKNOUT	1024	X'00000400'
SOUBOR MQGMO_ACCEPT_TRUNCATED_MSG	64	X'00000040'
MQGMO_CONVERT	16384	X'00004000'
MQGMO_LOGICAL_ORDER	32768	X'00008000'
ZPRÁVA MQGMO_COMPLETE_MSG	65536	X'00010000'
MQGMO_ALL_MSGS_AVAILABLE	131072	X'00020000'
DOSTUPNÉ MQGMO_ALL_SEGMENTS_AVAILABLE	262144	X'00040000'
POPISOVAČ MQGMO_MARK_ BROWSE_HANDLE	1048576	X'00100000'
MQGMO_MARKER_BROWSE_CO_OP	2097152	X'00200000'
MQGMO_UNMARK_ BROWSE_CO_OP	4194304	X'00400000'
POPISOVAČ 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'

Tabulka 177. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 178. Konstantní názvy 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_ * (Obsluha selektorů)

Tabulka 179. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQHA_FIRST	4001	X'00000FA1'
MQHA_BAG_HANDLE.	4001	X'00000FA1'
MQHA_LAST_USED	4001	X'00000FA1'
MQHA_LAST	6000	X'00001770'

MQHB_ * (Lak Handles)

Tabulka 180. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQHT_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_NONE	-2	X'FFFFFFFE'

MQHC_ * (Manipulátory připojení)

Tabulka 181. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQC_DEF_HCONN	0	X'00000000'
MQC_UNUSABLE_HCONN	-1	X'FFFFFFFF'
PŘIPOJENÍ MQC_UNASSOCIATED_HCONN	-3	X'FFFFFFFD'

MQHMC_ * (Popisovač zprávy)

Tabulka 182. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MAHL_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQM_NONE	0	X'00000000'

MQHO_ * (Popisovač objektu)

Tabulka 183. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQHO_UNUSABLE_HOBJ	-1	X'FFFFFFFF'
MQHO_NONE	0	X'00000000'

MQHSTATE_ * (obslužné rutiny formátu příkazu)

Tabulka 184. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQLSTATE_INACTIVE	0	X'00000000'
MQHSTATE_ACTIVE	1	X'00000001'

MQIA_ * (Selektory celočíselných atributů)

Tabulka 185. 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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_MAX_Q_DEPTH	15	X'0000000F'
MQIA_MAX_Q_TRIGGERS	90	X'0000005A'
MQIA_MAX_RECOVERY_TASKS	171	X'000000AB'
MQIA_MAX_UNCOMMITTED_MSGS	33	X'00000021'
MQIA_MCAST_BRIDGE	233	X'000000E9'
MQIA_MONITOR_INTERVAL	81	X'00000051'
MQIA_MONITORING_AUTO_CLUSSDR	124	X'0000007C'
MQIA_MONITORING_CHANNEL	122	X'0000007A'
MQIA_MONITORING_Q	123	X'0000007B'
MQIA_MSG_DELIVERY_SEQUENCE	16	X'00000010'
MQIA_MSG_DEQ_COUNT	38	X'00000026'
MQIA_MSG_ENQ_COUNT	37	X'00000025'
MQIA_NAME_COUNT	19	X'00000013'
MQIA_NAMELIST_TYPE	72	X'00000048'
MQIA_NPM_CLASS	78	X'0000004E'
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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_SHAREABILITY	23	X'00000017'
MQIA_SHARED_Q_Q_MGR_NAME	77	X'0000004D'
MQIA_SSL_EVENT	75	X'0000004B'
MQIA_SSL_FIPS_REQUIRED	92	X'0000005C'
MQIA_SSL_RESET_COUNT	76	X'0000004C'
MQIA_SSL_TASKS	69	X'00000045'
MQIA_START_STOP_EVENT	52	X'00000034'
MQIA_STATISTICS_CHANNEL	129	X'00000081'
MQIA_STATISTICS_AUTO_CLUSSDR	130	X'00000082'
MQIA_STATISTICS_INTERVAL	131	X'00000083'
MQIA_STATISTICS_MQI	127	X'0000007F'
MQIA_STATISTICS_Q	128	X'00000080'
MQIA_SUB_COUNT	204	X'000000CC'
MQIA_SUB_SCOPE	218	X'000000DA'
MQIA_SYNCPOINT	30	X'0000001E'
MQIA_TCP_CHANNELS	114	X'00000072'
MQIA_TCP_KEEP_ALIVE	115	X'00000073'
MQIA_TCP_STACK_TYPE	116	X'00000074'
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_ * (typy parametrů s celočíselnými parametry)

Tabulka 186. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_FIRST	1001	X'000003E9'
MQIACF_Q_MGR_ATTRS	1001	X'000003E9'
MQIAKF_Q_ATTRS	1002	X'000003EA'
MQIACF_PROCESS_ATTRS	1003	X'000003EB'
MQIACF_NAMELIST_ATTRS	1004	X'000003EC'
MQIACF_FORCE	1005	X'000003ED'
MQIAKF_REPLACE	1006	X'000003EE'
MQIACF_PURGE	1007	X'000003EF'
MQIAKF_TIŠENÍ	1008	X'000003F0'
REŽIM MQIACF_MODE	1008	X'000003F0'
MQIACF_ALL	1009	X'000003F1'
TYP_APL MQIACF_TYP_UDÁLOSTI	1010	X'000003F2'
MQIACF_EVENT_ORIGIN	1011	X'000003F3'
MQIACF_PARAMETER_ID	1012	X'000003F4'
MQIACF_ERROR_ID	1013	X'000003F5'
MQIACF_ERROR_IDENTIFIER.	1013	X'000003F5'
MQIACF_SELEC	1014	X'000003F6'
MQIACF_CHANNEL_ATTRS	1015	X'000003F7'
MQIACF_OBJECT_TYPE	1016	X'000003F8'
TYP MQIACF_ESCAPE_TYPE	1017	X'000003F9'
MQIACF_ERROR_OFFSET	1018	X'000003FA'
MQIACF_AUTH_INFO_ATTRS	1019	X'000003FB'
KVALIFIKÁTOR MQIACF_REASON_QUALI	1020	X'000003FC'
PŘÍKAZ MQIACF_COMMAND	1021	X'000003FD'
VOLBY MQIACF_OPEN_OPTIONS	1022	X'000003FE'
TYP OTEVŘENÍ MQIACF_OPENTYPE	1023	X'000003FF'
ID_PROCESU_MIME	1024	X'00000400'
ID_PODPROCESU MQIACF_THREAD_ID	1025	X'00000401'
MQIACF_Q_STATUS_ATTRS	1026	X'00000402'
MQIACF_UNCOMMITTED_MSGS	1027	X'00000403'
MQIACF_HANDLE_STATE,	1028	X'00000404'
MQIACF_AUX_ERROR_DATA_INT_1	1070	X'0000042E'
MQIACF_AUX_ERROR_DATA_INT_2	1071	X'0000042F'
MQIACF_CONV_REASON_CODE	1072	X'00000430'
MQIACF_BRIDGE_TYPE	1073	X'00000431'
MQIACF_DOTAZ	1074	X'00000432'
MQIACF_WAIT_INTERVAL	1075	X'00000433'
VOLBY MQIAKF_	1076	X'00000434'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_BROKER_OPTIONS	1077	X'00000435'
MQIACF_TYP_OBNOVY	1078	X'00000436'
ČÍSLO MQIAKF_SEQUENCE_NUMBER	1079	X'00000437'
MQIACF_INTEGER_DATA	1080	X'00000438'
VOLBY MQIACF_REGISTRATION_OPTIONS	1081	X'00000439'
MQIACF_PUBLICTION_OPTIONS	1082	X'0000043A'
INFORMACE O KLASTRU MQIACF_CLUSTER	1083	X'0000043B'
TYP_DEFINICE_MQIACF_Q_MGR_DEFINITION_TYPE	1084	X'0000043C'
TYP MQIACF_Q_MGR_TYPE	1085	X'0000043D'
MQIAKF_AKCE	1086	X'0000043E'
MQIACF_SUSPEND	1087	X'0000043F'
MQIACF_BROKER_COUNT	1088	X'00000440'
POČET MQIACF_APPL_COUNT	1089	X'00000441'
MQIACF_ANONYMOUS_COUNT	1090	X'00000442'
MQIACF_REG_REG_OPTIONS	1091	X'00000443'
VOLBY 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_DOTAZ	1101	X'0000044D'
SPRÁVA MQIACF_OPEN_BROWSE	1102	X'0000044E'
MQIACF_Q_STATUS_TYPE	1103	X'0000044F'
MQIACF_Q_HANDLE	1104	X'00000450'
STAV MQIAKF_Q_STATUS	1105	X'00000451'
MQIACF_SECURITY_TYPE	1106	X'00000452'
MQIACF_CONNECTION_ATTRS	1107	X'00000453'
VOLBY 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'
VOLÁNÍ MQIACF_AUTH_ADD_AUTHS	1116	X'0000045C'
MQIACF_AUTH_REMOVE_AUTHS	1117	X'0000045D'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_ENTITY_TYPE	1118	X'0000045E'
MQIACF_COMMAND_INFO	1120	X'00000460'
POČET MQIACF_CMDSCOPY_Q_MGR_COUNT	1121	X'00000461'
SYSTÉM MQIACF_Q_MGR_SYSTEM	1122	X'00000462'
MQIACF_Q_MGR_EVENT, UDÁLOST	1123	X'00000463'
MQIACF_Q_MGR_DQM	1124	X'00000464'
KLASTR 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_STRU_STATUS	1130	X'0000046A'
MQIACF_UOW_TYPE	1132	X'0000046C'
MQIACF_CF_STRU_ATTRS	1133	X'0000046D'
MQIACF_EXCLUDE_INTERVAL	1134	X'0000046E'
MQIACF_CF_STATUS_TYPE	1135	X'0000046F'
SOUHRN STAVU MQIAKF_CF_STATUS_SUMMARY	1136	X'00000470'
MQIACF_CF_STATUS_CONNECT	1137	X'00000471'
MQIACF_CF_STATUS_BACKUP.	1138	X'00000472'
MQIACF_CF_STRU_TYPE	1139	X'00000473'
MQIACF_CF_STRU_SIZE_MAX	1140	X'00000474'
MQIACF_CF_STRU_SIZE_USED	1141	X'00000475'
MQIACF_CF_STRUC_ENTRIES_MAX	1142	X'00000476'
POUŽÍVÁ SE MQIACF_CF_STRUC_ENTRIES_USED	1143	X'00000477'
MQIACF_CF_STRUC_BACKUP_SIZE	1144	X'00000478'
MQIACF_MOVETYPE	1145	X'00000479'
MQIACF_MOVE_TYPE_MOVE	1146	X'0000047A'
MQIACF_MOVE_TYPE_ADD	1147	X'0000047B'
HODNOTA MQIACF_Q_MGR_NUMBER	1148	X'0000047C'
STAV 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 MQIAKF_SECURITY_SETTING	1155	X'00000483'
TŘÍDA MQIACF_STORAGE_CLASS_ATTRS	1156	X'00000484'
TYP MQIACF_USAGE_TYPE	1157	X'00000485'
MQIACF_BUFFER_POOL_ID	1158	X'00000486'
MQIACF_USAGE_TOTAL_PAGES	1159	X'00000487'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
STRÁNKY MQIACF_USAGE_UNUSED_PAGES	1160	X'00000488'
MQIAKF_USAGE_PERSIST_PAGES	1161	X'00000489'
MQIAKF_USAGE_NONPERSIST_PAGES	1162	X'0000048A'
OBLASTI MQIACF_USAGE_RESTART_EX	1163	X'0000048B'
MQIACF_USAGE_EXPAND_COUNT	1164	X'0000048C'
STAV MQIACF_PAGESET_STATUS	1165	X'0000048D'
MQIAKF_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'
OBLAST MQIAKF_USAGE_BUFFER_POOL	1170	X'00000492'
MQIACF_MOVE_COUNT	1171	X'00000493'
MQIACF_EXPIRY_Q_COUNT	1172	X'00000494'
OBJEKTY MQIACF_CONFIGURATION_	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_PÁSKS	1178	X'0000049A'
MQIACF_SYSP_IN_BUFFER_SIZE	1179	X'0000049B'
VELIKOST VYROVNÁVACÍ PAMĚTI MQIACF_SYSP_OUT_BUFFER_SIZE	1180	X'0000049C'
POČET VYROVNÁVACÍCH PAMĚTÍ MQIACF_SYSP_OUT_BUFFER_COUNT	1181	X'0000049D'
MQIACF_SYSP_ARCHIVE	1182	X'0000049E'
MQIAKF_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_PROED	1187	X'000004A3'
MQIACF_SYSP_MAX_CONNS_BACK	1188	X'000004A4'
MQIACF_SYSP_EXIT_INTERVAL	1189	X'000004A5'
ÚLOHY 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'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAKF_SYSP_SMF_STATS	1198	X'000004AE'
MQIACF_SYSP_SMF_INTERVAL	1199	X'000004AF'
TŘÍDA MQIACF_SYSP_TRACE_CLASS	1200	X'000004B0'
VELIKOST 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'
KATALOG 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'
MQIACF_SYSP_QUIESCE_INTERVAL	1212	X'000004BC'
ČASOVÉ RAZÍTKO MQIAKF_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'
POZASTAVENÍ MQIACF_SYSP_LOG_SUSPEND	1218	X'000004C2'
STAV 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'
OBJEKTY MQIACF_SERVICE_ATTRS	1224	X'000004C8'
MQIACF_SERVICE_STATUS_ATTRS	1225	X'000004C9'
UKAZATEL 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'
CQIACF_CONNECTION_COUNT	1230	X'000004CE'
ZAŘÍZENÍ MQIACF_Q_MGR_FACILITY	1231	X'000004CF'
STAV MQIACF_CHINIT_STATUS	1232	X'000004D0'
STAV MQIACF_CMD_SERVER_STATUS	1233	X'000004D1'
MQIACF_ROUTE_DETAIL	1234	X'000004D2'
AKTIVITY MQIACF_RECORDED_ACTIVITIES	1235	X'000004D3'
MQIACF_MAX_AKTIVIT	1236	X'000004D4'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_DISTCONTINUITY_COUNT	1237	X'000004D5'
MQIACF_ROUTE_ACCULATION	1238	X'000004D6'
MQIACF_ROUTE_DELIVERY	1239	X'000004D7'
TYP OPERACE MQIACF_OPERATION_TYPE	1240	X'000004D8'
MQIACF_BACKOUT_COUNT.	1241	X'000004D9'
MQIACF_COMP_CODE	1242	X'000004DA'
KÓDOVÁNÍ MQIACF_	1243	X'000004DB'
MQIACF_EXPIRACE	1244	X'000004DC'
ZPĚTNÁ VAZBA MQIAKF_	1245	X'000004DD'
PŘÍZNAKY 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'
MQIACF_PERSISTENCE	1252	X'000004E4'
MQIAKF_PRIORITY	1253	X'000004E5'
MQIACF_REASON_CODE	1254	X'000004E6'
ZPRÁVA MQIACF_REPORT	1255	X'000004E7'
MQIACF_VERSION	1256	X'000004E8'
MQIACF_UNRECORDED_ACTIVITIES	1257	X'000004E9'
MQIAKF_MONITORING	1258	X'000004EA'
MQIACF_ROUTE_FORWARDING	1259	X'000004EB'
STAV SLUŽBY MQIACF_SERVICE_	1260	X'000004EC'
TYPY MQIACF_Q_TYPES	1261	X'000004ED'
PODPORA MQIACF_USER_ID_SUPPORT	1262	X'000004EE'
VERZE MQIAKF_INTERFACE_VERSION	1263	X'000004EF'
OBJEKTY MQIACF_AUTH_SERVICE_ATTRS	1264	X'000004F0'
MQIACF_USAGE_EXPAND_TYPE	1265	X'000004F1'
MEZIPAMĚŤ 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'
VLASTNOSTI MQIAKF_PUBSUB_PROPERTIES	1271	X'000004F7'
TŘÍDA MQIACF_DESTINATION_CLASS	1273	X'000004F9'
MQIACF_DERABLE_SUBSCRIPTION	1274	X'000004FA'
MQIACF_SUBSCRIPTION_SCOPE	1275	X'000004FB'
MQIACF_VARIABLE_USER_ID	1277	X'000004FD'
POUZE PRO MQIACF_REQUEST_ONLY	1280	X'00000500'
MQIACF_PUB_PRIORITY	1283	X'00000503'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_SUB_ATTRS	1287	X'00000507'
SCHÉMA MQIACF_WILDCARD_SCHEMA	1288	X'00000508'
TYP MQIACF_SUB_TYPE	1289	X'00000509'
POČET ZPRÁV 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'
STAV MQIAKF_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'
TYP STAVU MQIACF_TOPIC_STATUS_TYPE	1302	X'00000516'
MQIACF_SUB_OPTIONS	1303	X'00000517'
POČET PUBLIKOVÁNÍ MQIAKF_PUBLISH_	1304	X'00000518'
MQIACF_CLEAR_TYPE	1305	X'00000519'
MQIAKF_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'
STAV MQIACF_PUBSUB_STATUS	1311	X'0000051F'
TYP STAVU MQIACF_PS_STATUS_TYPE	1314	X'00000522'
MQIACF_PUBSUB_STATUS_ATTRS	1318	X'00000526'
TYP_SELEKTORU MQIACF_SELECTTYPE	1321	X'00000529'
MQIACF_MCAST_RELACE_INDIKÁTOR	1351	X'00000547'
MQIACF_CHLAUTH_TYPE	1352	X'00000548'
TYP MQXR_DIAGNOSTICS_TYPE	1354	X'0000054A'
MQIACF_CHLAUTH_ATTRS	1355	X'0000054B'
ID OPERACE MQIACF_OPERATION_ID	1356	X'0000054C'
MQIACF_API_CALLER_TYPE	1357	X'0000054D'
MQIACF_API_ENVIRONMENT	1358	X'0000054E'
MQIACF_TRACE_DETAIL	1359	X'0000054F'
MQIACF_HOBJ	1360	X'00000550'
MQIACF_CALL_TYPE	1361	X'00000551'
OPERACE MQIACF_MQCB_OPERATION	1362	X'00000552'
MQIACF_MQCB_TYPE	1363	X'00000553'
VOLBY MQIACF_MQCB_OPTIONS	1364	X'00000554'
VOLBY MQIACF_CLOSE_OPTIONS	1365	X'00000555'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
OPERACE MQIACF_CTL_CTL_	1366	X'00000556'
MQIAKF_GET_OPTIONS	1367	X'00000557'
MQIACF_RECS_RECT	1368	X'00000558'
MQIACF_KNOWN_DEST_COUNT	1369	X'00000559'
MQIACF_UNKNOWN_DEST_COUNT	1370	X'0000055A'
MQIACF_INVALID_DEST_COUNT	1371	X'0000055B'
TYP ATRIBUTU MQIACF_RESOLVED_NAME	1372	X'0000055C'
VOLBY MQIACF_PUT_OPTIONS	1373	X'0000055D'
MQIACF_BUFFER_LENGTH	1374	X'0000055E'
DÉLKA MQIACF_TRACE_DATA_LENGTH	1375	X'0000055F'
MQIACF_SMDS_EXPANDST	1376	X'00000560'
MQIACF_STRUST_LENGTH	1377	X'00000561'
POČET POLOŽEK MQIACF_ITEM_COUNT	1378	X'00000562'
MQIACF_EXPIRY_TIME,	1379	X'00000563'
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'
PARAMETRY MQIAKF_XA_FLAGS	1385	X'00000569'
MQIACF_XA_PROTECTCODE	1386	X'0000056A'
MQIACF_XA_HANDLE	1387	X'0000056B'
MQIACF_XA_RETVAL	1388	X'0000056C'
TYP STAVU MQIACF_STATUS	1389	X'0000056D'
MQIACF_XA_COUNT	1390	X'0000056E'
MQIACF_SELECTOR_COUNT	1391	X'0000056F'
PŘEDDEFINOVANÉ SELEKTORY	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'
MQIAKF_AMQP_ATTRS	1401	X'00000579'
MQIACF_EXPORT_TYPE	1402	X'0000057A'
MQIACF_EXPORT_ATTRS	1403	X'0000057B'
OBJEKTY MQIACF_SYSTEM_OBJECTS	1404	X'0000057C'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACF_CONNECTION_SWAP	1405	X'0000057D'
TYP MQIACF_AMQP_DIAGNOSTICS_TYPE	1406	X'0000057E'
VELIKOST FONDU VYROVNÁVACÍCH PAMĚTÍ MQIACF_BUFFER_LOCATION	1408	X'00000580'
STAV MQIACF_LDAP_CONNECTION_STATUS	1409	X'00000581'
MQIACF_SYSP_MAX_AC_POOL	1410	X'00000582'
MQIACF_PAGECLAS,	1411	X'00000583'
MQIACF_AUTH_REC_TYPE	1412	X'00000584'
MQIACF_SYSP_MAX_CONCTION_OFFLOADS	1413	X'00000585'
MQIAKF_SYSP_ZHYPERWRITE	1414	X'00000586'
PROTOKOL MQIACF_Q_MGR_STATUS_LOG	1415	X'00000587'
VELIKOST PROTOKOLU MQIACF_ARCHIVE_LOG_SIZE	1416	X'00000588'
MQIACF_MEDIA_LOG_SIZE	1417	X'00000589'
VELIKOST PROTOKOLU MQIACF_RESTART_LOG_SIZE	1418	X'0000058A'
VELIKOST PROTOKOLU MQIACF_REUSABLE_LOG_SIZE	1419	X'0000058B'
MQIACF_LOG_IN_USE	1420	X'0000058C'
MQIACF_LOG_VYUŽITÍ	1421	X'0000058D'
V 9.2.0 V 9.2.0 MQIACF_IGNORE_STATE	1423	X'0000058F'
POČET MQIACF_MOVABLE_APPL_COUNT	1424	X'00000590'
MQIACF_APPL_INFO_ATTRS	1425	X'00000591'
MQIACF_APPL_MOVE_	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'
POČET MQIACF_APPL_IMMOVABLE_COUNT	1432	X'00000598'
MQIACF_BALAN	1433	X'00000599'
MQIACF_BALSTATE	1434	X'0000059A'
MQIACF_APPL_IMMOVABLE_REASON	1435	X'0000059B'
MQIACF_DS_ENCRYPTED	1436	X'0000059C'
VELIKOST SOUBORU MQIACF_CUR_Q_FILE_SIZE	1437	X'0000059D'
VELIKOST SOUBORU 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_SMP_ACCT_ACCT_SECS	1444	X'000005A4'

Tabulka 186. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
LTS LTS MQIAKF_LAST_USED	1441	X'000005A1'
V 9.2.4 V 9.2.4 MQIAKF_LAST_USED	1444	X'000005A4'

MQIACH_ * (Formáty celočíselných kanálů formátu příkazu)

Tabulka 187. 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'
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'

Tabulka 187. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACH_CURRENT_SEQUENCE_NUMBER	1532	X'000005FC'
MQIACH_SSL_RETURN_CODE	1533	X'000005FD'
MQIACH_MSGS	1534	X'000005FE'
MQIACH_BYTES_SENT	1535	X'000005FF'
MQIACH_BYTES_RCVD	1536	X'00000600'
MQIACH_BYTES_RECEIVED	1536	X'00000600'
MQIACH_BATCHES	1537	X'00000601'
MQIACH_BUFFERS_SENT	1538	X'00000602'
MQIACH_BUFFERS_RCVD	1539	X'00000603'
MQIACH_BUFFERS_RECEIVED	1539	X'00000603'
MQIACH_LONG_RETRIES_LEFT	1540	X'00000604'
MQIACH_SHORT_RETRIES_LEFT	1541	X'00000605'
MQIACH_MCA_STATUS	1542	X'00000606'
MQIACH_STOP_REQUESTED	1543	X'00000607'
MQIACH_MR_COUNT	1544	X'00000608'
MQIACH_MR_INTERVAL	1545	X'00000609'
MQIACH_NPM_SPEED	1562	X'0000061A'
MQIACH_HB_INTERVAL	1563	X'0000061B'
MQIACH_BATCH_INTERVAL	1564	X'0000061C'
MQIACH_NETWORK_PRIORITY	1565	X'0000061D'
MQIACH_KEEP_ALIVE_INTERVAL	1566	X'0000061E'
MQIACH_BATCH_HB	1567	X'0000061F'
MQIACH_SSL_CLIENT_AUTH	1568	X'00000620'
MQIACH_ALLOC_RETRY	1570	X'00000622'
MQIACH_ALLOC_FAST_TIMER	1571	X'00000623'
MQIACH_ALLOC_SLOW_TIMER	1572	X'00000624'
MQIACH_DISC_RETRY	1573	X'00000625'
MQIACH_PORT_NUMBER	1574	X'00000626'
MQIACH_HDR_COMPRESSION	1575	X'00000627'
MQIACH_MSG_COMPRESSION	1576	X'00000628'
MQIACH_CLWL_CHANNEL_RANK	1577	X'00000629'
MQIACH_CLWL_CHANNEL_PRIORITY	1578	X'0000062A'
MQIACH_CLWL_CHANNEL_WEIGHT	1579	X'0000062B'
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'

Tabulka 187. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACH_DISPS_MAX	1586	X'00000632'
MQIACH_SSLTASKS_STARTED	1587	X'00000633'
MQIACH_SSLTASKS_MAX	1588	X'00000634'
MQIACH_CURRENT_CHL	1589	X'00000635'
MQIACH_CURRENT_CHL_MAX	1590	X'00000636'
MQIACH_CURRENT_CHL_TCP	1591	X'00000637'
MQIACH_CURRENT_CHL_LU62	1592	X'00000638'
MQIACH_ACTIVE_CHL	1593	X'00000639'
MQIACH_ACTIVE_CHL_MAX	1594	X'0000063A'
MQIACH_ACTIVE_CHL_PAUSED	1595	X'0000063B'
MQIACH_ACTIVE_CHL_STARTED	1596	X'0000063C'
MQIACH_ACTIVE_CHL_STOPPED	1597	X'0000063D'
MQIACH_ACTIVE_CHL_RETRY	1598	X'0000063E'
MQIACH_LISTENER_STATUS	1599	X'0000063F'
MQIACH_SHARED_CHL_RESTART	1600	X'00000640'
MQIACH_LISTENER_CONTROL	1601	X'00000641'
MQIACH_BACKLOG	1602	X'00000642'
MQIACH_XMITQ_TIME_INDICATOR	1604	X'00000644'
MQIACH_NETWORK_TIME_INDICATOR	1605	X'00000645'
MQIACH_EXIT_TIME_INDICATOR	1606	X'00000646'
MQIACH_BATCH_SIZE_INDICATOR	1607	X'00000647'
MQIACH_XMITQ_MSGS_AVAILABLE	1608	X'00000648'
MQIACH_CHANNEL_SUBSTATE	1609	X'00000649'
MQIACH_SSL_KEY_RESETS	1610	X'0000064A'
MQIACH_COMPRESSION_RATE	1611	X'0000064B'
MQIACH_COMPRESSION_TIME	1612	X'0000064C'
MQIACH_MAX_XMIT_SIZE	1613	X'0000064D'
MQIACH_DEF_CHANNEL_DISP	1614	X'0000064E'
MQIACH_SHARING_CONVERSATIONS	1615	X'0000064F'
MQIACH_MAX_SHARING_CONVS	1616	X'00000650'
MQIACH_CURRENT_SHARING_CONVS	1617	X'00000651'
MQIACH_MAX_INSTANCES	1618	X'00000652'
MQIACH_MAX_INSTS_PER_CLIENT	1619	X'00000653'
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'

Tabulka 187. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIACH_MULTICAST_PROPERTIES	1626	X'0000065A'
MQIACH_NEW_SUBSCRIBER_HISTORY	1627	X'0000065B'
MQIACH_MC_HB_INTERVAL	1628	X'0000065C'
MQIACH_USE_CLIENT_ID	1629	X'0000065D'
MQIACH_MQTT_KEEP_ALIVE	1630	X'0000065E'
MQIACH_IN_DOUBT_IN	1631	X'0000065F'
MQIACH_IN_DOUBT_OUT	1632	X'00000660'
MQIACH_MSGS_SENT<	1633	X'00000661'
MQIACH_MSGS_RECEIVED	1634	X'00000662'
MQIACH_MSGS_RCVD	1634	X'00000662'
MQIACH_PENDING_OUT	1635	X'00000663'
MQIACH_AVAILABLE_CIPHERSPECS	1636	X'00000664'
MQIACH_MATCH	1637	X'00000665'
MQIACH_USER_SOURCE	1638	X'00000666'
MQIACH_WARNING	1639	X'00000667'
MQIACH_DEF_RECONNECT	1640	X'00000668'
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_PROTOCOL	1643	X'0000066B'
MQIACH_AMQPKEEPALIVE	1644	X'0000066C'
MQIACH_SECURITY_PROTOCOL	1645	X'0000066D'
z/OS V 9.2.0 MQIACH_SPL_PROTECTION	1646	X'0000066E'
MQIACH_LAST_USED	1646	X'0000066E'

MQIAMO_ * (Typy parametrů parametru monitorování celého formátu příkazu)

Tabulka 188. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIAMO_FIRST	701	X'000002BD'
VELIKOST DÁVKY 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'
SELHÁNÍ PŘÍKAZU MQIAMO_BROWSES_FAILED	708	X'000002C4'
MQIAMO_CLOSES.	709	X'000002C5'
MQIAMO_COMMITS	710	X'000002C6'
VOLÁNÍ MQIAMO_COMMITS_FAILED	711	X'000002C7'
MQIAMO_CONNS	712	X'000002C8'

Tabulka 188. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAMO_CONNS_MAX	713	X'000002C9'
MQIAMO_DISKY	714	X'000002CA'
IMPLICITNÍ HODNOTA MQIAMO_DISCS_IMPLICIT	715	X'000002CB'
MQIAMO_DISC_TYPE	716	X'000002CC'
MQIAMO_EXIT_TIME_AVG	717	X'000002CD'
MQIAMO_EXIT_TIME_MAX	718	X'000002CE'
MQIAMO_EXIT_TIME_MIN	719	X'000002CF'
MQIAMO_FULL_BATS	720	X'000002D0'
MQIAMO_GENERATED_MSGS	721	X'000002D1'
MQIAMO_GETS	722	X'000002D2'
MQIAMO_GET_MAX_BYTE	723	X'000002D3'
MQIAMO_GET_MIN_BYTE	724	X'000002D4'
SELHÁNÍ MQIAMO_GETS_FAILED	725	X'000002D5'
MQIAMO_INCOMPLETE_BATS	726	X'000002D6'
MQIAMO_INQS	727	X'000002D7'
MQIAMO_MSGS	728	X'000002D8'
MQIAMO_NET_TIME_AVG	729	X'000002D9'
MQIAMO_NET_TIME_MAX	730	X'000002DA'
MQIAMO_NET_TIME_MIN	731	X'000002DB'
MQIAMO_OBJECT_COUNT	732	X'000002DC'
MQIAMO_OPENS	733	X'000002DD'
MQIAMO_PUT1S	734	X'000002DE'
MQIAMO_PUT	735	X'000002DF'
MQIAMO_PUT_MAX_BAJTŮ	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'
SELHÁNÍ MQIAMO_CONNS_FAILED	749	X'000002ED'
VOLÁNÍ MQIAMO_OPENS_FAILED	751	X'000002EF'
SELHÁNÍ MQIAMO_INQS_FAILED	752	X'000002F0'
SELHÁNÍ MQIAMO_SETS_FAILED	753	X'000002F1'
SELHÁNÍ FUNKCE MQIAMO_PUTS_FAILED.	754	X'000002F2'
MQIAMO_PUT1S_FAILED	755	X'000002F3'
MQIAMO_CLOSES_FAILED	757	X'000002F5'

Tabulka 188. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAMO_MSGS_EXPIRED	758	X'000002F6'
MQIAMO_MSGS_NOT_QUEUED	759	X'000002F7'
MQIAMO_MSGS_UVOLNĚNO	760	X'000002F8'
MQIAMO_SUBSC_DUR	764	X'000002FC'
MQIAMO_SUBSC_NDUR	765	X'000002FD'
MQIAMO_SUBSC_FAILED	766	X'000002FE'
MQIAMO_SUBRQS	767	X'000002FF'
VOLÁNÍ MQIAMO_SUBRQS_FAILED	768	X'00000300'
MQIAMO_CBS.	769	X'00000301'
SELHÁNÍ MQIAMO_CBS_FAILED	770	X'00000302'
MQIAMO_CTLS	771	X'00000303'
SELHÁNÍ MQIAMO_CTLS_FAILED	772	X'00000304'
MQIAMO_STATS.	773	X'00000305'
SELHÁNÍ 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'
MQIAMO_SUB_NDUR_LOWWATER	778	X'0000030A'
MQIAMO_TOPIC_PUTS	779	X'0000030B'
FUNKCE 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_UNSUBSC_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'
REŽIM MQIAMO_FEEDBACK_MODE	793	X'00000319'
TYP_MQIAO_RELIBILITY_TYPE	794	X'0000031A'
MQIAMO_LACE_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'

Tabulka 188. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAMO_MCAST_BATCH_TIME	802	X'00000322'
MQIAMAI_MCAST_HEARTBEAT	803	X'00000323'
MQIAMO_DEST_DATA_PORT	804	X'00000324'
MQIAMO_DEPAIR_PORT	805	X'00000325'
MQIAMO_ACKS_RCVD	806	X'00000326'
AKTÉŘI MQIAMO_ACTIVE_ACKERS	807	X'00000327'
MQIAMO_PKTS_SENT	808	X'00000328'
MQIAMO_CELKOVÝ_OPRAVA_PKT	809	X'00000329'
MQIAMO_TOTAL_PKTS_SENT	810	X'0000032A'
MQIAMO_TOTAL_MSGS_SENT	811	X'0000032B'
MQIAMO_TOTAL_BYTES_SENT	812	X'0000032C'
MQIAMO_NUM_STREAMS	813	X'0000032D'
MQIAMO_ACK_FEEDBACK	814	X'0000032E'
ZPĚTNÁ VAZBA MQIAMO_NACK_FEEDBACK	815	X'0000032F'
ZTRACENÁ MQIAMO_PKTS_LOST	816	X'00000330'
MQIAMO_MSGS_RCVD	817	X'00000331'
MQIAMO_MSG_BYTES_RCVD	818	X'00000332'
MQIAMO_MSGS_DELIVERS	819	X'00000333'
ZPRACOVANÉ MQIAMO_PKTS_PROCESSED	820	X'00000334'
MQIAMO_PKTS_DLVD	821	X'00000335'
MQIAMO_PKTS_DROPPED	822	X'00000336'
MQIAMO_PKTS_DUPLICATED	823	X'00000337'
VYTVOŘENÉ 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_REPACE_PKTS_RCVD	831	X'0000033F'
MQIAMO_TOTAL_REPACE_PKTS_RQSTD	832	X'00000340'
MQIAMO_TOTAL_MSGS_PROCESSED	833	X'00000341'
MQIAMO_TOTAL_MSGS_SELECTED	834	X'00000342'
VYPRŠELA PLATNOST MQIAMO_TOTAL_MSGS_EXPIRED	835	X'00000343'
MQIAMO_TOTAL_MSGS_DELIVERED,	836	X'00000344'
MQIAMO_TOTAL_MSGS_RETURNED	837	X'00000345'
MQIAMO_LAST_USED	837	X'00000345'

MQIAMO64_* (64bitové typy parametrů monitorování s 64bitovým systémem)

Tabulka 189. 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 celého systému)

Tabulka 190. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
NADABY_PRVNÍ	-1	X'FFFFFFFF'
ID_SADY_ZÁSADY_MQIASY_CODE_LIST_ID	-1	X'FFFFFFFF'
TYP MQIAS_TYPE	-2	X'FFFFFFFE'
PŘÍKAZ MQIAS_COMMAND	-3	X'FFFFFFFD'
MQIADY_MSG_SEQ_NUMBER	-4	X'FFFFFFFC'
MQIARY_CONTROL	-5	X'FFFFFFFB'
MQIAS_COMP_CODE	-6	X'FFFFFFFA'
NADÁVACÍ_DŮVOD	-7	X'FFFFFFF9'
MQIAFY_BAG_OPTIONS	-8	X'FFFFFFF8'
NADÁVACÍ_VERZE	-9	X'FFFFFFF7'
MQIABY_LAST_USED	-9	X'FFFFFFF7'
NADABY_LAST	-2000	X'FFFFF830'

MQIAUT_ * (IMS Ověřovatel záhlaví informací)

Tabulka 191. Konstantní názvy a hodnoty

Název	Hodnota
MQIAUT_NONE	"¬¬¬¬¬¬¬¬"
MQIAUT_NON_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQIAV_ * (hodnoty celočíselných atributů)

Tabulka 192. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
MQIAV_UNDEFINED	-2	X'FFFFFFFE'

MQICM_ * (IMS -režimy vázaného zpracování záhlaví informací)

Tabulka 193. Konstantní názvy a hodnoty

Název	Hodnota
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

MQIDO_ * (Nejisté volby příkazového formátu)

Tabulka 194. 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 195. Konstrukce konstant

Název	Struktura
ID_KONSTRUKCE_MQIEP_	"IEP¬"
MQIEP_STRUC_ID_POLE	'I','E','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 196. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIEP_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQDXP_CURRENT_VERSION	1	X'00000001'

MQIGQ_ * (řazení do front v rámci skupiny)

Tabulka 197. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

MQIGQPA_ * (funkce řazení do front v rámci skupiny)

Tabulka 198. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQIGQPA_DEFAULT	1	X'00000001'

Tabulka 198. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KONTEXT MQIGQPA_CONTEXT	2	X'00000002'
MQIGQPA_ONLY_IGQ	3	X'00000003'
MQIGQPA_ALTERNATE_NEBOIGQ	4	X'00000004'

MQIIH_ * (struktura a příznaky záhlaví informačního obsahu produktu IMS)

Struktura záhlaví informací produktu IMS

Tabulka 199. Konstrukce konstant	
Název	**Struktura**
MQIIH_STRUCTION_ID	"IIH¬"
MQIIH_STRUC_ID_POLE	'I','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 200. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIIH_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQIIH_VERSION	1	X'00000001'
MQIIH_LENGTH_1	84	X'00000054'

Parametry záhlaví informací produktu IMS

Tabulka 201. 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_ * (Dotaz na vlastnosti a strukturu vlastností zprávy)

Zjistit strukturu vlastností vlastností zprávy

Tabulka 202. Konstrukce konstant	
Název	**Struktura**
MQIMPO_STRUCT	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 203. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIMPO_VERSION_1	1	X'00000001'

Tabulka 203. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VERZE AKTUÁLNÍ_VERZE MQIMPO_CURRENT_VERSION	1	X'00000001'

Zjistit volby vlastností zprávy

Tabulka 204. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
TYP 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'
HODNOTA MQIMPO_CONVERT_VALUE	32	X'00000020'
MQIMPO_NONE	0	X'00000000'

MQINBD_ * (vstupní a výstupní formát příkazu)

Tabulka 205. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQINBD_Q_MGR	0	X'00000000'
SKUPINA MQINBD_GROUP	3	X'00000003'

MQIND_ * (Speciální hodnoty indexu)

Tabulka 206. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIND_NONE	-1	X'FFFFFFFF'
MQINDAL_VŠE	-2	X'FFFFFFFE'

MQIPADDR_ * (Verze IP adres)

Tabulka 207. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

MQISS_ * (IMS informační záhlaví Security Scopes)

Tabulka 208. Konstantní názvy a hodnoty	
Název	**Hodnota**
KONTROLA MQISS_CHECK	'C'
MQISS_FULL	'F'

MQIT_ * (Typy indexů)

Tabulka 209. 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_GROUP_ID	5	X'00000005'

MQITEM_ * (typ položky pro příkaz mqInquireItemInfo)

Tabulka 210. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQITEM_INTEGER	1	X'00000001'
ŘETĚZEC MQITEM_STRING	2	X'00000002'
MQITEM_BAG	3	X'00000003'
ŘETĚZEC MQITEM_BYTE_STRING	4	X'00000004'
MQITEM_INTEGER_FILTER	5	X'00000005'
FILTR MQITEM_STRING_FILTER	6	X'00000006'
MQITEM_INTEGER64	7	X'00000007'
FILTR MQITEM_BYTE_STRING_FILTER	8	X'00000008'

MQITII_ * (IMS Identifikátor instance transakce záhlaví informací)

Tabulka 211. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQITII_NONE	X'00...00' (16 nul)
MQITII_NON_ARRAY	'\0','\0',... (16 nul)

MQITS_ * (IMS Transaction States Transaction States)

Tabulka 212. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQITS_IN_CONVERSATION	'C'
MQITS_NOT_IN_CONVERSATION	'¬'
MQITS_ARCHITECTED	'A'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQKAI_ * (intervalKeepAlive)

Tabulka 213. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQKAI_AUTO	-1	X'FFFFFFFF'

MQMASTER_ * (hlavní administrace)

Tabulka 214. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMASTER_NO	0	X'00000000'
MQMASTER_ANO	1	X'00000001'

MQMCAS_ * (Stav agenta Message Channel Agent Status)

Tabulka 215. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCAS_STOPPED	0	X'00000000'
MQMCAS_RUNNING	3	X'00000003'

MQMCAT_ * (typy MCA)

Tabulka 216. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PROCES MQMCAT_PROCESS	1	X'00000001'
MQMCAT_THREAD	2	X'00000002'

MQMCD_ * (Informace o značce voleb publikování/odběru)

Značky typu mcd (Publish/Subscribe Options Tag Content Descriptor)

Tabulka 217. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCD_FOLDER_VERSION	1	X'00000001'

Názvy značek voleb značek publikování/odběru

Tabulka 218. Konstantní názvy a hodnoty	
Název	**Hodnota**
DOMÉNA MQMCD_MSG_DOMAIN	"Msd"
MQMCD_MSG_SET	"Set"
MQMCD_MSG_TYPE	"Type"
FORMÁT MQMCD_MSG_FORMAT	"Fmt"

Názvy značek XML značek voleb pro publikování/odběr

Tabulka 219. Konstantní názvy 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>"

Tabulka 219. Konstantní názvy a hodnoty (pokračování)	
Název	**Hodnota**
MQMCD_MSG_TYPE_E	"</Type>"
MQMCD_MSG_FORMÁT_B	"<Fmt>"
MQMCD_MSG_FORMÁT_E	"</Fmt>"

Hodnoty značek značek voleb publikování/odběru

Tabulka 220. Konstantní názvy 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"
OBJEKT MQMCD_DOMAIN_JMS_OBJECT	"jms_object"
MAPA MQMCD_DOMAIN_JMS_MAP	"jms_map"
MQMCD_DOMAIN_JMS_STREAM	"jms_stream"
MQMCD_DOMAIN_JMS_BYTES	"jms_bytes"

MQMD_ * (Struktura deskriptoru zpráv)

Tabulka 221. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQM_STRUCT	"MD¬¬"
POLE MQMD_STRUC_ID_ARRAY	'M','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 222. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMD_VERSION_1	1	X'00000001'
MQMD_VERSION_2	2	X'00000002'
VERZE MQM_AKTUÁLNÍ_VERZE	2	X'00000002'

MQMDE_ * (Struktura rozšíření deskriptoru zpráv)

Tabulka 223. Konstrukce konstant	
Název	**Struktura**
MQM_STRUCTION_ID	"MDE¬"
MQM_STRUC_STRUC_ID_POLE	'M','D','E','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 224. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDE_VERSION_2	2	X'00000002'

Tabulka 224. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQM_AKTUÁLNÍ_VERZE	2	X'00000002'
MQMDE_LENGTH_2	72	X'00000048'

MQMDEF_ * (parametry rozšíření deskriptoru zpráv)

Tabulka 225. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDEF_NONE	0	X'00000000'

MQMDS_ * (Posloupnost doručení zprávy)

Tabulka 226. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PRIORITA MQMS_PRIORITY	0	X'00000000'
MQSD_FIFO	1	X'00000001'

MQMF_ * (Příznaky zprávy)

Tabulka 227. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMF_SEGMENTATION_BLOKOVÁNO	0	X'00000000'
MQMF_SEGMENTATION_ALLOWED	1	X'00000001'
MQMF_MSG_IN_GROUP	8	X'00000008'
MQM_LAST_MSG_IN_GROUP	16	X'00000010'
SEGMENT MQMF_SEGMENT	2	X'00000002'
MQMF_LAST_SEGMENT	4	X'00000004'
MQMF_NONE	0	X'00000000'

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

Struktura volby pro zpracování zpráv do vyrovnávací paměti

Tabulka 228. Konstrukce konstant	
Název	**Struktura**
MQMHBO_STRUCTION_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M','H','B','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 229. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMHBO_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQMHBO_CURRENT_VERSION	1	X'00000001'

Volby popisovače zpráv do vyrovnávací paměti

Tabulka 230. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMHBO_PROPERTIES_IN_MQRFH2	1	X'00000001'
VLASTNOSTI MQMHBO_DELETE_PROPERTIES	2	X'00000002'
MQMHBO_NONE	0	X'00000000'

MQMI_ * (Identifikátor zprávy)

Tabulka 231. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQMI_NONE	X'00...00' (24 nul)
MQMI_NONE_ARRAY.	'\0','\0',... (24 nul)

MQMMBI_ * (časový interval procházení zpráv)

Tabulka 232. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMMBI_UNLIMITED	-1	X'FFFFFFFF'

MQMO_ * (Volby shody)

Tabulka 233. 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_OFSET	16	X'00000010'
MQMO_MATCH_MSG_TOKEN	32	X'00000020'
MQMO_NONE	0	X'00000000'

MQMODE_ * (volby režimu formátu příkazu)

Tabulka 234. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMODE_FORCE	0	X'00000000'
MQMODE_QUIESCE	1	X'00000001'
UKONČENÍ MQMODE_TERMINATE	2	X'00000002'

MQMON_ * (monitorování hodnot)

Tabulka 235. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMON_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQMON_NONE	-1	X'FFFFFFFF'

Tabulka 235. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMON_Q_MGR	-3	X'FFFFFFFD'
MQMON_OFF	0	X'00000000'
MQMON_ON	1	X'00000001'
MQMON_DISABLED	0	X'00000000'
MQMON_POVOLENO	1	X'00000001'
MQMON_LOW	17	X'00000011'
MQMON_MEDIUM	33	X'00000021'
MQMON_HIGH	65	X'00000041'

MQMT_ * (typy zpráv)

Tabulka 236. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQM_SYSTEM_FIRST	1	X'00000001'
POŽADAVEK MQMT_REQUEST	1	X'00000001'
MQMT_REPLY	2	X'00000002'
MQM_DATAGRAM	8	X'00000008'
SESTAVA MQMT_REPORT	4	X'00000004'
MQM_MQ_FIELDS_FROM_MQE	112	X'00000070'
POLE MQMT_MQE_FIELDS	113	X'00000071'
MQM_SYSTEM_LAST	65535	X'0000FFFF'
MQM_APPL_FIRST	65536	X'00010000'
MQM_APPL_LAST	999999999	X'3B9AC9FF'

MQMTOK_ * (Token zprávy)

Tabulka 237. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQMTOK_NONE	X'00...00' (16 nul)
MQMTOKEN_NE_ARRAY	'\0','\0',... (16 nul)

Tabulka 238. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMTOK_NONE	X'00...00'	(16 nulls)
MQMTOKEN_NE_ARRAY	'\0','\0',...	(16 nulls)

MQNC_ * (Počet názvů)

Tabulka 239. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POČET NÁZVŮ MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

MQNPM_ * (přechodná třída zpráv)

Tabulka 240. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNPM_CLASS_NORMAL	0	X'00000000'
VYSOKÁ HODNOTA MQNPM_CLASS_HIGH	10	X'0000000A'

MQNPMS_ * (NonPersistent-Rychlosti zpráv)

Tabulka 241. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

MQNT_ * (Typy seznamu názvů)

Tabulka 242. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQNT_NONE	0	X'00000000'
MQNT_Q	1	X'00000001'
KLASTR MQNT_CLUSTER	2	X'00000002'
MQNT_AUTH_INFO	4	X'00000004'
MQNT_ALL	1001	X'000003E9'

MQNVS_ * (Názvy pro řetězec název/hodnoty)

Tabulka 243. Konstantní názvy a hodnoty	
Název	**Hodnota**
TYP_APLIK MQNVS_	"OPT_APP_GRP¬"
TYP_ZPRÁVA_MQNVS_	"OPT_MSG_TYPE¬"

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQOA_ * (Omezení pro selektory pro atributy objektu)

Tabulka 244. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQOA_	1	X'00000001'
MQOA_LAST	9000	X'00002328'

MQOD_ * (Struktura deskriptoru objektu)

Tabulka 245. Konstrukce konstant	
Název	**Struktura**
MQOD_STRUCTION_ID	"OD¬¬"
MQOD_STRUC_ID_POLE	'O','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 246. 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'
VERZE AKTUÁLNÍ_VERZE	4	X'00000004'
AKTUÁLNÍ_DÉLKA_AKTUÁLNÍ_HODNOTY	(value differs by platform or version)	(value differs by platform or version)

MQOII_ * (Identifikátor instance objektu)

Tabulka 247. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQOII_NONE	X'00...00' (24 nul)
MQOII_NON_ARRAY	'\0','\0',... (24 nul)

MQOL_ * (Původní délka)

Tabulka 248. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOL_UNDEFINED	-1	X'FFFFFFFF'

MQOM_ * (Zastaralé volby zpráv produktu Db2 ve skupině zjišťování)

Tabulka 249. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOM_NO	0	X'00000000'
MQOM_ANO	1	X'00000001'

MQOO_ * (Otevřít volby)

Tabulka 250. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOO_BIND_AS_Q_DEF	0	X'00000000'
MQOO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQO_INPUT_AS_Q_DEF	1	X'00000001'
MQO_INPUT_SHARED	2	X'00000002'
MQO_INPUT_EXCLUSIVE	4	X'00000004'
MQOOK_BROWSE	8	X'00000008'
MQOOK_VÝSTUP	16	X'00000010'
MQO_ DOTÁZAT SE	32	X'00000020'
MQOOK_SADA	64	X'00000040'
ÁLNÍ_KONTEXT MQOO_SAVE_ALL_CONTEXT	128	X'00000080'
KONTEXT MQOO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQOO_PASS_ALL_CONTEXT, KONTEXT	512	X'00000200'

Tabulka 250. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KONTEXT MQOO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQO_SET_ALL_CONTEXT,	2048	X'00000800'
MQO_ALTERNATE_USER_AUTHORITY.	4096	X'00001000'
UVÁDĚNÍ MQOO_FAIL_IF_QUIESCING	8192	X'00002000'
MQO_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'
SKUPINA MQO_BIND_ON_GROUP	4194304	X'00400000'

*MQOO_ * (následující použití v jazyce C++)*

Tabulka 251. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NÁZVY MQOO_RESOLVE_NAMES	65536	X'00010000'
MQOO_RESOLVE_LOCAL_Q,	262144	X'00040000'

MQOP_ * (Operační kódy pro MQCTL a MQCB)

Operační kódy pro MQCTL

Tabulka 252. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOP_START	1	X'00000001'
MQOP_START_WAIT	2	X'00000002'
MQOP_STOP	4	X'00000004'

Operační kódy pro MQCB

Tabulka 253. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOP_REGISTER	256	X'00000100'
MQOP_DEREGISTRACI	512	X'00000200'

Operační kódy pro MQCTL a MQCB

Tabulka 254. 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 255. Hodnoty konstant

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

MQOPER_ * (Operace aktivity)

Tabulka 256. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
NEJPRVE ZADEJTE MQOPER_SYSTEM_SYSTEM	0	X'00000000'
MQOPER_UNKNOWN	0	X'00000000'
MQOPER_BROWSE	1	X'00000001'
VYŘADIT MQOPERACE_	2	X'00000002'
MQOPER_GET	3	X'00000003'
MQOPER_PUT	4	X'00000004'
MQOPER_PUT_REPLY	5	X'00000005'
ZPRÁVA MQOPER_PUT_REPORT	6	X'00000006'
MQOPER_RECEIVE	7	X'00000007'
MQOPERAČNÍ_ODESLÁNÍ	8	X'00000008'
TRANSFORMAČNÍ ALGORITMUS MQOPER_	9	X'00000009'
MQOPER_PUBLISH	10	X'0000000A'
OPERACE MQOPER_EXCLUDED_PUBLISH	11	X'0000000B'
OPERACE MQOPER_DISCARDED_PUBLISH	12	X'0000000C'
MQOPER_SYSTEM_LAST	65535	X'0000FFFF'
NEJPRVE MQOPER_APPL_FIRST	65536	X'00010000'
MQOPER_APPL_LAST	999999999	X'3B9AC9FF'

MQOT_ * (typy objektů a rozšířené typy objektů)

Typy objektů

Tabulka 257. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQOT_NONE	0	X'00000000'
MQOT_Q	1	X'00000001'
MQO_NAMELIST	2	X'00000002'
PROCES MQOT_PROCESS	3	X'00000003'
TŘÍDA 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'

Tabulka 257. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOL_CF_STRUKTURY	10	X'0000000A'
MQOT_LISTENER	11	X'0000000B'
SLUŽBA MQOT_SERVICE	12	X'0000000C'
MQOT_RESERVED_1	999	X'000003E7'

Rozšířené typy objektů

Tabulka 258. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOT_ALL	1001	X'000003E9'
ALIAS_ALIAS_DATABÁZE	1002	X'000003EA'
MQOK_MODEL_Q	1003	X'000003EB'
MQOT_LOKÁLNÍ_Q	1004	X'000003EC'
MQOK_VZDÁLENÝ_Q	1005	X'000003ED'
MQOT_SENDER_CHANNEL	1007	X'000003EF'
MQOT_SERVER_CHANNEL	1008	X'000003F0'
MQOT_REQUESTER_CHANNEL	1009	X'000003F1'
MQOT_RECEIVER_CHANNEL	1010	X'000003F2'
MQOT_AKTUÁLNÍ_KANÁL	1011	X'000003F3'
MQOT_ULOŽENÝ_KANÁL	1012	X'000003F4'
MQOT_SVRCONN_CHANNEL	1013	X'000003F5'
MQOT_CLNTCONN_CHANNEL	1014	X'000003F6'
MQOT_SHORT_CHANNEL	1015	X'000003F7'
MQOT_CHLAUTH	1016	X'000003F8'
MQOT_VZDÁLENÝ_NÁZEV_MGR_NAME	1017	X'000003F9'
MQOT_PROTO_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 259. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ HODNOTA MQPA_DEFAULT	1	X'00000001'
KONTEXT MQPA_CONTEXT	2	X'00000002'
POUZE MQPA_ONLY_MCA	3	X'00000003'
MQPA_ALTERNATE_NEBO_MCA	4	X'00000004'

MQPD_ * (Deskriptor vlastnosti, podpora a kontext)

Struktura deskriptoru vlastností

Tabulka 260. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY OBJEKTU MQPD_BEAN	"PD¬¬"
POLE MQPD_STRUC_ID_ARRAY	'P','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 261. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQPD_CURRENT_VERSION	1	X'00000001'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Volby deskriptoru vlastností

Tabulka 262. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NONE	0	X'00000000'

Volby podpory vlastností

Tabulka 263. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PODPORA MQPD_SUPPORT_OPTIONAL	1	X'00000001'
POŽADOVÁNA PODPORA 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 264. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NO_CONTEXT	0	X'00000000'
KONTEXT MQPD_USER_CONTEXT	1	X'00000001'

MQPER_ * (hodnoty perzistence)

Tabulka 265. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NOT_PERSISTENT	0	X'00000000'

Tabulka 265. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPER_PERSISTENT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

MQPL_ * (platformy)

Tabulka 266. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPL_MVS	1	X'00000001'
MQPL_OS390	1	X'00000001'
NEZOS_MQPL_	1	X'00000001'
MQPL_OS2	2	X'00000002'
MQPL_AIX	3	X'00000003'
MQPL_UNIX	3	X'00000003'
MQPL_OS400	4	X'00000004'
OKNA MQPL_WINDOWS	5	X'00000005'
POČ 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'
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
ZAŘÍZENÍ MQPL_APPLIANCE	28	X'0000001C'
MQPL_NATIVNÍ	1	X'00000001'

MQPMO_ * (Vložení voleb zpráv a struktury pro masku publikování)

Vložit strukturu voleb zprávy

Tabulka 267. Konstrukce konstant	
Název	**Struktura**
MQPMO_STRUC_ID	"PMO¬"
MQPMO_STRUC_ID_POLE	'P','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 268. 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'
AKTUÁLNÍ_VERZE MQPMO_AKTUÁLNÍ_VERZE	3	X'00000003'

Tabulka 268. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
AKTUÁLNÍ_DÉLKA MQPMO_LENGTH	(value differs by platform or version)	(value differs by platform or version)

Volby vložení zprávy

Tabulka 269. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_SYNCPOINT	2	X'00000002'
MQPMO_NE_SYNCPOINT	4	X'00000004'
MQPMO_VÝCHOZÍ_KONTEXT	32	X'00000020'
MQPMO_NOVÉ_ID_ZPRÁVY	64	X'00000040'
MQPMO_NOVÉ_ KOREL_ID	128	X'00000080'
KONTEXT MQPMO_PASS_IDENTITY_CONTEXT	256	X'00000100'
MQPMO_PASS_ALL_CONTEXT	512	X'00000200'
KONTEXT MQPMO_SET_IDENTITY_CONTEXT	1024	X'00000400'
MQPMO_SET_ALL_CONTEXT	2048	X'00000800'
MQPMO_ALTERNATE_USER_AUTHORITY	4096	X'00001000'
UVÁDĚNÍ MQPMO_FAIL_IF_QUIESCING	8192	X'00002000'
MQPMOTO_NE_KONTEXT	16384	X'00004000'
MQPMO_LOGICAL_ORDER	32768	X'00008000'
MQPMO_ASYNC_RESPONSE	65536	X'00010000'
MQPMO_SYNC_RESPONSE	131072	X'00020000'
MQPMOD_RESOLVE_LOKÁLNÍ_Q	262144	X'00040000'
MQPMO_RETAIN	2097152	X'00200000'
MQPMOD_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_ODEZVA_NA_DOBA_Q_DEF	0	X'00000000'
MQPMO_RESPONSE_AS_TOPIC_DEF	0	X'00000000'
MQPMO_NONE	0	X'00000000'

Volby vložení zpráv pro masku publikování

Tabulka 270. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_POPB_VOLBA_VOLBY	2097152	X'00200000'

MQPMRF_ * (Vložení polí záznamu zprávy)

Tabulka 271. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMRF_ID_ZPRÁVY	1	X'00000001'

Tabulka 271. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMRF_CORREL_ID	2	X'00000002'
ID SKUPINY MQPMRF_GROUP_ID	4	X'00000004'
ZPĚTNÁ VAZBA MQPMRF_FEEDBACK	8	X'00000008'
MQPMRF_ACCOUNTING_TOKEN	16	X'00000010'
MQPMRF_NONE	0	X'00000000'

MQPO_ * (Volby uvolnění příkazu ve formátu příkazu)

Tabulka 272. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPO_YES	1	X'00000001'
MQPO_NO	0	X'00000000'

MQPRI_* (priorita)

Tabulka 273. 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_ * (kontrolní hodnoty vlastností fronty a kanálu a maximální délka vlastností)

Řídicí hodnoty vlastností fronty a kanálu

Tabulka 274. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KOMPATIBILITA MQPROP_COMPATIBILITY	0	X'00000000'
MQPROP_NONE	1	X'00000001'
MQPROP_ALL	2	X'00000002'
MQPROP_FORCE_MQRFH2	3	X'00000003'

Maximální délka vlastností

Tabulka 275. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

MQPRT_ * (Put Response Values)

Tabulka 276. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'

Tabulka 276. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPRT_SYNC_RESPONSE	1	X'00000001'
ODEZVA MQPRT_ASYNC_RESPONSE	2	X'00000002'

MQPS_ * (Publikování/odběr)

Stav publikování/odběru ve formátu příkazu

Tabulka 277. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPS_STATUS_INACTIVE	0	X'00000000'
STAV_STAV_MQP	1	X'00000001'
STAV_STAV_MQPS_STOPPING	2	X'00000002'
STAV MQPS_STATUS_ACTIVE	3	X'00000003'
MQPS_STATUS_COMPAT	4	X'00000004'
CHYBA_STAV_MQP	5	X'00000005'
STAV MQPS_STATUS_REFUSED	6	X'00000006'

Značky publikování/odběru jako řetězce

PŘÍKAZ MQPS_	"MQPSCommand"
MQPS_COMP_CODE	"MQPSCompCode"
MQPS_CORREL_ID	"MQPSCorrelId"
VOLBY MQPS_DELETE_OPTIONS	"MQPSDelOpts"
ID_CHYBY MQPS_ERROR_ID	"MQPSErrorId"
MQPS_ERROR_POS	"MQPSErrorPos"
MQPS_INTEGER_DATA	"MQPSIntData"
PARAMETR MQPS_PARAMETER_ID	"MQPSParmId"
VOLBY PUBLIKOVÁNÍ MQPS_PUBLICTION_OPTIONS	"MQPSPubOpts"
ČASOVÉ RAZÍTKO MQPS_PUBLISH_TIMESTAMP	"MQPSPubTime"
MQPS_Q_MGR_NAME	"MQPSQMgrName"
NÁZEV MQPS_Q_NAME	"MQPSQName"
MQPS_REASON	"MQPSReason"
MQPS_REASON_TEXT	"MQPSReasonText"
VOLBY REGISTRACE MQPS_REGISTRATION_OPTIONS	"MQPSRegOpts"
MQPS_SEQUENCE_NUMBER	"MQPSSeqNum"
NÁZEV_PROUDU MQPS_STREAM_NAME	"MQPSStreamName"
MQPS_STRING_DATA	"MQPSStringData"
MQPS_SUBSCRIPTION_IDENTITY	"MQPSSubIdentity"
MQPS_SUBSCRIPTION_NAME	"MQPSSubName"

MQPS_SUBSCRIPTION_USER_DATA	"MQPSSubUserData"
TÉMA MQPS_TOPIC	"MQPSTopic"
ID_UŽIVATELE_MQPS_	"MQPSUserId"

Značky publikování/odběru jako prázdné uzavřené řetězce

PŘÍKAZ MQPS_COMMAND_B	"¬MQPSCommand¬"
MQPS_COMP_CODE_B	"¬MQPSCompCode¬"
MQPS_CORRELA_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_NÁZEVB	"¬MQPSQName¬"
MQPS_REASON_B	"¬MQPSReason¬"
MQPS_REASON_TEXT_B	"¬MQPSReasonText¬"
VOLBY REGISTRACE MQPS_REGISTRATIONS_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_PUBLICATION	"DeletePub"
MQPS_DEREGISTER_PUBLISHER	"DeregPub"
MQPS_DEREGISTER_SUBSCRIBER	"DeregSub"
PUBLIKOVÁNÍ MQPS_PUBLISH	"Publish"
MQPS_REGISTER_PUBLISHER	"RegPub"

MQPS_REGISTER_SUBSCRIBER	"RegSub"
MQPS_REQUEST_UPDATE	"ReqUpdate"

Hodnoty značek příkazů 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

NÁZEV_ADR_MQP	"AddName"
ANONYMNÍ MQPS_ANONYMOUS	"Anon"
MQPS_CORRELAC_ID_AS_IDENTITY	"CorrelAsId"
MQPS_DEREGISTER_ALL	"DeregAll"
MQPS_DIRECT_REQUESTS	"DirectReq"
DUPLIKACE_MQPS_DUPLICATES_OK	"DupsOK"
MQPS_FULL_RESPONSE	"FullResp"
MQPS_INCLUDE_STREAM_NAME	"InclStreamName"
MQPS_INFORCE_IF_RETAINED	"InformIfRet"
MQPS_IS_RETAINED_PUBLICATION	"IsRetainedPub"
MQPS_JOIN_EXCLUSIVE	"JoinExcl"
MQPS_JOIN_SHARED	"JoinShared"
POUZE MQPS_LEAVA_ONLY	"LeaveOnly"
LOKÁLNÍ MQPS_LOCAL	"Local"
MQPS_LOCKED	"Locked"
POUZE NOVÉ_VEŘEJNÉ_PUBLIKACE_MQPS_ONLY	"NewPubsOnly"
ZMĚNA MQPS_NO_ALTERING	"NoAlter"
MQPS_NO_REGISTRATION	"NoReg"
MQPS_NON_PERSISTENT	"NonPers"
MQPS_NONE	"None"
POUZE MQPS_OTHER_SUBSCRIBERS_ONLY	"OtherSubsOnly"
MQPS_PERSISTENT	"Pers"

MQPS_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPS_PERSISTENT_AS_Q	"PersAsQueue"
MQPS_PUBLISH_ON_REQUEST_ONLY	"PubOnReqOnly"
MQPS_RETAIN_PUBLICATION	"RetainPub"
ID_UŽIVATELE_PROMĚNNÉ_MQP	"VariableUserId"

Hodnoty značek voleb publikování/odběru jako prázdné uzavřené řetězce

NÁZEV_SADY_MQ_ADR_B	"¬AddName¬"
MQPS_ANONYMOUS_B	"¬Anon¬"
MQPS_CORRELAC_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_INFORCE_IF_RETAINED_B	"¬InformIfRet¬"
MQPS_IS_RETAINED_PUBLICATION_B	"¬IsRetainedPub¬"
MQPS_JOIN_EXCLUSIVE_B	"¬JoinExcl¬"
SDÍLENÝ_SDÍLENÝ_NÁZEV_SDÍLENÉ_FRONTY	"¬JoinShared¬"
POUZE MQPS_LEAV_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_NON_B	"¬None¬"
POUZE KEM_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_PUBLICTION_B	"¬RetainPub¬"
MQPS_VARIABLE_USER_ID_B	"¬VariableUserId¬"

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQPSC_ * (volby publikování/odběru značek publikování/odběru ve složkách příkazu psc)

Tabulka 278. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSC_FOLDER_VERSION	1	X'00000001'

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

PŘÍKAZ MQPSC_	"Command"
VOLBA MQPSC_REGISTRATION_OPTION	"RegOpt"
VOLBA MQPSC_PUBLICATION_OPTION	"PubOpt"
VOLBA MQPSC_DELETE_OPTION	"DelOpt"
MQPSC_TOPIC-TÉMA	"Topic"
MQPSC_SUBSCRIPTION_POINT	"SubPoint"
FILTR MQPSC_FILTER	"Filter"
MQPSC_Q_MGR_NAME	"QMgrName"
NÁZEV QPSC_Q_NAME	"QName"
ČASOVÉ RAZÍTKO 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 publikování/odběru značek XML)*

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_FILTR_B	"<Filter>"
MQPSC_FILTR_E	"</Filter>"

MQPSC_Q_MGR_NAME_B	"<QMgrName>"
MQPSC_Q_MGR_NÁZEVE	"</QMgrName>"
MQPSC_Q_NÁZEVB	"<QName>"
MQPSC_QNAME_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_CORRELA_ID_B	"<CorrelId>"
MQPSC_CORRELA_ID_E	"</CorrelId>"

*MQPSC_ * (Volby vydavatele značek publikování/odběru jako řetězce)*

MQPSC_DELETE_PUBLICATION	"DeletePub"
MQPSC_DEREGISTER_SUBSCRIBER	"DeregSub"
PUBLIKOVÁNÍ MQPSC_PUBLISH	"Publish"
MQPSC_REGISTER_SUBSCRIBER	"RegSub"
MQPSC_REQUEST_UPDATE	"ReqUpdate"

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

MQPSC_NÁZEV_ADR.	"AddName"
MQPSC_CORRELAC_ID_AS_IDENTITY	"CorrelAsId"
MQPSC_DEREGISTER_ALL	"DeregAll"
MQPSC_DUPLICATES_OK	"DupsOK"
MQPSC_FULL_RESPONSE	"FullResp"
MQPSC_INFORMACE_IF_RETAINED	"InformIfRet"
MQPSC_IS_RETAINED_PUB	"IsRetainedPub"
MQPSC_JOIN_SHARED	"JoinShared"
MQPSC_JOIN_EXCLUSIVE	"JoinExcl"
POUZE PRO MQPSC_LEAVE_ONLY	"LeaveOnly"
MQPSC_LOCAL	"Local"
MQPSC_LOCKED	"Locked"

POUZE MQPSC_NEW_PUBS_ONLY	"NewPubsOnly"
ÚPRAVA MQPSC_NO_ALTERING	"NoAlter"
MQPSC_NON_PERSISTENT	"NonPers"
POUZE MQPSC_OTHER_SUBSC_ONLY	"OtherSubsOnly"
MQPSC_PERSISTENT	"Pers"
MQPSC_PERSISTENT_AS_PUBLISH	"PersAsPub"
MQPSC_PERSISTENT_AS_Q	"PersAsQueue"
MQPSC_NONE	"None"
POUZE MQPSC_PUT_ON_REQUEST_ONLY	"PubOnReqOnly"
MQPSC_RETAIN_PUB	"RetainPub"
MQPSC_PROMĚNNÁ_ID_UŽIVATELE	"VariableUserId"

MQPSCR_ * (Volby publikování/odběru)

Značky publikování/Odběry značek publikování/odběru značek (pscr)

Tabulka 279. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQPSCR_FOLDER_VERSION	1	X'00000001'

Názvy značek voleb značek publikování/odběru

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

Názvy značek XML značek voleb pro publikování/odběr

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

Hodnoty značek značek voleb publikování/odběru

MQPSCR_OK	"ok"
VAROVÁNÍ MQPSCR_WARNING	"warning"
CHYBA MQPSCR_ERROR	"error"

MQPSM_ * (režim/odběr-režim)

Tabulka 280. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSM_DISABLED	0	X'00000000'
MQPSM_COMPAT	1	X'00000001'
MQPSM_ENABLED	2	X'00000002'

MQPSPROP_ * (Publikování publikování/odběru zpráv)

Tabulka 281. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSPROP_NONE	0	X'00000000'
MQPSPROP_COMPAT	1	X'00000001'
MQPSPROP_RFH2	2	X'00000002'
MQPSPROP_MSGPROP	3	X'00000003'

MQPSST_ * (typ příkazu publikování/odběru ve formátu příkazu)

Tabulka 282. 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 283. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPUBO_NONE	0	X'00000000'
MQPUBO_CORRELA_ID_AS_IDENTITY	1	X'00000001'
MQPUBO_RETAIN_PUBLICATION	2	X'00000002'
POUZE MQPUBO_OTHER_SUBSCRIBERS_ONLY	4	X'00000004'
MQPUBO_NO_REGISTRATION	8	X'00000008'
MQPUBO_IS_RETAINED_PUBLICATION	16	X'00000010'

MQPXP_ * (Struktura výstupního parametru směrování publikování/odběru)

Tabulka 284. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQPXP_STRUCT	"PXP¬"
POLE MQPXP_STRUC_ID_ARRAY	'P','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 285. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPXP_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQPXP_CURRENT_VERSION	1	X'00000001'

MQQA_ * (atributy fronty)

Blokování hodnot získání

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

Blokovat hodnoty Put

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

Sdílitelnost fronty

Tabulka 288. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

Upevňování zadního zatížení

Tabulka 289. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MACKY_BACKOUT_HARDENED	1	X'00000001'
MQQA_BACKUT_NOT_HARDENED	0	X'00000000'

MQQDT_ * (typy definic fronty)

Tabulka 290. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQDT_PREDEFINED	1	X'00000001'
MQQDT_PERMANENT_DYNAMIC	2	X'00000002'
MQQDT_DOČASNÝ_DYNAMICKÝ	3	X'00000003'
DYNAMICKÝ_SDÍLENÝ_ADRESÁŘ_MQQQ	4	X'00000004'

MQQF_ * (Příznaky fronty)

Tabulka 291. 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 definičních typů správce front ve formátu příkazu)

Tabulka 292. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
ODESILATEL MQQMDT_EXPLICIT_CLUSTER_	1	X'00000001'
ODESILATEL MQQMDT_AUTO_CLUSTER_	2	X'00000002'
ODESILATEL MQQMDT_AUTOEXP_CLUSTER_	4	X'00000004'
PŘIJÍMAČ MQQMDT_CLUSTER_	3	X'00000003'

MQQMF_ * (parametry správce front)

Tabulka 293. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMF_REPOSITORY_Q_MGR	2	X'00000002'
MQQMF_CLUSDR_USER_DEFINED	8	X'00000008'
MQQMF_CLUSDR_AUTO_DEFINED	16	X'00000010'
MQQMF_AVAILABLE	32	X'00000020'

MQQMFAC_ * (Zařízení správce front správce front)

Tabulka 294. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MOST MQQMFAC_IMS_BRIDGE	1	X'00000001'
MQQMFAC_DB2	2	X'00000002'

MQQMSTA_ * (Stav správce front správce front)

Tabulka 295. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMSTA_STARTING	1	X'00000001'
MQQMSTA_RUNNING	2	X'00000002'
MQQMSTA_QUIESCING	3	X'00000003'

MQQMT_ * (Typy příkazů správce front formátu příkazů)

Tabulka 296. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQMT_NORMAL	0	X'00000000'
ÚLOŽIŠTĚ MQQMT_REPOSITORY	1	X'00000001'

MQQO_ * (Volby uvedení příkazu do klidového stavu)

Tabulka 297. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQO_ANO	1	X'00000001'
MQQO_0	0	X'00000000'

MQQSGD_ * (skupina sdílející skupinu sdílení front)

Tabulka 298. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQSGD_VŠE	-1	X'FFFFFFFF'
MQQSGD_Q_MGR	0	X'00000000'
MQQSD_KOPIE	1	X'00000001'
SDÍLENÝ MQQSGD_SHARED	2	X'00000002'
SKUPINA MQQSGD_GROUP	3	X'00000003'
MQQSGD_PRIVATE	4	X'00000004'
MQQSSGD_LIVE	6	X'00000006'

MQQSGS_ * (Stav skupiny sdílení front ve formátu příkazu)

Tabulka 299. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
NEZNÁMÉ MQQSSGS_UNKNOWN	0	X'00000000'
VYTVOŘENÉ MQQSGS_CREATED	1	X'00000001'
MQQSSGS_ACTIVE	2	X'00000002'
MQQSSGS_INACTIVE	3	X'00000003'
SELHÁNÍ MQQSGS_FAILED	4	X'00000004'
NEVYŘÍZENÉ MQQSGS_PENDING	5	X'00000005'

MQQSIE_ * (Fronta zpráv-fronta událostí-Interval událostí)

Tabulka 300. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQSIE_NONE	0	X'00000000'
MQQSIE_HIGH	1	X'00000001'
MQQSIE_OK	2	X'00000002'

MQQSO_ * (Otevřené volby stavu fronty příkazů pro SET, BROWSE, INPUT)

Tabulka 301. 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 příkazu-typy otevření-typy otevření)

Tabulka 302. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQSOT_ALL	1	X'00000001'
MQQSOT_INPUT	2	X'00000002'
VÝSTUP MQQSOT_OUTPUT	3	X'00000003'

MQQSUM_ * (Zprávy ve stavu fronty nepotvrzené zprávy)

Tabulka 303. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQSUM_ANO	1	X'00000001'
MQQSUM_NO	0	X'00000000'

MQQT_ * (typy fronty a rozšířené typy front)

Typy front

Tabulka 304. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQ_LOCAL	1	X'00000001'
MQQ_MODEL	2	X'00000002'
ALIAS MQQ_ALIAS	3	X'00000003'
MQQT_REMOTE	6	X'00000006'
KLASTR MQQ_CLUSTER	7	X'00000007'

Rozšířené typy front

Tabulka 305. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQQ_VŠE	1001	X'000003E9'

MQRC_ * (kódy příčiny)

Tabulka 306. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_NONE	0	X'00000000'
NEJPRVE MQRC_APPL_FIRST	900	X'00000384'
MQRC_APPL_LAST	999	X'000003E7'
CHYBA MQRC_ALIAS_BASE_Q_TYPE_ERROR	2001	X'000007D1'
MQRC_ALREADY_CONNECTED	2002	X'000007D2'
MQRC_BACKED_OUT	2003	X'000007D3'
CHYBA MQRC_BUFFER_ERROR	2004	X'000007D4'
CHYBA_MQRC_BUFFER_LENGTH_ERROR	2005	X'000007D5'
MQRC_CHAR_ATTR_LENGTH_ERROR	2006	X'000007D6'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRC_CHAR_ATTRS_ERROR	2007	X'000007D7'
MQRC_CHAR_ATTRS_TOO_SHORT	2008	X'000007D8'
PORCC_CONNECTION_CONNECTION_LO	2009	X'000007D9'
CHYBA MQRC_DATA_LENGTH_ERROR	2010	X'000007DA'
CHYBA MQRC_DYNAMIC_Q_NAME_ERROR	2011	X'000007DB'
CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR	2012	X'000007DC'
MQRC_EXPIRY_ERROR	2013	X'000007DD'
CHYBA MQRC_FEEDBACK_ERROR	2014	X'000007DE'
MQRC_GET_INHIBITED	2016	X'000007E0'
MQRC_HANDLE_NOT_AVAILABLE	2017	X'000007E1'
CHYBA MQRC_HCONN_ERROR	2018	X'000007E2'
CHYBA MQRC_HOBJ_ERROR	2019	X'000007E3'
CHYBA MQRC_INHIBIT_VALUE_ERROR	2020	X'000007E4'
CHYBA_MQRC_INT_ATTR_COUNT_ERROR	2021	X'000007E5'
ÚČ_PŘÍST_OBR_MQRC_ATR_ATTR_TOO_SMALL	2022	X'000007E6'
CHYBA POLE MQRC_INT_ATTRS_ARRAY_ERROR	2023	X'000007E7'
MQRC_SYNCPOINT_LIMIT_REACHED	2024	X'000007E8'
MQRC_MAX_CONNS_LIMIT_DOSAŽEN	2025	X'000007E9'
CHYBA MQRC_MD_ERROR	2026	X'000007EA'
MQRC_MISSING_REPLY_TO_Q	2027	X'000007EB'
CHYBA MQRC_MSG_TYPE_ERROR	2029	X'000007ED'
MQRC_MSG_TOO_BIG_FOR_Q	2030	X'000007EE'
MQRC_MSG_TOO_BIG_FOR_Q_MGR	2031	X'000007EF'
MQRC_NO_MSG_AVAILABLE	2033	X'000007F1'
MQRC_NO_MSG_UNDER_CURSOR	2034	X'000007F2'
AUTORIZOVANÝ 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_DOTÁZAT SE	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'
CHYBA MQRC_OBJECT_TYPE_ERROR	2043	X'000007FB'
CHYBA MQRC_OD_ERROR	2044	X'000007FC'
MQRC_OPTION_NOT_VALID_FOR_TYPE	2045	X'000007FD'
CHYBA MQRC_OPTIONS_ERROR	2046	X'000007FE'
CHYBA MQRC_PERSISTENCE_ERROR	2047	X'000007FF'
MQRC_PERSISTENT_NOT_ALLOWED	2048	X'00000800'

Tabulka 306. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRC_PRIORITY_EXCEEDS_MAXIMUM	2049	X'00000801'
CHYBA MQRC_PRIORITY_ERROR	2050	X'00000802'
MQRC_PUT_BLOKOVÁNO	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'
CHYBA MQRC_Q_TYPE_ERROR	2057	X'00000809'
CHYBA MQRC_Q_MGR_NAME_ERROR	2058	X'0000080A'
MQRC_Q_MGR_NOT_AVAILABLE	2059	X'0000080B'
CHYBA NEPOVINNOSTI_SESTAVY_MQRC_REPORT	2061	X'0000080D'
MQRC_SECOND_MARK_NOT_ALLOWED	2062	X'0000080E'
MQRC_SECURITY_ERROR	2063	X'0000080F'
CHYBA MQRC_SELECTOR_COUNT_ERROR	2065	X'00000811'
MQRC_SELECTOR_LIMIT_EXCEEDED	2066	X'00000812'
CHYBA MQRC_SELECTOR_ERROR	2067	X'00000813'
MQRC_SELECTOR_NOT_FOR_TYPE	2068	X'00000814'
MQRC_SIGNAL_OUTSTANDING	2069	X'00000815'
MQRC_SIGNAL_REQUEST_ACCEPTED	2070	X'00000816'
MQRC_STORAGE_NOT_AVAILABLE	2071	X'00000817'
MQRC_SYNCPOINT_NOT_AVAILABLE	2072	X'00000818'
CHYBA ŘÍZENÍ MQRC_TRIGGER_CONTROL_ERROR	2075	X'0000081B'
CHYBA MQRC_TRIGGER_DEPTH_ERROR	2076	X'0000081C'
MQRC_TRIGGER_MSG_PRIORITY_ERR	2077	X'0000081D'
CHYBA MQRC_TRIGGER_TYPE_ERROR	2078	X'0000081E'
MQRC_TRUNCATED_MSG_ACCEPTED	2079	X'0000081F'
OPERACE 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'
CHYBA MQRC_WAIT_INTERVAL_ERROR	2090	X'0000082A'
CHYBA MQRC_XMIT_Q_TYPE_ERROR	2091	X'0000082B'
CHYBA 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'
CHYBA OBJEKTU MQRC_CONTEXT_HANDLE_ERROR	2097	X'00000831'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_CONTEXT_NOT_AVAILABLE	2098	X'00000832'
MQRC_SIGNAL1_ERROR	2099	X'00000833'
MQRC_OBJECT_ALREADY_EXISTS	2100	X'00000834'
MQRC_OBJECT_DAMAGED	2101	X'00000835'
PROBLÉM MQRC_RESOURCE_PROBLEM	2102	X'00000836'
PŘIPOJENÉ MQRC_ANOTHER_Q_MGR_CONNECTED	2103	X'00000837'
VOLBA MQRC_UNKNOWN_REPORT_OPTION	2104	X'00000838'
CHYBA TŘÍDY MQRC_STORAGE_CLASS_ERROR	2105	X'00000839'
MQRC_CED_NOT_VALID_FOR_XCF_Q	2106	X'0000083A'
ZRUŠENÉ MQRC_XWAIT_CANCELED	2107	X'0000083B'
CHYBA MQRC_XWAIT_ERROR	2108	X'0000083C'
MQRC_SUPPRESSED_BY_EXIT	2109	X'0000083D'
CHYBA MQRC_FORMAT_ERROR	2110	X'0000083E'
CHYBA MQRC_SOURCE_CCSID_ERROR	2111	X'0000083F'
MQRC_SOURCE_INTEGER_ENC_ERROR	2112	X'00000840'
MQRC_SOURCE_DECIMAL_ENC_ERROR	2113	X'00000841'
CHYBA MQRC_SOURCE_FLOAT_ENC_ERROR	2114	X'00000842'
CHYBA 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_OSEKNUTO	2120	X'00000848'
MQRC_NO_EXTERNAL_PARTICIPANTS	2121	X'00000849'
MQRC_PARTICIPANT_NOT_AVAILABLE	2122	X'0000084A'
MQRC_OUTCOME_MIXED	2123	X'0000084B'
NEVYŘÍZENÉ MQRC_OUTCOME_PENDING	2124	X'0000084C'
MQRC_BRIDGE_STARTED	2125	X'0000084D'
MQRC_BRIDGE_STOPPED	2126	X'0000084E'
MQRC_ADAPTER_STORAGE_NEDOSTATEK	2127	X'0000084F'
MQRC_UOW_IN_PROGRESS	2128	X'00000850'
CHYBA MQRC_ADAPTER_CONN_LOAD_ERROR	2129	X'00000851'
CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR	2130	X'00000852'
CHYBA MQRC_ADAPTER_DEFS_ERROR	2131	X'00000853'
MQRC_ADAPTER_DEFS_LOAD_ERROR	2132	X'00000854'
CHYBA MQRC_ADAPTER_CONV_LOAD_ERROR	2133	X'00000855'
CHYBA MQRC_BO_ERROR	2134	X'00000856'
CHYBA MQRC_DH_ERROR	2135	X'00000857'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_MULTIPLE_PŘÍČINY	2136	X'00000858'
FUNKCE MQRC_OPEN_FAILED	2137	X'00000859'
CHYBA MQRC_ADAPTER_DIC_LOAD_ERROR	2138	X'0000085A'
CHYBA MQRC_CNO_ERROR	2139	X'0000085B'
MQRC_CICS_WAIT_FAILED	2140	X'0000085C'
CHYBA MQRC_DLH_ERROR	2141	X'0000085D'
CHYBA MQRC_HEADER_ERROR	2142	X'0000085E'
CHYBA MQRC_SOURCE_LENGTH_ERROR	2143	X'0000085F'
MQRC_TARGET_LENGTH_ERROR	2144	X'00000860'
CHYBA MQRC_SOURCE_BUFFER_ERROR	2145	X'00000861'
MQRC_TARGET_BUFFER_ERROR	2146	X'00000862'
CHYBA MQRC_IIH_ERROR	2148	X'00000864'
CHYBA MQRC_PCF_ERROR	2149	X'00000865'
CHYBA MQRC_DBCS_ERROR	2150	X'00000866'
CHYBA MQRC_OBJECT_NAME_ERROR	2152	X'00000868'
CHYBA MQRC_OBJECT_Q_MGR_NAME_ERROR	2153	X'00000869'
CHYBA MQRC_RECS_PRESENT_ERROR	2154	X'0000086A'
CHYBA MQRC_OBJECT_RECORDS_ERROR	2155	X'0000086B'
CHYBA MQRC_RESPONSE_RECORDS_ERROR	2156	X'0000086C'
NESROVNALOST MQRC_ASID_	2157	X'0000086D'
CHYBOVÁ_CHYBA MQRC_PMO_RECORD_FLAGS_ERROR	2158	X'0000086E'
CHYBA MQRC_PUT_MSG_RECORDS_ERROR	2159	X'0000086F'
MQRC_CONN_ID_IN_USE	2160	X'00000870'
UVÁDĚNÍ MQRC_Q_MGR_QUIESCING	2161	X'00000871'
MQRC_Q_MGR_STOPPING	2162	X'00000872'
MQRC_DUPLICATE_RECOV_COORD	2163	X'00000873'
CHYBA MQRC_PMO_ERROR	2173	X'0000087D'
MQRC_API_EXIT_NOT_FOUND	2182	X'00000886'
CHYBA MQRC_API_EXIT_LOAD_ERROR	2183	X'00000887'
CHYBA MQRC_REMOTE_Q_NAME_ERROR	2184	X'00000888'
MQRC_INCONSISTENT_PERSISTENCE	2185	X'00000889'
CHYBA MQRC_GMO_ERROR	2186	X'0000088A'
OMEZENÍ MQRC_CICS_BRIDGE_RESTRICTION	2187	X'0000088B'
UŽIVATELSKÁ PROCEDURA MQRC_STOPPED_BY_CLUSTER_EXIT	2188	X'0000088C'
CHYBA MQRC_CLUSTER_RESOLVUTION_ERROR	2189	X'0000088D'
MQRC_CONVERTED_STRING_TOO_BIG	2190	X'0000088E'
CHYBA MQRC_TMC_	2191	X'0000088F'
ÚPLNÁ OPERACE MQRC_PAGESET_FULL	2192	X'00000890'
MQRC_STORAGE_MEDIUM_FULL	2192	X'00000890'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA OBJEKTU MQRC_PAGESET_ERROR	2193	X'00000891'
MQRC_NAME_NE_VALID_STAR_TYP	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'
CHYBA MQRC_DEF_XMIT_Q_TYPE_ERROR	2198	X'00000896'
CHYBA 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'
ZASTAVIT_PŘIPOJENÍ_MQRC	2203	X'0000089B'
MQRC_ADAPTER_NOT_AVAILABLE	2204	X'0000089C'
CHYBA MQRC_MSG_ID_	2206	X'0000089E'
CHYBA MQRC_CORRELA_ID_ERROR	2207	X'0000089F'
CHYBA MQRC_FILE_SYSTEM_ERROR	2208	X'000008A0'
MQRC_NO_MSG_LOCKED	2209	X'000008A1'
CHYBA MQRC_SOAP_DOTNET_ERROR	2210	X'000008A2'
CHYBA MQRC_SOAP_AXIS_ERROR	2211	X'000008A3'
CHYBA MQRC_SOAP_URL_ERROR	2212	X'000008A4'
MQRC_FILE_NOT_AUDITED	2216	X'000008A8'
AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED	2217	X'000008A9'
MQRC_MSG_TOO_BIG_FOR_CHANNEL	2218	X'000008AA'
MQRC_CALL_IN_PROGRESS	2219	X'000008AB'
CHYBA MQRC_RMH_ERROR	2220	X'000008AC'
MQRC_Q_MGR_ACTIVE	2222	X'000008AE'
MQRC_Q_MGR_NOT_ACTIVE	2223	X'000008AF'
MQRC_Q_DEPTH_HIGH	2224	X'000008B0'
MQRC_Q_DEPTH_LOW	2225	X'000008B1'
MQRC_Q_SERVICE_INTERVAL_HIGH	2226	X'000008B2'
MQRC_Q_SERVICE_INTERVAL_OK	2227	X'000008B3'
CHYBA POLE MQRC_RFH_HEADER_FIELD_ERROR	2228	X'000008B4'
CHYBA MQRC_RAS_PROPERTY_ERROR	2229	X'000008B5'
MQRC_UNIT_WORK_NOT_STARTED	2232	X'000008B8'
MQRC_CHANNEL_AUTO_DEF_OK	2233	X'000008B9'
MQRC_CHANNEL_AUTO_DEF_ERROR	2234	X'000008BA'
CHYBA MQRC_CFH_ERROR	2235	X'000008BB'
CHYBA MQRC_CFIL_ERROR	2236	X'000008BC'
CHYBA MQRC_CFIN_ERROR	2237	X'000008BD'
CHYBA MQRC_CFSL_ERROR	2238	X'000008BE'

Tabulka 306. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
CHYBA MQRC_CFST_ERROR	2239	X'000008BF'
SKUPINA MQRC_INCOMPLETE_GROUP	2241	X'000008C1'
ZPRÁVA MQRC_INCOMPLETE_MSG	2242	X'000008C2'
MQRC_INCONSISTENT_CCCSIDS	2243	X'000008C3'
KÓDOVÁNÍ MQRC_INCONSISTENT_ENCODINGS	2244	X'000008C4'
NEKONZISTENCE MQRC_INCONSISTENT_UOW	2245	X'000008C5'
MQRC_INVALID_MSG_UNDER_CURSOR	2246	X'000008C6'
CHYBA MQRC_MATCH_OPTIONS_ERROR	2247	X'000008C7'
CHYBA MQRC_MDE_ERROR	2248	X'000008C8'
CHYBA MQRC_MSG_FLAGS_ERROR	2249	X'000008C9'
MQRC_MSG_SEQ_NUMBER_ERROR	2250	X'000008CA'
CHYBA MQRC_OFFSET_ERROR	2251	X'000008CB'
MQRC_ORIGINAL_LENGTH_ERROR	2252	X'000008CC'
MQRC_SEGMENT_LENGTH_ZERO	2253	X'000008CD'
MQRC_UOW_NOT_AVAILABLE	2255	X'000008CF'
MQRC_WRONG_GMO_VERSION	2256	X'000008D0'
VERZE MQRC_WRONG_MD_VERSION	2257	X'000008D1'
CHYBA MQRC_GROUP_ID_ERROR	2258	X'000008D2'
MQRC_INCONSISTENT_BROWSE	2259	X'000008D3'
CHYBA MQRC_XQHL_ERROR	2260	X'000008D4'
CHYBA MQRC_SRC_ENV_ERROR	2261	X'000008D5'
CHYBA MQRC_SRC_NAME_ERROR	2262	X'000008D6'
CHYBA MQRC_DEST_ENV_ERROR	2263	X'000008D7'
CHYBA MQRC_DEST_NAME_ERROR	2264	X'000008D8'
CHYBA MQRC_TM_ERROR	2265	X'000008D9'
CHYBA MQRC_CLUSTER_EXIT_ERROR	2266	X'000008DA'
CHYBA MQRC_CLUSTER_EXIT_LOAD_ERROR	2267	X'000008DB'
MQRC_CLUSTER_PUT_BLOKOVÁNO	2268	X'000008DC'
CHYBA MQRC_CLUSTER_RESOURCE_	2269	X'000008DD'
MQRC_NO_DESTINATIONS_AVAILABLE	2270	X'000008DE'
MQRC_CONN_TAG_IN_USE	2271	X'000008DF'
MQRC_PARTIALLY_CONVERTED	2272	X'000008E0'
CHYBA PŘIPOJENÍ MQRC_CONNECTION_ERROR	2273	X'000008E1'
CHYBA PROSTŘEDÍ MQRC_OPTION_ENVIRONMENT_ERROR	2274	X'000008E2'
CHYBA MQRC_CD_ERROR	2277	X'000008E5'
CHYBA MQRC_CLIENT_CONN_ERROR	2278	X'000008E6'
MQRC_CHANNEL_STOPPED_BY_USER	2279	X'000008E7'
CHYBA MQRC_HCONFIG_ERROR	2280	X'000008E8'
CHYBA FUNKCE MQRC_FUNCTION_ERROR	2281	X'000008E9'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
INICIALIZACE MQRC_INITIALIZATION_SELHALA	2286	X'000008EE'
SELHÁNÍ MQRC_TERMINATION_FAILED	2287	X'000008EF'
MQRC_UNKNOWN_Q_NAME	2288	X'000008F0'
CHYBA SLUŽBY MQRC_SERVICE_	2289	X'000008F1'
MQRC_Q_ALREADY_EXISTS	2290	X'000008F2'
MQRC_USER_ID_NOT_AVAILABLE	2291	X'000008F3'
ENTITA MQRC_UNKNOWN_ENTITY	2292	X'000008F4'
ENTITA MQRC_UNKNOWN_AUTH_ENTITY	2293	X'000008F5'
OBJEKT MQRC_UNKNOWN_REF_OBJECT	2294	X'000008F6'
MQRC_CHANNEL_ACTIVATED	2295	X'000008F7'
MQRC_CHANNEL_NOT_ACTIVATED	2296	X'000008F8'
ZRUŠENÉ MQRC_UOW_CANCELED	2297	X'000008F9'
PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED	2298	X'000008FA'
CHYBA MQRC_SELECTOR_TYPE_ERROR	2299	X'000008FB'
CHYBA MQRC_COMMAND_TYPE_ERROR	2300	X'000008FC'
CHYBA INSTANCE MQRC_MULTIPLE_INSTANCE_	2301	X'000008FD'
MQRC_SYSTEM_ITEM_NOT_ALTERABLE	2302	X'000008FE'
CHYBA MQRC_BAG_CONVERSION_ERROR	2303	X'000008FF'
MQRC_SELECTOR_OUT_OF_RANGE	2304	X'00000900'
MQRC_SELECTOR_NOT_UNIQUE	2305	X'00000901'
MQRC_INDEX_NOT_PRESENT	2306	X'00000902'
CHYBA MQRC_STRING_ERROR	2307	X'00000903'
MQRC_ENCODING_NOT_SUPPORTED	2308	X'00000904'
MQRC_SELECTOR_NOT_PRESENT	2309	X'00000905'
CHYBA MQRC_OUT_SELECTOR_ERROR	2310	X'00000906'
MQRC_STRING_ZKRÁCENÁ	2311	X'00000907'
MQRC_SELECTOR_NEOPRÁVNĚNÝ_TYP	2312	X'00000908'
MQRC_INCONSISTENT_ITEM_TYPE	2313	X'00000909'
CHYBA MQRC_INDEX_ERROR	2314	X'0000090A'
MQRC_SYSTEM_BAG_NOT_ALTERABLE	2315	X'0000090B'
CHYBA MQRC_ITEM_COUNT_ERROR	2316	X'0000090C'
MQRC_FORMAT_NOT_SUPPORTED, PODPOROVANÉ	2317	X'0000090D'
MQRC_SELECTOR_NOT_SUPPORTED	2318	X'0000090E'
CHYBA MQRC_ITEM_VALUE_ERROR	2319	X'0000090F'
CHYBA MQRC_HBAG_ERROR	2320	X'00000910'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_PARAMETER_MISSING	2321	X'00000911'
MQRC_CMD_SERVER_NOT_AVAILABLE	2322	X'00000912'
CHYBA MQRC_STRING_LENGTH_ERROR	2323	X'00000913'
CHYBA PŘÍKAZU MQRC_INQUIRY_COMMAND_ERROR	2324	X'00000914'
MQRC_NESTED_BAG_NOT_SUPPORTED	2325	X'00000915'
MQRC_BAG_NEOPRÁVNĚNÝ TYP	2326	X'00000916'
CHYBA MQRC_ITEM_TYPE_ERROR	2327	X'00000917'
MQRC_SYSTEM_BAG_NOT_DELETABLE	2328	X'00000918'
MQRC_SYSTEM_ITEM_NOT_DELETABLE	2329	X'00000919'
CHYBA MQRC_CODED_CHAR_SET_ID_ERROR	2330	X'0000091A'
CHYBA MQRC_MSG_TOKEN_ERROR	2331	X'0000091B'
MQRC_MISSING_WIH	2332	X'0000091C'
CHYBA MQRC_WIH_ERROR	2333	X'0000091D'
CHYBA MQRC_RFH_ERROR	2334	X'0000091E'
CHYBA MQRC_RFH_STRING_ERROR	2335	X'0000091F'
CHYBA PŘÍKAZU MQRC_RFH_COMMAND_ERROR	2336	X'00000920'
CHYBA MQRC_RFH_PARM_ERROR	2337	X'00000921'
MQRC_RFH_DUPLICATE_PARM	2338	X'00000922'
CHYBÍ MQRC_RFH_PARM_MISSING	2339	X'00000923'
CHYBA 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'
MQRC_CF_NOT_AVAILABLE	2345	X'00000929'
MQRC_CF_STRUC_IN_USE	2346	X'0000092A'
MQRC_CF_STRU_LIST_HDR_IN_USE	2347	X'0000092B'
MQRC_CF_STRUC_AUTH_FAILED	2348	X'0000092C'
CHYBA MQRC_CF_STRUC_STRUCT	2349	X'0000092D'
MQRC_CONN_TAG_NOT_USABLE	2350	X'0000092E'
KONFLIKT MQRC_GLOBAL_UOW_CONFLICT	2351	X'0000092F'
KONFLIKT MQRC_LOCAL_UOW_CONFLICT	2352	X'00000930'
FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW	2353	X'00000931'
CHYBA MQRC_UOW_ENLISTMENT_ERROR	2354	X'00000932'
MQRC_UOW_MIX_NOT_SUPPORTED	2355	X'00000933'
CHYBA MQRC_WXP_ERROR	2356	X'00000934'
CHYBA MQRC_CURRENT_RECORD_ERROR	2357	X'00000935'
CHYBA MQRC_NEXT_OFFSET_ERROR	2358	X'00000936'
MQRC_NO_RECORD_AVAILABLE	2359	X'00000937'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_OBJECT_LEVEL_INCOMPATIBLE	2360	X'00000938'
CHYBA MQRC_NEXT_RECORD_ERROR	2361	X'00000939'
MQRC_BACKOUT_THRESHOLD_REACHED	2362	X'0000093A'
MQRC_MSG_NOT_MATCHED	2363	X'0000093B'
CHYBA MQRC_JMS_FORMAT_ERROR	2364	X'0000093C'
MQRC_SEGMENTS_NOT_SUPPORTED	2365	X'0000093D'
MQRC_WRONG_CF_LEVEL	2366	X'0000093E'
OBJEKT MQRC_CONFIG_CREATE_OBJECT	2367	X'0000093F'
OBJEKT MQRC_CONFIG_CHANGE_OBJECT	2368	X'00000940'
MQRC_CONFIG_DELETE_OBJECT	2369	X'00000941'
OBJEKT MQRC_CONFIG_REFRESH_OBJECT	2370	X'00000942'
MQRC_CHANNEL_SSL_ERROR	2371	X'00000943'
MQRC_PARTICIPANT_NOT_DEFINED	2372	X'00000944'
MQRC_CF_STRU_FAILED	2373	X'00000945'
CHYBA MQRC_API_EXIT_ERROR	2374	X'00000946'
MQRC_API_EXIT_INIT_ERROR	2375	X'00000947'
CHYBA MQRC_API_EXIT_TERM_ERROR	2376	X'00000948'
MQRC_EXIT_REASON_ERROR	2377	X'00000949'
CHYBA MQRC_RESERVEED_VALUE_ERROR	2378	X'0000094A'
MQRC_NO_DATA_AVAILABLE	2379	X'0000094B'
CHYBA MQRC_SCO_ERROR	2380	X'0000094C'
CHYBA MQRC_KEY_REPOSITORY_ERROR	2381	X'0000094D'
CHYBA MQRC_CRYPTO_HARDWARE_ERROR	2382	X'0000094E'
MQRC_AUTH_INFO_REC_COUNT_ERROR	2383	X'0000094F'
MQRC_AUTH_INFO_REC_ERROR	2384	X'00000950'
CHYBA MQRC_AIR_ERROR	2385	X'00000951'
CHYBA MQRC_AUTH_INFO_TYPE_ERROR	2386	X'00000952'
CHYBA MQRC_AUTH_INFO_CONN_NAME_ERROR	2387	X'00000953'
CHYBA MQRC_LDAP_USER_NAME_ERROR	2388	X'00000954'
MQRC_LDAP_USER_NAME_LENGTH_ERR	2389	X'00000955'
CHYBA MQRC_LDAP_PASSWORD_ERROR	2390	X'00000956'
MQRC_SSL_ALREADY_INITIALIZOVÁNO	2391	X'00000957'
CHYBA MQRC_SSL_CONFIG_ERROR	2392	X'00000958'
CHYBA MQRC_SSL_INITIALIZATION_ERROR	2393	X'00000959'
CHYBA MQRC_Q_INDEX_TYPE_ERROR	2394	X'0000095A'
CHYBA MQRC_CFBS_ERROR	2395	X'0000095B'
MQRC_SSL_NOT_ALLOWED	2396	X'0000095C'
CHYBA MQRC_JSSE_ERROR	2397	X'0000095D'
NESROVNALOST MQRC_SSL_PEER_NAME_	2398	X'0000095E'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRC_SSL_PEER_NAME_ERROR	2399	X'0000095F'
MQRC_UNSUPPORTED_CIPHER_SUITE	2400	X'00000960'
MQRC_SSL_CERTIFICATE_ODVOLÁNO	2401	X'00000961'
CHYBA MQRC_SSL_CERT_STORE_ERROR	2402	X'00000962'
MQRC_CLIENT_EXIT_LOAD_ERROR	2406	X'00000966'
CHYBA 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'
STAV MQRC_LOGGER_STATUS	2411	X'0000096B'
MQRC_COMMAND_MQSC	2412	X'0000096C'
MQRC_COMMAND_PCF	2413	X'0000096D'
CHYBA MQRC_CFIF_ERROR	2414	X'0000096E'
CHYBA MQRC_CFSF_ERROR	2415	X'0000096F'
CHYBA MQRC_CFGR_ERROR	2416	X'00000970'
MQRC_MSG_NOT_ALLOWED_IN_GROUP	2417	X'00000971'
CHYBA MQRC_FILTER_OPERATOR_ERROR	2418	X'00000972'
CHYBA MQRC_NESTED_SELECTOR_ERROR	2419	X'00000973'
CHYBA MQRC_EF_ERROR	2420	X'00000974'
CHYBA MQRC_RFH_FORMAT_ERROR	2421	X'00000975'
CHYBA MQRC_CFBF_ERROR	2422	X'00000976'
KONFLIKT MQRC_CLIENT_CHANNEL_CONFLICT	2423	X'00000977'
CHYBA MQRC_SD_ERROR	2424	X'00000978'
CHYBA MQRC_TOPIC_STRING_ERROR	2425	X'00000979'
CHYBA MQRC_STS_ERROR	2426	X'0000097A'
MQRC_NO_SUBSCRIPTION	2428	X'0000097C'
MQRC_SUBSCRIPTION_IN_USE	2429	X'0000097D'
CHYBA MQRC_STAT_TYPE_ERROR	2430	X'0000097E'
MQRC_SUB_USER_DATA_ERROR	2431	X'0000097F'
MQRC_SUB_ALREADY_EXISTS	2432	X'00000980'
NESROVNALOST MQRC_IDENTITY_	2434	X'00000982'
MQRC_ALTER_SUB_ERROR	2435	X'00000983'
MQRC_DURABILITY_NOT_ALLOWED	2436	X'00000984'
MQRC_NO_RETAINED_MSG	2437	X'00000985'
CHYBA MQRC_SRO_ERROR	2438	X'00000986'
CHYBA MQRC_SUB_NAME_ERROR	2440	X'00000988'
CHYBA MQRC_OBJECT_STRING_ERROR	2441	X'00000989'
CHYBA MQRC_PROPERTY_NAME_ERROR	2442	X'0000098A'
MQRC_SEGMENTATION_NOT_ALLOWED	2443	X'0000098B'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRC_CBD_ERROR	2444	X'0000098C'
CHYBA 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'
CHYBA MQRC_SELECTOR_SYNTAX_ERROR	2459	X'0000099B'
CHYBA MQRC_HMSG_ERROR	2460	X'0000099C'
CHYBA MQRC_CMHO_ERROR	2461	X'0000099D'
CHYBA MQRC_DMHO_ERROR	2462	X'0000099E'
CHYBA MQRC_SMPO_ERROR	2463	X'0000099F'
CHYBA MQRC_IMPO_ERROR	2464	X'000009A0'
MQRC_PROPERTY_NAME_TOO_BIG	2465	X'000009A1'
MQRC_PROP_VALUE_NOT_CONVERTED	2466	X'000009A2'
MQRC_PROP_TYPE_NOT_SUPPORTED	2467	X'000009A3'
HODNOTA MQRC_PROPERTY_VALUE_TOO_BIG	2469	X'000009A5'
MQRC_PROP_CONV_NOT_SUPPORTED	2470	X'000009A6'
MQRC_PROPERTY_NOT_AVAILABLE	2471	X'000009A7'
CHYBA MQRC_PROP_NUMBER_FORMAT_ERROR	2472	X'000009A8'
CHYBA MQRC_PROPERTY_TYPE_ERROR	2473	X'000009A9'
VLASTNOSTI MQRC_PROPERTIES_TOO_BIG	2478	X'000009AE'
MQRC_PUT_NOT_RETAINED	2479	X'000009AF'
ZMĚNĚNO ALIAS_ALIAS_MQRC_TARGETYPE_CHANGED	2480	X'000009B0'
CHYBA MQRC_DMPO_ERROR	2481	X'000009B1'
CHYBA MQRC_PD_ERROR	2482	X'000009B2'
MQRC_CALLBACK_TYPE_ERROR	2483	X'000009B3'
CHYBA MQRC_CBD_OPTIONS_ERROR	2484	X'000009B4'
CHYBA MQRC_MAX_MSG_LENGTH_ERROR	2485	X'000009B5'
CHYBA MQRC_CALLBACK_ROUTINE_ERROR	2486	X'000009B6'
MQRC_CALLBACK_LINK_ERROR	2487	X'000009B7'
CHYBA OPERACE MQRC_OPERATION_ERROR	2488	X'000009B8'
CHYBA MQRC_BMHO_ERROR	2489	X'000009B9'
VLASTNOST 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'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_MSG_HANDLE_IN_USE	2499	X'000009C3'
MQRC_HCONN_ASYNC_ACTIVE	2500	X'000009C4'
CHYBA MQRC_MHBO_ERROR	2501	X'000009C5'
MQRC_PUBLICATION_FAILURE.	2502	X'000009C6'
MQRC_SUB_BLOKOVÁNO	2503	X'000009C7'
MQRC_SELECTOR_ALWAYS_FALSE	2504	X'000009C8'
CHYBA MQRC_XEPO_ERROR	2507	X'000009CB'
MQRC_DURABILITY_NOT_ALTERABLE	2509	X'000009CD'
MQRC_TOPIC_NOT_ALTERABLE	2510	X'000009CE'
MQRC_SUBLEVEL_NOT_ALTERABLE	2512	X'000009D0'
MQRC_PROPERTY_NAME_LENGTH_ERR	2513	X'000009D1'
MQRC_DUPLICATE_GROUP_SUB	2514	X'000009D2'
MQRC_GROUPING_NOT_ALTERABLE	2515	X'000009D3'
MQRC_SELECTOR_INVALID_FOR_TYPE	2516	X'000009D4'
MQRC_HOBJ_KLIDOVÉM STAVU	2517	X'000009D5'
MQRC_HOBJ_QUIESCED_NO_MSGS	2518	X'000009D6'
CHYBA MQRC_SELECTION_STRING_ERROR	2519	X'000009D7'
CHYBA MQRC_RES_OBJECT_STRING_ERROR	2520	X'000009D8'
POZASTAVIT_PŘIPOJENÍ_MQRC	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, DORUČENO	2526	X'000009DE'
MQRC_RFH_RESTRICTED_FORMAT_ERR	2527	X'000009DF'
FUNKCE MQRC_CONNECTION_STOPPED	2528	X'000009E0'
KONFLIKT MQRC_ASYNC_UOW_CONFLICT	2529	X'000009E1'
MQRC_ASYNC_XA_CONFLICT	2530	X'000009E2'
MQRC_PUBSUB_BLOKOVÁNO	2531	X'000009E3'
SELHÁNÍ PŘÍKAZU MQRC_MSG_HANDLE_COPY_FAILURE	2532	X'000009E4'
MQRC_DEST_CLASS_NOT_ALTERABLE	2533	X'000009E5'
POVOLENÁ OPERACE MQRC_OPERATION_NOT_ALLOWED	2534	X'000009E6'
CHYBA MQRC_ACTION_ERROR	2535	X'000009E7'
MQRC_CHANNEL_NOT_AVAILABLE	2537	X'000009E9'
MQRC_HOST_NOT_AVAILABLE	2538	X'000009EA'
CHYBA MQRC_CHANNEL_CONFIG_ERROR	2539	X'000009EB'
NÁZEV_KANÁLU_MQRC_UNKNOWN_CHANNE_NAME	2540	X'000009EC'
PUBLIKOVÁNÍ MQRC_LOOPING_PUBLICATION	2541	X'000009ED'
MQRC_ALREADY_JOINED	2542	X'000009EE'

Tabulka 306. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRC_CHANNEL_SSL_WARNING, VAROVÁNÍ	2552	X'000009F8'
CHYBA MQRC_OCSP_URL_ERROR	2553	X'000009F9'
MQRC_CIPHER_SPEC_NOT_SUITE_B	2591	X'00000A1F'
CHYBA MQRC_SUITE_B_CHYB	2592	X'00000A20'
CHYBA_OCHRANY MQRC_PASSWORD_ERROR	2594	X'00000A22'
CHYBA 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'
CHYBA MQRC_ENCODING_ERROR	6106	X'000017DA'
CHYBA MQRC_STRUC_ID_ERROR	6107	X'000017DB'
MQRC_NULL_POINTER	6108	X'000017DC'
ODKAZ MQRC_NO_CONNECTION_REFERENCE	6109	X'000017DD'
MQRC_NO_BUFFER	6110	X'000017DE'
CHYBA MQRC_BINARY_DATA_LENGTH_ERROR	6111	X'000017DF'
MQRC_BUFFER_NOT_AUTOMATIC	6112	X'000017E0'
MQRC_INSUFFICIENT_BUFFER	6113	X'000017E1'
MQRC_INSUFFICIENT_DATA	6114	X'000017E2'
MQRC_DATA_OŘÍZNUTÁ	6115	X'000017E3'
MQRC_ZERO_LENGTH	6116	X'000017E4'
MQRC_NEGATIVNÍ_DÉLKA	6117	X'000017E5'
MQRC_NEGATIVNÍ_POSUN	6118	X'000017E6'
FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT	6119	X'000017E7'
MQRC_INCONSISTENT_OBJECT_STATE	6120	X'000017E8'
MQRC_CONTEXT_OBJECT_NOT_VALID	6121	X'000017E9'
MQRC_CONTEXT_OPEN_ERROR	6122	X'000017EA'
CHYBA MQRC_STRUC_LENGTH_ERROR	6123	X'000017EB'
PŘIPOJENÍ MQRC_NOT_CONNECTED	6124	X'000017EC'
MQRC_NOT_OPEN	6125	X'000017ED'
MQRC_DISTRIBUTION_LIST_EMPTY	6126	X'000017EE'
VOLBY NEKONZISTENCE MQRC_INCONSISTENT_OPEN_OPTIONS	6127	X'000017EF'
MQRC_WRONG_VERSION	6128	X'000017F0'
CHYBA MQRC_REFERENCE_ERROR	6129	X'000017F1'

MQRCCF_ * (kódy příčiny záhlaví příkazového řádku)

Další informace o odpovědi programátora najdete v tématu Kódy příčiny příkazu PCF .

Tabulka 307. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
CHYBA MQRCCF_CFH_TYPE_ERROR	3001	X'00000BB9'
CHYBA MQRCCF_CFH_LENGTH_ERROR	3002	X'00000BBA'
CHYBA 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'
CHYBA PŘÍKAZU MQRCCF_CFH_COMMAND_ERROR	3007	X'00000BBF'
PŘÍKAZ MQRCCF_COMMAND_FAILED	3008	X'00000BC0'
CHYBA MQRCCF_CFIN_LENGTH_ERROR	3009	X'00000BC1'
CHYBA MQRCCF_CFST_LENGTH_ERROR	3010	X'00000BC2'
MQRCCF_CFST_STRHING_LENGTHER_ERR	3011	X'00000BC3'
CHYBA_FORCE_MQRCCF_FORCE_FORCE_	3012	X'00000BC4'
CHYBOVÝ_TYP_FRONTY_MQRCCF_STRUCTURE_ERROR	3013	X'00000BC5'
CHYBA MQRCCF_CFIN_PARM_ID_ERROR	3014	X'00000BC6'
CHYBA MQRCCF_CFST_PARM_ID_ERROR	3015	X'00000BC7'
CHYBA 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'
CHYBA MQRCCF_Q_TYPE_ERROR	3022	X'00000BCE'
CHYBA MQRCCF_MD_FORMAT_ERROR	3023	X'00000BCF'
CHYBA MQRCCF_CFSL_LENGTH_ERROR	3024	X'00000BD0'
CHYBA MQRCCF_REPLACE_VALUE_ERROR	3025	X'00000BD1'
HODNOTA MQRCCF_CFIL_DUPLICATE_VALUE	3026	X'00000BD2'
POČET CHYB: MQRCCF_CFIL_COUNT_ERROR	3027	X'00000BD3'
CHYBA MQRCCF_CFIL_LENGTH_ERROR	3028	X'00000BD4'
CHYBA MQRCCF_QUIESCE_VALUE_ERROR	3029	X'00000BD5'
CHYBA MQRCCF_MODE_VALUE_ERROR	3029	X'00000BD5'
MQRCCF_MSG_SEQ_NUMBER_ERROR	3030	X'00000BD6'
CHYBA MQRCCF_PING_DATA_COUNT_ERROR	3031	X'00000BD7'
CHYBA OBJEKTU MQRCCF_PING_PING_DATA_COMPARE_ERROR	3032	X'00000BD8'
CHYBA MQRCCF_CFSL_PARM_ID_ERROR	3033	X'00000BD9'
CHYBA MQRCCF_CHANNEL_TYPE_ERROR	3034	X'00000BDA'
CHYBA MQRCCF_PARM_SEQUENCE_ERROR	3035	X'00000BDB'
MQRCCF_XMIT_PROTOCOL_TYPY_CHYB	3036	X'00000BDC'
MQRCCF_BATCH_SIZE_ERROR	3037	X'00000BDD'
MQRCCF_DISC_INT_ERROR	3038	X'00000BDE'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRCCF_SHORT_RETRY_ERROR	3039	X'00000BDF'
CHYBA MQRCCF_SHORT_TIMER_ERROR	3040	X'00000BE0'
CHYBA MQRCCF_LONG_RETRY_ERROR	3041	X'00000BE1'
CHYBA MQRCCF_LONG_TIMER_ERROR	3042	X'00000BE2'
MQRCCF_SEQ_NUMBER_WRAP_ERROR	3043	X'00000BE3'
CHYBA MQRCCF_MAX_MSG_LENGTH_ERROR	3044	X'00000BE4'
CHYBA OBJEKTU MQRCCF_PUT_AUTH_ERROR	3045	X'00000BE5'
CHYBA MQRCCF_PURGE_VALUE_ERROR	3046	X'00000BE6'
CHYBA MQRCCF_CFIL_PARM_ID_ERROR	3047	X'00000BE7'
SOUBOR MQRCCF_MSG_ZKRÁCEN	3048	X'00000BE8'
CHYBA MQRCCF_CCSID_ERROR	3049	X'00000BE9'
CHYBA KÓDOVÁNÍ MQRCCF_ENCODING_ERROR	3050	X'00000BEA'
CHYBA MQRCCF_QUEUE_VALUE_ERROR	3051	X'00000BEB'
CHYBA MQRCCF_DATA_CONV_VALUE_ERROR	3052	X'00000BEC'
CHYBA MQRCCF_INDOUBT_VALUE_ERROR	3053	X'00000BED'
CHYBA MQRCCF_ESCAPE_TYPE_ERROR	3054	X'00000BEE'
CHYBA MQRCCF_REPOS_VALUE_ERROR	3055	X'00000BEF'
CHYBA MQRCCF_CHANNEL_TABLE_TABLE_ERROR	3062	X'00000BF6'
CHYBA MQRCCF_MCA_TYPE_ERROR	3063	X'00000BF7'
CHYBA MQRCCF_CHL_INST_TYPE_ERROR	3064	X'00000BF8'
MQRCCF_CHL_STATUS_NOT_FOUND	3065	X'00000BF9'
MQRCCF_CFSL_DUPLICATE_PARM	3066	X'00000BFA'
CHYBA MQRCCF_CFSL_TOTAL_LENGTH_ERROR	3067	X'00000BFB'
POČET CHYB: MQRCCF_CFSL_COUNT_ERROR	3068	X'00000BFC'
MQRCCF_CFSL_DÉLKA_CHYBOVÉ_CHYBY	3069	X'00000BFD'
MQRCCF_BROKER_DELETED	3070	X'00000BFE'
CHYBA MQRCCF_STREAM_ERROR	3071	X'00000BFF'
CHYBA MQRCCF_TOPIC_ERROR	3072	X'00000C00'
MQRCCF_NOT_REGISTERED	3073	X'00000C01'
CHYBA MQRCCF_Q_MGR_NAME_ERROR	3074	X'00000C02'
MQRCCF_INCORRECT_STREAM,	3075	X'00000C03'
CHYBA MQRCCF_Q_NAME_ERROR	3076	X'00000C04'
ZPRÁVA MQRCCF_NO_RETAINED_MSG	3077	X'00000C05'
MQRCCF_DUPLICATE_IDENTITY	3078	X'00000C06'
MQRCCF_INCORRECT_Q.	3079	X'00000C07'
CHYBA MQRCCF_CORREL_ID_ERROR	3080	X'00000C08'
AUTORIZOVANÝ OBJEKT MQRCCF_NOT_AUTHORIZED	3081	X'00000C09'
MQRCCF_UNKNOWN_STREAM	3082	X'00000C0A'
CHYBA OBJEKTU MQRCCF_REG_OPTIONS_ERROR	3083	X'00000C0B'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRCCF_PUT_OPTIONS_ERROR	3084	X'00000C0C'
ZPROSTŘEDKOVATEL MQRCCF_UNKNOWN_BROKER	3085	X'00000C0D'
CHYBA MQRCCF_Q_MGR_CCSID_ERROR	3086	X'00000C0E'
CHYBA MQRCCF_DEL_OPTIONS_ERROR	3087	X'00000C0F'
KONFLIKT MQRCCF_CLUSTER_NAME_CONFLICT	3088	X'00000C10'
KONFLIKT MQRCCF_REPOSNAME_CONFLICT	3089	X'00000C11'
CHYBA MQRCCF_CLUSTER_Q_USAGE_ERROR	3090	X'00000C12'
CHYBA MQRCCF_ACTION_VALUE_ERROR	3091	X'00000C13'
CHYBA MQRCCF_COMMS_LIBRARY_ERROR	3092	X'00000C14'
CHYBA MQRCCF_NETBIOS_NAME_ERROR	3093	X'00000C15'
PŘÍKAZ MQRCCF_BROKER_COMMAND_FAILED	3094	X'00000C16'
MQRCCF_CFST_CONFLICTING_PARM	3095	X'00000C17'
MQRCCF_PATH_NOT_VALID	3096	X'00000C18'
CHYBA MQRCCF_PARM_SYNTAX_ERROR	3097	X'00000C19'
CHYBA_DÉLKA_VYPRŠENÍ_MQRCCF_PWD	3098	X'00000C1A'
CHYBA MQRCCF_FILTER_ERROR	3150	X'00000C4E'
UŽIVATEL MQRCCF_WRONG_USER	3151	X'00000C4F'
DUPLICITNÍ_ODBĚR MQRCCF_DUPLICATION	3152	X'00000C50'
CHYBA MQRCCF_SUB_NAME_ERROR	3153	X'00000C51'
CHYBA OBJEKTU 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'
MQRCCF_OBJECT_IN_USE	3160	X'00000C58'
SOUBOR MQRCCF_UNKNOWN_FILE_NAME	3161	X'00000C59'
MQRCCF_FILE_NOT_AVAILABLE	3162	X'00000C5A'
CHYBA MQRCCF_DISC_RETRY_ERROR	3163	X'00000C5B'
CHYBA MQRCCF_ALLOC_RETRY_ERROR	3164	X'00000C5C'
MQRCCF_ALLOC_SLOW_TIMER_ERROR	3165	X'00000C5D'
CHYBA OBJEKTU MQRCCF_ALLOCATION_FAST_TIMER_ERROR	3166	X'00000C5E'
MQRCCF_PORT_NUMBER_ERROR	3167	X'00000C5F'
MQRCCF_CHL_SYSTEM_NOT_ACTIVE	3168	X'00000C60'
CHYBÍ POLOŽKA MQRCCF_ENTITY_NAME_	3169	X'00000C61'
CHYBA MQRCCF_PROFILE_NAME_ERROR	3170	X'00000C62'
CHYBA MQRCCF_AUTH_VALUE_ERROR	3171	X'00000C63'
CHYBÍ HODNOTA MQRCCF_AUTH_VALUE_MISSING	3172	X'00000C64'
CHYBÍ MQRCCF_OBJECT_TYPE_	3173	X'00000C65'
CHYBA OBJEKTU MQRCCF_CONNECTION_ID_	3174	X'00000C66'
CHYBA MQRCCF_LOG_TYPE_ERROR	3175	X'00000C67'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCCF_PROGRAM_NOT_AVAILABLE	3176	X'00000C68'
MQRCCF_PROGRAM_AUTH_FAILED	3177	X'00000C69'
SOUBOR MQRCCF_NON_FOUND	3200	X'00000C80'
MQRCCF_SECURITY_SWITCH_OFF	3201	X'00000C81'
SOUBOR MQRCCF_SECURITY_REFRESH_FAILED	3202	X'00000C82'
KONFLIKT MQRCCF_PARM_CONFLICT	3203	X'00000C83'
MQRCCF_COMMAND_BLOKOVÁNO	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_EXCEED	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'
CHYBA MQRCCF_NAMELIST_ERROR	3215	X'00000C8F'
INICIALIZÁTOR MQRCCF_NO_CHANNEL_INITIATOR	3217	X'00000C91'
CHYBA DÉMONA MQRCCF_CHANNEL_INITIATOR_ERROR	3218	X'00000C92'
KONFLIKT PŘÍKAZOVÉHO_ÚROVNĚ_MQRCCF_COMMAND_LEVEL	3222	X'00000C96'
KONFLIKT MQRCCF_Q_ATTR_CONFLICT	3223	X'00000C97'
MQRCCF_EVENTS_DISABLED	3224	X'00000C98'
CHYBA OBJEKTU MQRCCF_COMMAND_SCOPE_ERROR	3225	X'00000C99'
CHYBA PŘI ODPOVĚDI MQRCCF_COMMAND_REPLY_ERROR	3226	X'00000C9A'
FUNKCE MQRCCF_FUNCTION_RESTRICTED	3227	X'00000C9B'
CHYBÍ MQRCCF_PARM_MISSING	3228	X'00000C9C'
CHYBA MQRCCF_PARM_VALUE_ERROR	3229	X'00000C9D'
MQRCCF_COMMAND_LENGTH_ERROR	3230	X'00000C9E'
MQRCCF_COMMAND_ORIGIN_ERROR	3231	X'00000C9F'
KONFLIKT MQRCCF_LISTENER_CONFLICT	3232	X'00000CA0'
MQRCCF_LISTENER_STARTED	3233	X'00000CA1'
MQRCCF_LISTENER_STOPPED	3234	X'00000CA2'
CHYBA MQRCCF_CHANNEL_ERROR	3235	X'00000CA3'
CHYBA OBJEKTU MQRCCF_CF_STRUC_	3236	X'00000CA4'
MQRCCF_UNKNOWN_USER_ID	3237	X'00000CA5'
CHYBA MQRCCF_UNEXPECTED_ERROR	3238	X'00000CA6'
MQRCCF_NO_XCF_PARTNER	3239	X'00000CA7'
CHYBA MQRCCF_CFGR_PARM_ID_ERROR	3240	X'00000CA8'
MQRCCF_CFIF_LENGTH_ERROR	3241	X'00000CA9'
CHYBA MQRCCF_CFIFIC_OPERATOR_ERROR	3242	X'00000CAA'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBA MQRCCF_CFIF_PARM_ID_ERROR	3243	X'00000CAB'
MQRCCF_CFSF_FILTER_VAL_LEN_ERR	3244	X'00000CAC'
CHYBA MQRCCF_CFSF_LENGTH_ERROR	3245	X'00000CAD'
CHYBA MQRCCF_CFSF_OPERATOR_ERROR	3246	X'00000CAE'
CHYBA 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'
SLUŽBA MQRCCF_SERVICE_RUNNING	3251	X'00000CB3'
MQRCCF_SERV_STATUS_NOT_FOUND	3252	X'00000CB4'
SLUŽBA MQRCCF_SERVICE_STOPPED	3253	X'00000CB5'
MQRCCF_CFBS_DUPLICATE_PARM	3254	X'00000CB6'
MQRCCF_CFBS_LENGTH_ERROR	3255	X'00000CB7'
CHYBA MQRCCF_CFBS_PARM_ID_ERROR	3256	X'00000CB8'
MQRCCF_CFBS_STRH_LENGTH_ERR	3257	X'00000CB9'
CHYBA MQRCCF_CFGR_LENGTH_ERROR	3258	X'00000CBA'
MQRCCF_CFGR_PARM_COUNT_ERROR	3259	X'00000CBB'
MQRCCF_CONN_NOT_STOPPED	3260	X'00000CBC'
NEVYŘÍZENÝ MQRCCF_SERVICE_REQUEST_PENDING	3261	X'00000CBD'
MQRCCF_NO_START_CMD	3262	X'00000CBE'
MQRCCF_NO_STOP_CMD	3263	X'00000CBF'
CHYBA MQRCCF_CFBF_LENGTH_ERROR	3264	X'00000CC0'
CHYBA MQRCCF_CFBF_PARM_ID_ERROR	3265	X'00000CC1'
CHYBA MQRCCF_CFBF_OPERATOR_ERROR	3266	X'00000CC2'
MQRCCF_CFBF_FILTER_VAL_LEN_ERR	3267	X'00000CC3'
MQRCCF_LISTENER_STILL_ACTIVE	3268	X'00000CC4'
CHYBA MQRCCF_DEF_XMIT_Q_CLUS_ERROR	3269	X'0000CC5'
MQRCCF_TOPICSTR_ALREADY_EXISTS	3300	X'00000CE4'
CHYBA SOUBORU MQRCCF_SHARING_CONVS_ERROR	3301	X'00000CE5'
TYP SOUBORU MQRCCF_SHARING_CONVS_TYPE	3302	X'00000CE6'
KONFLIKT MQRCCF_SECURITY_CASE_CONFLICT	3303	X'00000CE7'
CHYBA MQRCCF_TOPIC_TYPE_ERROR	3305	X'00000CE9'
CHYBA MQRCCF_MAX_INSTANCES_ERROR	3306	X'00000CEA'
MQRCCF_MAX_INST_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'
CHYBA MQRCCF_REMOTE_Q_NAME_ERROR	3313	X'00000CF1'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCCF_DURABILITY_NOT_ALLOWED	3314	X'00000CF2'
CHYBA MQRCCF_HOBJ_ERROR	3315	X'00000CF3'
CHYBA MQRCCF_DEST_NAME_ERROR	3316	X'00000CF4'
MQRCCF_NEPLATNÉ_MÍSTO URČENÍ	3317	X'00000CF5'
MQRCCF_PUBSUB_BLOKOVÁNO	3318	X'00000CF6'
CHYBA MQRCCF_CHLAUTH_TYPE_ERROR	3326	X'00000CFE'
MQRCCF_CHLAUTH_ACTION_ERROR	3327	X'00000CFF'
CHYBA 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'
AKCE MQRCCF_WRONG_CHLAUTH_ACTION	3339	X'00000D0B'
MQRCCF_WRONG_CHLAUTH_USERSRC	3340	X'00000D0C'
CHYBA MQRCCF_CHLUTH_WARN_ERROR	3341	X'00000D0D'
MQRCCF_WRONG_CHLAUTH_MATCH	3342	X'00000D0E'
KONFLIKT ROZSAHU MQRCCF_IPADDR_RANGE_	3343	X'00000D0F'
MQRCCF_CHLAUTH_MAX_EXCEEDED	3344	X'00000D10'
CHYBA MQRCCF_IPADDR_ERROR	3345	X'00000D11'
CHYBA MQRCCF_IPADDR_RANGE_ERROR	3346	X'00000D12'
CHYBÍ POLOŽKA MQRCCF_PROFILE_NAME_MISSING	3347	X'00000D13'
CHYBA MQRCCF_CHLAUTH_CLNTUSER_ERROR	3348	X'00000D14'
CHYBA MQRCCF_CHLAUTH_NAME_ERROR	3349	X'00000D15'
CHYBA MQRCCF_SUITE_B_CHYB	3353	X'00000D19'
MQRCCF_PSCLUS_DISABLE_TOPDEF	3359	X'00000D1F'
MQRCCF_PSCLUS_TOPIC_EXISTS	3360	X'00000D20'
PROTOKOL NEPLATNÉHO_PROTOKOLU MQRCCF_INVALID	3365	X'00000D25'
MQRCCF_ACCESS_BLOCKED	3382	X'00000D36'
MQRCCF_OBJECT_ALREADY_EXISTS	4001	X'00000FA1'
MQRCCF_OBJECT_WRONG_TYPE	4002	X'00000FA2'
MQRCCF_LIKE_OBJECT_WRONG_TYPE	4003	X'00000FA3'
MQRCCF_OBJECT_OPEN	4004	X'00000FA4'
CHYBA MQRCCF_ATTR_VALUE_ERROR	4005	X'00000FA5'
NÁZEV SPRÁVCE FRONT MQRCCF_UNKNOWN_Q_MGR	4006	X'00000FA6'
MQRCCF_Q_NEOPRÁVNĚNÝ_TYP	4007	X'00000FA7'
CHYBA OBJEKTU MQRCCF_OBJECT_NAME_ERROR	4008	X'00000FA8'
SELHÁNÍ PŘÍKAZU MQRCCF_ALLOCATE_FAILED	4009	X'00000FA9'
MQRCCF_HOST_NOT_AVAILABLE	4010	X'00000FAA'
CHYBA KONFIGURACE MQRCCF_CONFIGURATION_	4011	X'00000FAB'
MQRCCF_CONNECTION_REFUSED	4012	X'00000FAC'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
CHYBOVÁ CHYBA MQRCCF_ERROR	4013	X'00000FAD'
MQRCCF_SEND_FAILED	4014	X'00000FAE'
CHYBA OBJEKTU MQRCCF_RECEIVED_DATA_ERROR	4015	X'00000FAF'
NEZDAŘILO SE: 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'
SELHÁNÍ MQRCCF_BIND_FAILED	4024	X'00000FB8'
MQRCCF_CHANNEL_NEOVĚŘENÝ	4025	X'00000FB9'
MQRCCF_MQCONN_FAILED	4026	X'00000FBA'
MQRCCF_MQOPEN_FAILED	4027	X'00000FBB'
MQRCCF_MQGET_FAILED	4028	X'00000FBC'
SOUBOR MQRCCF_MQPUT_FAILED	4029	X'00000FBD'
CHYBA 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'
MQRCCF_USER_EXIT_NOT_AVAILABLE	4039	X'00000FC7'
MQRCCF_COMMIT_FAILED	4040	X'00000FC8'
TYP_KANÁLU MQRCCF_WRONG_LAW_TYPE	4041	X'00000FC9'
MQRCCF_CHANNEL_ALREADY_EXISTS	4042	X'00000FCA'
MQRCCF_DATA_TOO_LARGE	4043	X'00000FCB'
CHYBA MQRCCF_CHANNEL_NAME_ERROR	4044	X'00000FCC'
CHYBA MQRCCF_XMIT_Q_NAME_ERROR	4045	X'00000FCD'
CHYBA MQRCCF_MCA_NAME_ERROR	4047	X'00000FCF'
CHYBA MQRCCF_SEND_EXIT_NAME_ERROR	4048	X'00000FD0'
CHYBA MQRCCF_SEC_EXIT_NAME_ERROR	4049	X'00000FD1'
CHYBA MQRCCF_MSG_EXIT_NAME_ERROR	4050	X'00000FD2'
CHYBA MQRCCF_RCV_EXIT_NAME_ERROR	4051	X'00000FD3'
MQRCCF_XMIT_QNAME_CHYBNÝ_TYP	4052	X'00000FD4'
MQRCCF_MCANAME_NEOPRÁVNĚNÝ TYP	4053	X'00000FD5'
MQRCCF_DISC_INT_INQUIL_TYPE	4054	X'00000FD6'
MQRCCF_SHORT_RETRY_ANTER_TYPE	4055	X'00000FD7'

Tabulka 307. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCCF_SHORT_TIMER_QUANNER_TYPE	4056	X'00000FD8'
MQRCCF_LONG_RETRY_NEOPRÁVNĚNÝ_TYP	4057	X'00000FD9'
MQRCCF_LONG _TIMER_QUANGI_TYPE	4058	X'00000FDA'
MQRCCF_PUT_AUTH_NEOPRÁVNĚNÝ TYP	4059	X'00000FDB'
MQRCCF_KEEP_ALIVE_INT_ERROR	4060	X'00000FDC'
MQRCCF_MISSING_CONN_NAME	4061	X'00000FDD'
CHYBA MQRCCF_CONN_NAME_ERROR	4062	X'00000FDE'
NEZDAŘILO SE: MQRCCF_MQSET_FAILED	4063	X'00000FDF'
MQRCCF_CHANNEL_NOT_ACTIVE	4064	X'00000FE0'
MQRCCF_TERMINATED_BY_SEC_EXIT	4065	X'00000FE1'
CHYBA OBJEKTU MQRCCF_DYNAMIC_Q_SCOPE_ERROR	4067	X'00000FE3'
MQRCCF_CELL_DIR_NOT_AVAILABLE	4068	X'00000FE4'
POČET CHYB: MQRCCF_MR_COUNT_ERROR	4069	X'00000FE5'
CHYBNÝ_TYP_MEZIPAMĚTI MQRCCF_MR	4070	X'00000FE6'
CHYBA MQRCCF_MR_EXIT_NAME_ERROR	4071	X'00000FE7'
MQRCCF_MR_EXIT_NAME_CHYBNÝ_TYP	4072	X'00000FE8'
CHYBA_INTERVAL_MR MQRCCF_MR_	4073	X'00000FE9'
MQRCCF_MR_INTERVAL_NEOPRÁVNĚNÝ_TYP	4074	X'00000FEA'
CHYBA MQRCCF_NPM_SPEED_ERROR	4075	X'00000FEB'
MQRCCF_NPM_SPÉM_CHYBNÝ_TYP	4076	X'00000FEC'
MQRCCF_HB_INTERVAL_ERROR	4077	X'00000FED'
MQRCCF_HB_INTERVAL_NEOPRÁVNĚNÝ_TYP	4078	X'00000FEE'
CHYBA MQRCCF_CHAD_ERROR	4079	X'00000FEF'
MQRCCF_CHAD_NEOPRÁVNĚNÝ_TYP	4080	X'00000FF0'
CHYBA OBJEKTU MQRCCF_CHAD_EVENT_ERROR	4081	X'00000FF1'
MQRCCF_CHAD_EVENT_NEOPRÁVNĚNÝ TYP	4082	X'00000FF2'
CHYBA MQRCCF_CHAD_EXIT_ERROR	4083	X'00000FF3'
MQRCCF_CHAD_EXIT_NEOPRÁVNĚNÝ_TYP	4084	X'00000FF4'
MQRCCF_SUPPRESSED_BY_EXIT	4085	X'00000FF5'
CHYBA MQRCCF_BATCH_INT_ERROR	4086	X'00000FF6'
MQRCCF_BATCH_INT_ANQUIL_TYPE	4087	X'00000FF7'
CHYBA MQRCCF_NET_PRIORITY_ERROR	4088	X'00000FF8'
MQRCCF_NET_PRIORITY_NEOPRÁVNĚNÝ TYP	4089	X'00000FF9'
MQRCCF_CHANNEL_CLOSED	4090	X'00000FFA'
MQRCCF_Q_STATUS_NOT_FOUND	4091	X'00000FFB'
CHYBA_ŠIFRU_SSL MQRCCF_SSL_ŠIFR	4092	X'00000FFC'
CHYBA MQRCCF_SSL_PEER_NAME_ERROR	4093	X'00000FFD'
MQRCCF_SSL_CLIENT_AUTH_ERROR	4094	X'00000FFE'
MQRCCF_RETAINED_NOT_SUPPORTED	4095	X'00000FFF'

Tabulka 307. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
z/OS V 9.2.0 MQRCCF_KWD_VALUE_WRONG_TYPE	4096	X'00001000'

MQRCN_ * (Konstanty opětovného připojení klienta)

Tabulka 308. 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 časových limitů příjmu)

Tabulka 309. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRCVTIME_MULTIPLY	0	X'00000000'
MQRCVTIME_ADD	1	X'00000001'
MQRCVTIME_EQUAL	2	X'00000002'

MQREADA_ * (hodnoty dopředného čtení)

Tabulka 310. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQREADA_NO	0	X'00000000'
MQREADA_YES	1	X'00000001'
MQREADA_DISABLED	2	X'00000002'
MQREADA_BLOKOVÁNO	3	X'00000003'
NEVYŘÍZENÉ POŽADAVKY MQREADA_	4	X'00000004'

MQRECORDING_ * (Volby záznamu)

Tabulka 311. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRECORDING_DISABLED	0	X'00000000'
MQRECORDING_Q	1	X'00000001'
ZPRÁVA MQRECORDING_MSG	2	X'00000002'

MQREGAR_ * (Volby registrace publikování/odběru)

Tabulka 312. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQREGO_NONE	0	X'00000000'
MQREGO_CORRELA_ID_AS_IDENTITY	1	X'00000001'
MQREGO_ANONYMOUS	2	X'00000002'

Tabulka 312. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQREGION_LOCAL	4	X'00000004'
MQREGO_DIRECT_REQUESTS	8	X'00000008'
POUZE NOVÉ_VEŘEJNÉ_PUBLIKACE_MQREGO_	16	X'00000010'
POUZE MQREGO_PUBLISH_ON_REQUEST_ONLY	32	X'00000020'
MQREGO_DEREGISTER_ALL	64	X'00000040'
MQREGO_INCLUDE_STREAM_NAME	128	X'00000080'
MQREGO_INFORCE_IF_RETAINED	256	X'00000100'
MQREGO_DUPLICATES_OK	512	X'00000200'
MQREGO_NON_PERSISTENT	1024	X'00000400'
MQREGO_PERSISTENT	2048	X'00000800'
MQREGO_PERSISTENT_AS_PUBLISH	4096	X'00001000'
MQREGO_PERSISTENT_AS_Q	8192	X'00002000'
NÁZEV OBJEKTU MQREGO_ADD_NAME	16384	X'00004000'
ÚPRAVA MQREGO_NO_ALTERING	32768	X'00008000'
MQREGO_FULL_RESPONSE	65536	X'00010000'
MQREGO_JOIN_SHARED	131072	X'00020000'
MQREGO_JOIN_EXCLUSIVE	262144	X'00040000'
POUZE MQREGO_LEAVE_ONLY	524288	X'00080000'
MQREGO_VARIABLE_USER_ID	1048576	X'00100000'
MQREGO_LOCKED	2097152	X'00200000'

MQRFH_ * (Pravidla a formátování struktury a příznaků záhlaví)

Pravidla a formátování struktury záhlaví

Tabulka 313. Konstrukce konstant	
Název	**Struktura**
MQRFH_STRUCT	"RFH¬"
MQRFH_STRUC_ID_POLE	'R','F','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 314. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH_VERSION_1	1	X'00000001'
MQRFH_VERSION_2	2	X'00000002'
PEVNÉ PRODLOUŽENÍ MQRFH_STRUCLOTH_FIXED	32	X'00000020'
MQRFH_STRUC_LENGTH_FIXED_2	36	X'00000024'

Pravidla a formátovací parametry záhlaví

Tabulka 315. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH_NONE	0	X'00000000'
PŘÍZNAKY MQRFH_NO_FLAGS	0	X'00000000'

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

Tabulka 316. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

MQRFH2_* (Názvy značek publikování a odběru značek)

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

MQRFH2_* (Názvy značek publikování a odběru značek XML)

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 317. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRL_UNDEFINED	-1	X'FFFFFFFF'

MQRMH_ * (Struktura záhlaví referenční zprávy)

Tabulka 318. Konstrukce konstant	
Název	**Struktura**
MQRMH_STRUCTION_ID	"RMH¬"
MQRMH_STRUC_ID_ARRAY	'R','M','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 319. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRMH_VERSION_1	1	X'00000001'
MQRMH_AKTUÁLNÍ_VERZE	1	X'00000001'

MQRMHF_ * (Referenční parametry záhlaví zprávy)

Tabulka 320. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

MQRO_ * (Volby sestav)

Tabulka 321. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝJIMKA MQRO_EXCEPTION	16777216	X'01000000'
MQRO_EXCEPTION_WIT_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_CED_WITH_DATA	6144	X'00001800'
MQRO_COD_WITH_FULL_DATA	14336	X'00003800'
MQRO_PAN	1	X'00000001'
MQRO_NAN	2	X'00000002'
AKTIVITA 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'
ID_KOLEKCE_MQRO_PASS_RELACE_	64	X'00000040'
MQRO_DEAD_LETTER_Q	0	X'00000000'
MQRO_DISCARD_MSG	134217728	X'08000000'
MQRO_PASS_DISCARD_AND_EXPIRY	16384	X'00004000'
MQRO_NONE	0	X'00000000'

*MQRO_ * (Masky z voleb sestavy)*

Tabulka 322. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRO_REJECT_UNSUP_MASK	270270464	X'101C0000'

Tabulka 322. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRO_ACCEPT_UNSUP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUP_IF_XMIT_MASK	261888	X'0003FF00'

MQROUTE_ * (trasová-trasa)

Maximální počet aktivit trasování cesty (MQIACF_MAX_ACTIVITIES)

Tabulka 323. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_UNLIMITED_ACTIVITIES	0	X'00000000'

Podrobnosti trasy trasování (MQIACF_ROUTE_DETAIL)

Tabulka 324. 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'

Postoupení přenosové cesty (MQIACF_ROUTE_FORWARDING)

Tabulka 325. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_FORWARD_ALL	256	X'00000100'
PODPOROVANÁ MQROUTE_FORWARD_IF_SUPPORTED	512	X'00000200'
MQROUTE_FORWARD_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Doručení trasování cesty (MQIACF_ROUTE_DELIVERY)

Tabulka 326. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_DELIVER_YES	4096	X'00001000'
MQROUTE_DELIVER_NO	8192	X'00002000'
MQROUTE_DELIVER_REJ_UNSUP_MASK	-65536	X'FFFF0000'

Kumulace trasování cesty (MQIACF_ROUTE_ACCUMULATION)

Tabulka 327. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQROUTE_ACCUMULATE_NONE	65539	X'00010003'
MQROUTE_ACCUMULATE_IN_MSG	65540	X'00010004'
MQROUTE_ACCUMULATE_AND_REPLY	65541	X'00010005'

MQRP_ * (Volby nahrazení formátu příkazu)

Tabulka 328. Hodnoty konstant

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

MQRQ_ * (Kvalifikátory důvodu formátu příkazu)

Tabulka 329. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
AUTORIZOVANÝ MQRQ_CONN_NOT_AUTHORIZED	1	X'00000001'
MQRQ_OPEN_NOT_AUTHORIZED	2	X'00000002'
MQRQ_CLOSE_NOT_AUTHORIZED	3	X'00000003'
AUTORIZOVANÝ OBJEKT MQRQ_CMD_NOT_AUTHORIZED	4	X'00000004'
ZASTAVOVÁNÍ MQRQ_Q_MGR_STOPPING	5	X'00000005'
UVÁDĚNÍ 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'
CHYBA DÉMONA MQRQ_BRIDGE_STOPPED_ERROR	12	X'0000000C'
CHYBA MQRQ_SSL_HANDSHAKE_ERROR	13	X'0000000D'
CHYBA 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_AUTHORIZED	17	X'00000011'
MQRQ_SUB_DEST_NOT_AUTHORIZED	18	X'00000012'
MQRQ_SSL_UNKNOWN_REVOCATION	19	X'00000013'
AUTORIZOVANÝ OBJEKT MQRQ_SYS_CONN_NOT_AUTHORIZED	20	X'00000014'
ADRESA_KANÁLŮ_MQRQ_CHANNEL_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'
AUTORIZOVANÝ MQRQ_CSP_NOT_AUTHORIZED	29	X'0000001D'
POVOLENÉ HODNOTY MQRQ_FAILOVER_PERMITTED	30	X'0000001E'
POVOLENÁ HODNOTA MQRQ_FAILOVER_NOT_PERMITTED	31	X'0000001F'
MQRQ_STANDBY_ACTIVATED	32	X'00000020'

Tabulka 329. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRQ_REPLICA_ACTIVATED	33	X'00000021'

MQRT_ * (Typy obnovení příkazového formátu)

Tabulka 330. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KONFIGURACE MQRT_CONFIGURATION	1	X'00000001'
MQRT_EXPIRY	2	X'00000002'
MQRT_NSPROC	3	X'00000003'
MQRT_PROXYSUB	4	X'00000004'

MQRU_ * (pouze požadavek)

Tabulka 331. 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 332. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POŽADOVÁNO MQSCA_REQUIRED	0	X'00000000'
MQSCA_OPTIONAL	1	X'00000001'

MQSCO_ * (volby konfigurace TLS)

Struktura voleb konfigurace TLS

Tabulka 333. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQSCO_	"SCO¬"
POLE MQSCO_STRUC_ID_ARRAY	'S','C','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 334. 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'
AKTUÁLNÍ_VERZE MQSCO_	4	X'00000004'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Počet resetů klíčů voleb konfigurace TLS

Tabulka 335. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ HODNOTA MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

Rozsah definice definice fronty příkazů

Tabulka 336. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCOM_Q_MGR	1	X'00000001'
BUŇKA MQSCO_CELL	2	X'00000002'

MQSCOPE_ * (obor publikování)

Tabulka 337. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCOPE_ALL	0	X'00000000'
MQSCOPE_AS_PARENT	1	X'00000001'
MQSCOPE_QMGR	4	X'00000004'

MQSCYC_ * (Bezpečnostní případ)

Tabulka 338. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCYC_UPPER	0	X'00000000'
MQSCYC_SMÍŠENÝ	1	X'00000001'

MQSD_ * (struktura deskriptoru objektu)

Tabulka 339. Konstantní názvy a struktury	
Název	**Struktura**
ID_STRUKTURY OBJEKTU MQSD_STRUCT	"SD¬¬"
POLE MQSD_STRUC_ID_ARRAY	'S','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 340. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSD_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQSD_AKTUÁLNÍ_VERZE	1	X'00000001'

MQSECITEM_ * (Položky zabezpečení ve formátu příkazu)

Tabulka 341. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECITEM_VŠE	0	X'00000000'
MQSECITEM_MQADMIN	1	X'00000001'
TABULKA MQSECITEM_MQNLIST	2	X'00000002'

Tabulka 341. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PROCEDURA MQSECITEM_MQPROC	3	X'00000003'
FRONTA 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'
FRONTA MQSECITEM_MXQUEUE	10	X'0000000A'
MQSECITEM_MXTOPIC	11	X'0000000B'

MQSECPROT_ * (Typy protokolů zabezpečení)

Tabulka 342. 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_ * (Přepínače zabezpečení a přepínače příkazů ve formátu příkazu)

Přepínače zabezpečení ve formátu příkazu

Tabulka 343. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PROCES MQSECSW_PROCESS	1	X'00000001'
SEZNAM NÁZVŮ MQSECSW_NAMELIST	2	X'00000002'
MQSECSW_Q	3	X'00000003'
TÉMA MQSECSW_TOPIC	4	X'00000004'
KONTEXT MQSECSW_CONTEXT	6	X'00000006'
UŽIVATEL MQSECSW_ALTERNATE_USER	7	X'00000007'
PŘÍKAZ MQSECSW_COMMAND	8	X'00000008'
PŘIPOJENÍ MQSECSW_CONNECTION	9	X'00000009'
SUBSYSTÉM MQSECSW_SUBSYSTEM	10	X'0000000A'
PROSTŘEDKY PŘÍKAZU MQSECSW_COMMAND_RESOURCES	11	X'0000000B'
MQSECSW_Q_MGR	15	X'0000000F'
MQSECSW_QSG	16	X'00000010'

Stavy přepínačů zabezpečení formátu příkazu

Tabulka 344. 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'
CHYBA MQSECSW_OFF_ERROR	25	X'00000019'
MQSECSW_ON_OVERRIDDEN	26	X'0000001A'

MQSECTYPE_ * (Typy zabezpečení formátu příkazu)

Tabulka 345. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECTYPE_AUTHSERV	1	X'00000001'
MQSEC_SSL	2	X'00000002'
TŘÍDY MQSECTYPE_CLASSES	3	X'00000003'

MQSEG_ * (Segmentace)

Tabulka 346. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQSEG_BLOKOVÁNO	'¬'
MQSEG_ALLOWED	'A'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQSEL_ * (speciální hodnoty selektoru)

Tabulka 347. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SELEKTOR MQSEL_ANY_SELECTOR	-30001	X'FFFF8ACF'
SELEKTOR MQSEL_ANY_USER_SELECTOR	-30002	X'FFFF8ACE'
SELEKTOR MQSEL_ANY_SYSTEM_SELECTOR	-30003	X'FFFF8ACD'
MQSEL_ALL_SELECTORS	-30001	X'FFFF8ACF'
SELEKTORY MQSEL_ALL_USER_SELEKT	-30002	X'FFFF8ACE'
MQSEL_ALL_SYSTEM_SELECTORS	-30003	X'FFFF8ACD'

MQSELTYPE_ * (Typy selektoru)

Tabulka 348. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSELTYPE_NONE	0	X'00000000'
MQSELTYPE_STANDARD	1	X'00000001'
MQSELTYPE_EXTENDED.	2	X'00000002'

MQSID_ * (Identifikátor zabezpečení)

Tabulka 349. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQSID_NONE	X'00...00' (40 nul)
MQSID_NON_ARRAY	'\0','\0',... (40 nul)

MQSIDT_ * (typy identifikátoru zabezpečení)

Tabulka 350. Konstantní názvy a hodnoty	
Název	**Hexadecimální hodnota**
MQSIDT_NONE	X'00'
ID_BEZPEČNOSTNÍHO_ZABEZPEČENÍ MQSID_NT_ID_	X'01'
ID_ADMINISTRÁTORA_MQSID_WAS	X'02'

MQSMPO_ * (Nastavení vlastností a struktury vlastností zprávy)

Nastavit strukturu voleb vlastností zprávy

Tabulka 351. Konstrukce konstant	
Název	**Struktura**
MQSMPO_STRUCTION_ID	"SMPO"
POLE MQSMPO_STRUC_ID_ARRAY	'S','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 352. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSMPO_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQSMPO_CURRENT_VERSION	1	X'00000001'

Nastavit volby vlastností zprávy

Tabulka 353. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE LQSMPO_SET_FIRST	0	X'00000000'
MQSMPO_SET_PROP_UNDER_CURSOR	1	X'00000001'
MQSMPO_SET_PROP_AFTER_CURSOR	2	X'00000002'
VLASTNOST MQSMPO_APPEND_PROPERTY	4	X'00000004'
FUNKCE MQSMPO_SET_PROP_BEFORE_CURSOR	8	X'00000008'
MQSMPO_NONE	0	X'00000000'

MQSO_ * (Volby odběru)

Tabulka 354. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSO_NONE	0	X'00000000'
MQSO_NON_DURABLE	0	X'00000000'

Tabulka 354. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
FUNKCE MQSO_READ_AHEAD_AS_Q_DEF	0	X'00000000'
MQSO_ALTER	1	X'00000001'
VYTVOŘENÉ MQSO_CREATE	2	X'00000002'
MQSO_RESUME	4	X'00000004'
MQSO_TRVALKA	8	X'00000008'
MQSO_GROUP_SUB	16	X'00000010'
SPRAVOVANÉ MQSO_MANAGED	32	X'00000020'
KONTEXT MQSO_SET_IDENTITY_CONTEXT	64	X'00000040'
ID UŽIVATELE MQSO_FIXED_USERID	256	X'00000100'
MQSO_ANY_USERID	512	X'00000200'
POŽADAVEK MQSO_PUBLICATIONS_ON_REQUEST	2048	X'00000800'
POUZE NOVÉ_VEŘEJNÉ_VEŘEJNÉ_PUBLIKOVÁNÍ	4096	X'00001000'
MQSO_FAIL_IF_QUIESCING	8192	X'00002000'
OPRÁVNĚNÍ UŽIVATELE MQSO_ALTERNATE_USER_AUTHORITY	262144	X'00040000'
MQSO_WILDCARD_CHAR	1048576	X'00100000'
TÉMA MQSO_WILDCARD_TOPIC	2097152	X'00200000'
ID_SADY_MQSO_SET_CORRELACE_	4194304	X'00400000'
MQSO_SCOPE_QMGR	67108864	X'04000000'
MQSO_NO_READ_AHEAD	134217728	X'08000000'
MQSO_READ_AHEAD	268435456	X'10000000'

MQSP_ * (Dostupnost bodu synchronizace)

Tabulka 355. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSP_AVAILABLE	1	X'00000001'
MQSP_NOT_AVAILABLE	0	X'00000000'

V 9.2.0 MQSPL_ * (Volby ochrany zásad zabezpečení)

Tabulka 356. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSPL_PASSTHRU	0	X'00000000'
MQSPL_REMOVE	1	X'00000001'
ZÁSADA MQSPL_AS_POLICY	2	X'00000002'

MQSQQM_ * (Název správce front sdílené fronty)

Tabulka 357. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

MQSR_ * (Akce)

Tabulka 358. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSR_ACTION_PUBLICATION	1	X'00000001'

MQSRO_ * (Struktura voleb požadavku na odběr)

Tabulka 359. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQSRO_STRUCTION_ID	"SRO¬"
POLE MQSRO_STRUC_ID_ARRAY	'S','R','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 360. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSRO_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQSRO_CURRENT_VERSION	1	X'00000001'
MQSRO_NONE	0	X'00000000'
MQSRO_FAIL_IF_QUIESCING	8192	X'00002000'

MQSS_ * (Stav segmentu)

Tabulka 361. Konstantní názvy a struktury	
Název	**Struktura**
SEGMENT MQSS_NOT_SEGMENT	'¬'
SEGMENT 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" . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C certificate a měli by si být vědomi jakýchkoli 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 362. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSSL_FIPS_NO	0	X'00000000'
MQSSL_FIPS_YES	1	X'00000001'

MQSTAT_ * (volby statistiky)

Tabulka 363. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
CHYBA MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'

Tabulka 363. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION	0	X'00000000'
CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

MQSTS_ * (Struktura struktury tvorby sestav stavu)

Tabulka 364. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQSTS_	"STAT"
POLE MQSTS_STRUC_STRUC_ID_ARRAY	'S','T','A','T'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 365. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSTS_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQSTS_CURRENT_VERSION	1	X'00000001'

MQSUB_ * (trvalé odběry)

Odběry povolených odběrů

Tabulka 366. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUB_DURABLE_AS_PARENT	0	X'00000000'
MQSUB_DURABLE_ALLOWED	1	X'00000001'
MQSUB_DURABLE_BLOKOVÁNO	2	X'00000002'

Rozsah trvalých odběrů

Tabulka 367. 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 v typech odběrů)

Tabulka 368. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUBTYPY_API	1	X'00000001'
MQSUBTYPE_ADMIN	2	X'00000002'
MQSUBTYPY_PROXY	3	X'00000003'
MQSUBTYPY_VŠE	-1	X'FFFFFFFF'
UŽIVATEL MQSUBTYPE_USER	-2	X'FFFFFFFE'

MQSUS_ * (Stav pozastavení ve formátu příkazu)

Tabulka 369. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUS_YES	1	X'00000001'
MQSUS_NO	0	X'00000000'

MQSVC_ * (Služba)

Typy služeb

Tabulka 370. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSVC_TYPE_COMMAND, PŘÍKAZ	0	X'00000000'
SERVER_SPRÁVY MQSVC_TYPE_SERVER	1	X'00000001'

Ovládací prvky služeb

Tabulka 371. 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 372. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
STAV MQSVC_STATUS_STOPPED	0	X'00000000'
STAV MQSVC_STATUS_STARTING	1	X'00000001'
STAV MQSVC_STATUS_RUNNING	2	X'00000002'
STAV_STAV_MQSVC	3	X'00000003'
STAV MQSVC_STATUS_RETRYING	4	X'00000004'

MQSYNCPMINT_ * (Hodnoty synchronizačního bodu ve formátu příkazu pro migraci publikování/odběru)

Tabulka 373. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYNCPOINT_YES	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

MQSYSP_ * (Systémové hodnoty parametrů systému)

Tabulka 374. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYSP_NO	0	X'00000000'

Tabulka 374. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYSP_YES	1	X'00000001'
MQSYSP_EXTENDED.	2	X'00000002'
VÝCHOZÍ HODNOTA MQSYSP_TYPE_INITIAL	10	X'0000000A'
MQSYSP_TYPE_SET	11	X'0000000B'
MQSYSP_TYPE_LOG_COPY	12	X'0000000C'
STAV PROTOKOLU 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_ALLOCA_ARCHIVE	34	X'00000022'
MQSYSP_STATUS_COPYING_BSDS	35	X'00000023'
PROTOKOL MQSYSP_STATUS_COPYING_LOG	36	X'00000024'

MQTA_ * (atributy témat)

zástupné znaky

Tabulka 375. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_BLOCK	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

Povolené odběry

Tabulka 376. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_SUB_AS_PARENT	0	X'00000000'
MQTA_SUB_BLOKOVÁNO	1	X'00000001'
MQTA_SUB_ALLOWED	2	X'00000002'

Subpropagace proxy

Tabulka 377. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

Publikace povoleny

Tabulka 378. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_PUB_AS_PARENT	0	X'00000000'
MQTA_PUB_BLOKOVÁNO	1	X'00000001'
MQTA_PUB_ALLOWED	2	X'00000002'

MQTC_ * (ovládací prvky spouštěče)

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

MQTCPKEEP_ * (TCP Keepalive)

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

MQTCPSTACK_ * (Typy zásobníků TCP)

Tabulka 381. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

MQTIME_ * (jednotky času formátu příkazu)

Tabulka 382. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTIME_JEDNOTKA_MIN	0	X'00000000'
MQTIME_JEDNOTKY_SEKUNDY	1	X'00000001'

MQTM_ * (Struktura zprávy spouštěče)

Tabulka 383. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQTM_STRUCT	"TM¬¬"
POLE MQTM_STRUC_ID_ARRAY	'T','M','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 384. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTM_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQTM_AKTUÁLNÍ_VERZE	1	X'00000001'

MQTMC_ * (Struktura formátu znaků zprávy spouštěče)

Tabulka 385. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQTC_STRUCT	"TMC¬"
POLE MQTC_STRUC_ID_ARRAY	'T','M','C','¬'
MQTMC_VERSION_1	"bbb1"
MQTMC_VERSION_2	"bbb2"
AKTUÁLNÍ_VERZE MQTMC_VERSION	"bbb2"
MQTMC_VERSION_1_ARRAY	'¬','¬','¬','1'
MQTMC_VERSION_2_ARRAY	'¬','¬','¬','2'
POLE MQTMC_CURRENT_VERSION_VERSION_ARRAY	'¬','¬','¬','2'

MQTOPT_ * (typ tématu)

Tabulka 386. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTOP_LOCAL	0	X'00000000'
KLASTR MQTOP_CLUSTER	1	X'00000001'
MQTOP_ALL	2	X'00000002'

MQTRXSTR_ * (Automatické spuštění trasování inicializátoru kanálu)

Tabulka 387. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

MQTSCOPE_ * (Obor odběru)

Tabulka 388. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTSCOPY_QMGR	1	X'00000001'
MQTSCOPY_ALL	2	X'00000002'

MQTT_ * (Typy spouštěčů)

Tabulka 389. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTTE_NONE	0	X'00000000'
NEJPRVE MQTT_FIRST	1	X'00000001'
MQTT_EVERY	2	X'00000002'
MQTT_DEPTH	3	X'00000003'

MQTYPE_ * (datové typy vlastností)

Tabulka 390. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'
LOGICKÁ HODNOTA MQTYPE_BOOLEAN	4	X'00000004'
ŘETĚZEC 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'
ŘETĚZEC MQTYPE_STRING	1024	X'00000400'

MQUA_ * (Selektory atributu uživatele publikování/odběru)

Tabulka 391. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
NEJPRVE MQUA_	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

MQUIDSUPPR_ * (Podpora ID uživatele formátu příkazu)

Tabulka 392. Hodnoty konstant

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

MQUNDELIVERED_ * (Formát příkazu pro migraci publikování/odběru a hodnot bez doručení)

Tabulka 393. Hodnoty konstant

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

MQUOWST_ * (stavy příkazů pracovní jednotky UOW)

Tabulka 394. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQUOW_NONE	0	X'00000000'

Tabulka 394. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUOWST_ACTIVE	1	X'00000001'
MQUOW_PREPART	2	X'00000002'
MQUOW_UNRESOLVED	3	X'00000003'

MQUOWT_ * (Formát příkazů pracovní jednotky)

Tabulka 395. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUOW_Q_MGR	0	X'00000000'
MQUOW_CICS	1	X'00000001'
MQUOW_RRS	2	X'00000002'
MQUOW_IMS	3	X'00000003'
MQUOW_XA	4	X'00000004'

MQUS_ * (uživatelské použití)

Tabulka 396. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUS_NORMAL	0	X'00000000'
PŘENOS MQUS_TRANSMISSION	1	X'00000001'

MQUSAGE_ * (Hodnoty použití sady stránek sady stránek a hodnoty použití datové sady)

Hodnoty použití sady stránek ve formátu příkazu

Tabulka 397. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
DOSTUPNÉ MQUSAGE_PS_AVAILABLE	0	X'00000000'
DEFINOVÁNO MQUSAGE_PS_DEFINED	1	X'00000001'
MQUSAGE_PS_OFFLINE	2	X'00000002'
MQUSAGE_PS_NOT_DEFINED	3	X'00000003'
MQUSAGE_PS_SUSPENDED	4	X'00000004'
MQUSAGE_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

Hodnoty použití datové sady formátu příkazu

Tabulka 398. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUSAGE_DS_OLDEST_ACTIVE_UOW	10	X'0000000A'
OBNOVA MQUSAGE_DS_OLDEST_PS_RECOVERY	11	X'0000000B'
MQUSAGE_DS_OLDEST_CF_RECOVERY	12	X'0000000C'

MQVL_ * (délka hodnoty)

Tabulka 399. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQVL_NULL_ UKONČENO	-1	X'FFFFFFFF'
MQVL_EMPTY_STRING	0	X'00000000'

MQVU_ * (variabilní ID uživatele)

Tabulka 400. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
UŽIVATEL_OPRAVY_MQVU_USER	1	X'00000001'
MQVU_ANY_USER	2	X'00000002'

MQWDR_ * (Struktura záznamu místa určení uživatelské procedury pracovní zátěže klastru)

Tabulka 401. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQWDR_	"WDR¬"
POLE MQWDR_STRUC_ID_ARRAY	'W','D','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 402. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWDR_VERSION_1	1	X'00000001'
MQWDR_VERSION_2	2	X'00000002'
AKTUÁLNÍ_VERZE MQWDR_CURRENT_VERSION	2	X'00000002'
MQWDR_LENGTH_1	124	X'0000007C'
MQWDR_LENGTH_2	136	X'00000088'
AKTUÁLNÍ_DÉLKA MQWDR_CURRENT_LENGTH	136	X'00000088'

MQWI_ * (Interval čekání)

Tabulka 403. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWI_UNLIMITED	-1	X'FFFFFFFF'

MQWIH_ * (Struktura záhlaví a příznaky informačního obsahu pracovní zátěže)

Struktura záhlaví informací o pracovní zátěži

Tabulka 404. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQWIH_	"WIH¬"
MQWIH_STRUC_ID_POLE	'W','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 405. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQWIH_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQWIH_CURRENT_VERSION	1	X'00000001'
MQWIH_LENGTH_1	120	X'00000078'
AKTUÁLNÍ_DÉLKA MQWIH_CURRENT_LENGTH	120	X'00000078'

Příznaky záhlaví informací o pracovní zátěži

Tabulka 406. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQWIH_NONE	0	X'00000000'

MQWQR_ * (Struktura záznamu výstupní fronty pracovní zátěže klastru)

Tabulka 407. Konstrukce konstant

Název	Struktura
ID_STRUKTURY MQWQ_STRUCTION_ID	"WQR¬"
POLE MQWQR_STRUC_ID_ARRAY	'W','Q','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 408. 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'
AKTUÁLNÍ_VERZE MQWQR_CURRENT_VERSION	3	X'00000003'
MQWQR_LENGTH_1	200	X'000000C8'
MQWQR_LENGTH_2	208	X'000000D0'
MQWQR_LENGTH_3	212	X'000000D4'
AKTUÁLNÍ_DÉLKA_MQWQR_	212	X'000000D4'

MQWS_ * (zástupné schéma)

Tabulka 409. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
VÝCHOZÍ HODNOTA MQWS_DEFAULT	0	X'00000000'
MQWS_CHAR	1	X'00000001'
TÉMA MQWS_TOPIC	2	X'00000002'

MQWXP_ * (Struktura parametru uživatelské procedury pracovní zátěže klastru)

MQWXP_ * (Struktura parametru uživatelské procedury pracovní zátěže klastru)

Tabulka 410. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQWXP_STRUCTION_ID	"WXP¬"
POLE MQWXP_STRUC_ID_ARRAY	'W','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 411. 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'
AKTUÁLNÍ_VERZE MQWXP_CURRENT_VERSION	4	X'00000004'

MQWXP_ * (příznaky pracovní zátěže klastru)

Tabulka 412. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWXP_PUT_BY_CLUSTER_CHL	2	X'00000002'

Související odkazy

“Pole v MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru” na stránce 1527
Popis polí v MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru

MQXACT_ * (Typy volajících rozhraní API)

Tabulka 413. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

MQXC_ * (Výstupní příkazy)

Tabulka 414. 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'

Tabulka 414. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXC_MQBACK	9	X'00000009'
MQXC_MQCMIT	10	X'0000000A'

MQXCC_ * (Odezvy ukončení)

Tabulka 415. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXCC_OK	0	X'00000000'
FUNKCE 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'
UŽIVATELSKÁ PROCEDURA MQXCC_SUPPRESS_EXIT	-5	X'FFFFFFFB'
MQXCC_CLOSE_CHANNEL	-6	X'FFFFFFFA'
MQXCC_REQUEST_ACK	-7	X'FFFFFFF9'
SELHÁNÍ MQXCC_FAILED	-8	X'FFFFFFF8'

MQXDR_ * (Ukončení odezvy)

Tabulka 416. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXDR_OK	0	X'00000000'
MQXDR_CONVERSION_FAILED	1	X'00000001'

MQXE_ * (Prostředí)

Tabulka 417. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXE_OTHER	0	X'00000000'
MQXE_MCA	1	X'00000001'
MQXE_MCA_SVRCONN	2	X'00000002'
MQXE_PŘÍKAZOVÝ_SERVER	3	X'00000003'
MQXE_MQSC	4	X'00000004'

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

Zaregistrovat strukturu voleb vstupního bodu

Tabulka 418. Konstrukce konstant	
Název	**Struktura**
MQXE_STRUCTION_ID	"XEPO"
MQXEEPO_STRUC_ID_POLE	'X','E','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 419. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXEPO_VERSION_1	1	X'00000001'
MQXEEPO_AKTUÁLNÍ_VERZE	1	X'00000001'

Volby ukončení

Tabulka 420. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXEEPO_NONE	0	X'00000000'

MQXF_ * (identifikátory funkce rozhraní API)

Tabulka 421. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KONFIGURACE MQXF_INIT	1	X'00000001'
VÝRAZ MQXF_TERM	2	X'00000002'
PŘIPOJENÍ MQXF_CONN	3	X'00000003'
MQXF_CONNX	4	X'00000004'
DISK MQXF_DISK	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'
SADA MQXF_SET	13	X'0000000D'
ZAČÁTEK MQXF_ZAČÁTEK	14	X'0000000E'
VYKOŘITKA MQXF_	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_XAKRAX	24	X'00000018'
MQXF_SB,% 2	25	X'00000019'
MQXF_XACOPLK.	26	X'0000001A'
MQXF_XAKONEC	27	X'0000001B'
% 1 X 2%	28	X'0000001C'
MQXF_XBOPEN	29	X'0000001D'

Tabulka 421. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXF_XPREPARE	30	X'0000001E'
MQXF_XARECOVER	31	X'0000001F'
MQXF_XAROLBACK	32	X'00000020'
MQXF_XASSTART	33	X'00000021'
MQXF_AXREG	34	X'00000022'
MQXF_AXUNREG	35	X'00000023'

MQXP_ * (struktura parametru ukončení přeletu rozhraní API)

Tabulka 422. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQXP_STRUCTION_ID	"XP¬¬"
POLE MQXP_STRUC_ID_ARRAY	'X','P','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 423. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXP_VERSION_1	1	X'00000001'

MQXPDA_ * (oblast určení problému)

Tabulka 424. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQXPDA_NONE	X'00...00' (48 nul)
POLE MQXPDA_NONE_ARRAY	'\0','\0',... (48 nul)

MQXPT_ * (typy přenosu)

Tabulka 425. 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 426. Konstrukce konstant	
Název	**Struktura**
ID STRUKTURY MQXQ_STRUCTION_ID	"XQH¬"

Tabulka 426. Konstrukce konstant (pokračování)	
Název	**Struktura**
MQXQHL_STRUC_ID_POLE	'X','Q','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 427. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXQH_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQXQHL_AKTUÁLNÍ_VERZE	1	X'00000001'

MQXR_ * (důvody ukončení)

Tabulka 428. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXR_PŘED	1	X'00000001'
MQXR_PO	2	X'00000002'
PŘIPOJENÍ MQXR_CONNECTION	3	X'00000003'
FUNKCE MQXR_INIT	11	X'0000000B'
VÝRAZ MQXR_	12	X'0000000C'
ZPRÁVA MQXR_MSG	13	X'0000000D'
MQXR_XMIT	14	X'0000000E'
Z_ZPR_ZA_ZPRÃ	15	X'0000000F'
MQXR_INIT_SEC	16	X'00000010'
MQXR_RETRY	17	X'00000011'
MQXR_AUTO_CLUSDR	18	X'00000012'
MQXR_AUTO_RECEIVER	19	X'00000013'
MQXR_CLWL_OPEN	20	X'00000014'
MQXR_CLWL_PUT	21	X'00000015'
MQXR_CLWL_MOVE	22	X'00000016'
MQXR_CLWL_REPOS	23	X'00000017'
MQXR_CLWL_REPOS_MOVE	24	X'00000018'
MQXR_END_BATCH	25	X'00000019'
PŘIJATÉ MQXR_ACK_RECEIVED	26	X'0000001A'
FUNKCE MQXR_AUTO_SVRCONN	27	X'0000001B'
SOUBOR MQXR_AUTO_CLURCVR	28	X'0000001C'
MQXR_SEC_PARMS	29	X'0000001D'

MQXR2_* (Ukončení odpovědi 2)

Tabulka 429. 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'

Tabulka 429. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 konce)

Tabulka 430. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
UŽIVATELSKÁ PROCEDURA MQXT_API_CROSSING_EXIT	1	X'00000001'
UŽIVATELSKÁ PROCEDURA MQXT_API_EXIT	2	X'00000002'
MQXT_CHANNEL_SEC_EXIT	11	X'0000000B'
MQXT_CHANNEL_MSG_EXIT	12	X'0000000C'
UŽIVATELSKÁ PROCEDURA MQXT_CHANNEL_SEND_EXIT	13	X'0000000D'
UKONČOVACÍ PROCEDURA MQXT_CHANNEL_RCV_EXIT	14	X'0000000E'
MQXT_CHANNEL_MSG_RETRY_EXIT	15	X'0000000F'
MQXT_CHANNEL_AUTO_DEF_EXIT	16	X'00000010'
UŽIVATELSKÁ PROCEDURA MQXT_CLUSTER_WORKLOAD_EXIT	20	X'00000014'
UŽIVATELSKÁ PROCEDURA MQXT_PUBSUB_ROUTING_EXIT	21	X'00000015'

MQXUA_ * (Výstupní hodnota uživatelské oblasti)

Tabulka 431. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQXA_NONE	X'00...00' (16 nul)
POLE MQXA_NON_ARRAY	'\0','\0',... (16 nul)

MQXWD_ * (Konec struktury deskriptoru čekání)

Tabulka 432. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQXWD_STRUCTION_ID	"XWD¬"
POLE MQXWD_STRUC_ID_ARRAY	'X','W','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 433. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXWD_VERSION_1	1	X'00000001'

MQZAC_ * (Struktura kontextu aplikace)

Tabulka 434. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZAC_STRUCT	"ZAC¬"
MQZAC_STRUC_ID_ARRAY	'Z','A','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 435. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAC_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQZAC_AKTUÁLNÍ_VERZE	1	X'00000001'

MQZAD_ * (struktura dat oprávnění)

Tabulka 436. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQZAD_OBJEKTU	"ZAD¬"
MQZAD_STRUC_ID_POLE	'Z','A','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 437. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAD_VERSION_1	1	X'00000001'
MQZAD_VERSION_2	2	X'00000002'
V_AKTUÁLNÍ_VERZE MQZAD_	2	X'00000002'

MQZAET_ * (typy entit instalovatelné služby)

Tabulka 438. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAET_NONE	0	X'00000000'
ČINITEL MQZAET_PRINCIPAL	1	X'00000001'
SKUPINA MQZAET_GROUP	2	X'00000002'
MQZAET_NEZNÁMÝ	3	X'00000003'

MQZAO_ * (autorizace instalovatelných služeb)

Tabulka 439. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZAO_PŘIPOJENÍ	1	X'00000001'
MQZAO_BROWSE	2	X'00000002'
MQZAO_VSTUP	4	X'00000004'
MQZAO_VÝSTUP	8	X'00000008'
MQZAO_DOTÁZAT SE	16	X'00000010'
MQZAO_SADA	32	X'00000020'

Tabulka 439. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
KONTEXT MQZAO_PASS_IDENTITY_CONTEXT	64	X'00000040'
MQZAO_PASS_ALL_CONTEXT	128	X'00000080'
KONTEXT MQZAO_SET_IDENTITY_CONTEXT	256	X'00000100'
FUNKCE MQZAO_SET_ALL_CONTEXT	512	X'00000200'
MQZAO_ALTERNATE_USER_AUTHORITY	1024	X'00000400'
MQZAO_PUBLIKOVÁNÍ	2048	X'00000800'
MQZAO_SUBSCRIBE	4096	X'00001000'
MQZAO_RESUME	8192	X'00002000'
MQZA_ALL_MQI	16383	X'00003FFF'
VYTVOŘIT_VYTVOŘIT_MQZAO_	65536	X'00010000'
MQZAO_DELETE	131072	X'00020000'
MQZAO_ZOBRAZENÍ	262144	X'00040000'
ZMĚNA MQZAO_CHANGE	524288	X'00080000'
MQZAO_CLEAR	1048576	X'00100000'
MQZAO_CONTROL	2097152	X'00200000'
MQZAO_CONTROL_EXTENDED	4194304	X'00400000'
MQZAO_AUTORIZOVAT	8388608	X'00800000'
MQZAODE_ALL_ADMIN	16646144	X'00FE0000'
MQZAO_VŠE	16662527	X'00FE3FFF'
MQZAO_REMOVE	16777216	X'01000000'
MQZAO_NONE	0	X'00000000'

MQZAS_ * (Verze rozhraní služby instalovatelné služby)

Tabulka 440. 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 441. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POČÁTEČNÍ_KONTEXT MQZATR_CONTEXT	0	X'00000000'
KONTEXT MQZAT_CHANGE_CONTEXT	1	X'00000001'

MQZCI_ * (indikátor pokračování instalovatelné služby)

Tabulka 442. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ HODNOTA MQZCI_DEFAULT	0	X'00000000'
MQZCI_CONTINUE	0	X'00000000'
MQZCI_STOP	1	X'00000001'

MQZED_ * (Struktura dat entity)

Tabulka 443. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZED_STRUCT	"ZED¬"
POLE MQZED_STRUC_ID_ARRAY	'Z','E','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 444. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZED_VERSION_1	1	X'00000001'
MQZED_VERSION_2	2	X'00000002'
AKTUÁLNÍ_VERZE MQZED_VERSION	2	X'00000002'

MQZFP_ * (struktura parametrů Free)

Tabulka 445. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZFP_STRUCT	"ZFP¬"
POLE MQZFP_STRUC_ID_	'Z','F','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 446. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZFP_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQZFP_CURRENT_VERSION	1	X'00000001'

MQZIC_ * (Struktura kontextu identity)

Tabulka 447. Konstrukce konstant	
Název	**Struktura**
MQZIC_STRUCTION_ID	"ZIC¬"
MQZIC_STRUC_ID_POLE	'Z','I','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 448. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZIC_VERSION_1	1	X'00000001'

Tabulka 448. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
AKTUÁLNÍ_VERZE MQZIC_AKTUÁLNÍ_VERZE	1	X'00000001'

MQZID_ * (ID funkcí pro služby)

ID funkcí společná pro všechny služby

Tabulka 449. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT	0	X'00000000'
MQZID_TERM.	1	X'00000001'

ID funkcí pro službu Autorita

Tabulka 450. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_AUTHORITY	0	X'00000000'
MQZID_TERM_AUTHORITY	1	X'00000001'
MQZID_CHECK_AUTHORITY	2	X'00000002'
MQZID_COPY_ALL_AUTHORITY	3	X'00000003'
MQZID_DELETE_AUTHORITY	4	X'00000004'
MQZID_SET_AUTHORITY	5	X'00000005'
MQZID_GET_AUTHORITY	6	X'00000006'
MQZID_GET_EXPLICITNÍ_AUTORITA	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_DOTÁZAT SE	12	X'0000000C'
MQZID_CHECK_PRIVILEGED	13	X'0000000D'

ID funkcí pro službu názvů

Tabulka 451. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_NAME	0	X'00000000'
MQZID_TERM_NAME	1	X'00000001'
MQZID_VYHLEDÁVACÍ_NÁZEV	2	X'00000002'
MQZID_INSERT_NAME	3	X'00000003'
MQZID_DELETE_NAME	4	X'00000004'

ID funkcí pro službu ID uživatele

Tabulka 452. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZID_INIT_ID_UŽIVATELE	0	X'00000000'
MQZID_TERM_USERID	1	X'00000001'
ID UŽIVATELE MQZID_FIND_USERID	2	X'00000002'

MQZIO_ * (Volby inicializace instalovatelných služeb)

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

MQZNS_ * (Verze rozhraní služby názvů)

Tabulka 454. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZNS_VERSION_1	1	X'00000001'

MQZSE_ * (Spuštění instalovatelných služeb-Indikátor výčtu)

Tabulka 455. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SPUŠTĚNÍ MQZSE_START	1	X'00000001'
MQZ_CONTINUE	0	X'00000000'

MQZSL_ * (indikátor selektoru instalovatelných služeb)

Tabulka 456. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZSL_NOT_RETURNED	0	X'00000000'
FUNKCE MQZSL_RETURNED	1	X'00000001'

MQZTO_ * (Instalovatelné služby-volby ukončení)

Tabulka 457. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZTO_PRIMÁRNÍ	0	X'00000000'
MQZ_SEKUNDÁRNÍ	1	X'00000001'

MQZUS_ * (Verze rozhraní služby ID uživatele)

Tabulka 458. 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, polí a deklarace jazyka 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 .

Elementární datové typy

Tato sekce obsahuje informace o datových typech použitých v modulu MQI (nebo ve funkcích ukončení). Jsou podrobně popsány v příkladech, které ukazují, jak deklarovat základní datové typy v podporovaných programovacích jazycích v následujících tématech.

Datové typy použité v rozhraní MQI (nebo ve funkcích ukončení) jsou buď:

- Elementární datové typy nebo
- Agregáty elementárních datových typů (polí nebo struktur)

V modulu MQI (nebo ve funkcích ukončení) se používají následující elementární datové typy:

Tabulka 459. Názvy elementárních datových typů, typy a popisy

Název elementárního datového typu	Datový typ	Popis
MQBOOL	Logická hodnota	Datový typ MQBOOL představuje logickou hodnotu. Hodnota 0 reprezentuje hodnotu false. Jakákoli jiná hodnota představuje true. MQBOOL musí být zarovnán jako pro datový typ MQLONG.

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQBYTE	bajt	Datový typ MQBYTE představuje jeden bajt dat. Žádná konkrétní interpretace není umístěna na bajt; je považována za řetězec bitů, nikoli jako binární číslo nebo znak. Není vyžadováno žádné zvláštní zarovnání. Když jsou data MQBYTE zasílána mezi správci front, kteří používají různé znakové sady nebo kódování, data MQBYTE se nepřevádí žádným způsobem. Pole *MsgId* a *CorrelId* ve struktuře MQMD jsou jako tato. Pole MQBYTE se někdy používá k reprezentaci oblasti hlavní paměti, která není známa správci front. Oblast může například obsahovat data zprávy aplikace nebo strukturu. Vyrovnání hranice této oblasti musí být slučitelné s povahou údajů, které jsou v něm obsaženy. V programovacím jazyku C lze použít libovolný datový typ pro parametry funkce, které jsou zobrazeny jako pole objektů MQBYTE. Důvodem je to, že tyto parametry jsou vždy předávány adresou a v C je parametr funkce deklarován jako ukazatel-to-void.

Tabulka 459. Názvy elementárních datových typů, typy a popisy (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 zvláštní zarovnání. Jsou-li data v bajtovém řetězci kratší než definovaná délka řetězce, data musí být polstrovaná s hodnotami null, aby vyplnily řetězec. Když správce front vrátí do aplikace bajtové řetězce (například na volání MQGET), budou vycpávky správce front s hodnotami null do definované délky řetězce. Pojmenované konstanty jsou k dispozici pro definování délek polí bajtových řetězců. Ty jsou vypsány v “Konstanty” na stránce 60
MQCHAR	Znak	Datový typ MQCHAR představuje jednobajtový znak nebo jeden bajt dvoubajtového nebo vícebajtového znaku. Není vyžadováno žádné zvláštní zarovnání. Při odesílání dat MQCHAR mezi správci front, kteří používají různé znakové sady nebo kódování, data MQCHAR obvykle vyžadují převod, aby byla data interpretována správně. Správce front toto automaticky provede pro data MQCHAR ve struktuře MQMD. Převod dat MQCHAR v datech zprávy aplikace je řízen volbou MQGMO_CONVERT zadanou v rámci volání MQGET. Další podrobnosti naleznete v popisu této volby v části “MQGMO-Volby získání zprávy” na stránce 366 .

Tabulka 459. Názvy elementárních datových typů, typy a popisy (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 zvláštní zarovnání. Pokud jsou data v řetězci kratší než definovaná délka řetězce, data musí být vyplněna mezerami, aby se řetězec vyplnil. V některých případech může být znak null použit k ukončení řetězce předčasně, místo doplnění mezerami; znak hex 00 a znaky následující za ním jsou považovány za mezery, až do definované délky řetězce. Místa, kde může být použita hodnota null, jsou identifikována v popisech volání a datových typů. Když správce front vrátí do aplikace znakové řetězce (například při volání MQGET), bude správce front vždy obsahovat mezery až do definované délky řetězce; správce front nepoužívá k oddělování řetězce znak null. Jsou k dispozici pojmenované konstanty, které definují délky polí znakového řetězce a jsou vypsány v “Konstanty” na stránce 60.
MQFLOAT32	32bitové číslo s pohyblivou řádovou čárkou	Datový typ MQFLOAT32 je 32bitové číslo s pohyblivou řádovou čárkou, které je reprezentováno pomocí standardního formátu IEEE s pohyblivou řádovou čárkou. MQFLOAT32 musí být zarovnáno na 4bajtové hranici. Použití parametru MQFLOAT32 v jazyce C v systému z/OS vyžaduje použití příznaku kompilátoru FLOAT (IEEE). Použití MQFLOAT32 v COBOLu je omezeno na kompilátory, které podporují čísla s pohyblivou řádovou čárkou ve formátu IEEE. To může vyžadovat použití příznaku kompilátoru FLOAT (NATIVE).

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQFLOAT64	číslo s 64bitovým číslem 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 8bajtovou hranici. Použití parametru MQFLOAT64 v jazyce C v systému z/OS vyžaduje použití příznaku kompilátoru FLOAT (IEEE). Použití MQFLOAT64 v COBOLu je omezeno na kompilátory, které podporují čísla s pohyblivou řádovou čárkou ve formátu IEEE. To může vyžadovat použití příznaku kompilátoru FLOAT (NATIVE).
KONFIGURACE MQHCONFIG	Popisovač konfigurace	Datový typ MQHCONFIG představuje konfigurační popisovač, tj. komponentu, která je konfigurována pro konkrétní instalovatelnou službu. Manipulátor konfigurace musí být zarovnán na jeho přirozené hranici. Aplikace se nesmí spoléhat na formát dat uložených uvnitř tohoto popisovače. Je-li tato hodnota platná, bude její hodnota určena k použití v dalších voláních MQI, ale neměla by mít žádný význam kromě tohoto účelu.
MQHCONN	Manipulátor připojení	Datový typ MQHCONN představuje manipulátor připojení, tj. připojení ke konkrétnímu správci front. Manipulátor připojení musí být zarovnán na 4bajtové hranici. Aplikace se nesmí spoléhat na formát dat uložených uvnitř tohoto popisovače. Je-li tato hodnota platná, bude její hodnota určena k použití v dalších voláních MQI, ale neměla by mít žádný význam kromě tohoto účelu.

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)		
Název elementárního datového typu	**Datový typ**	**Popis**
MQZPR	popisovač zprávy	Datový typ MQHMSG představuje popisovač zprávy, který poskytuje přístup ke zprávě. Popisovač zprávy musí být zarovnán na 8bajtovou hranici. Aplikace se nesmí spoléhat na formát dat uložených uvnitř tohoto popisovače. Je-li tato hodnota platná, bude její hodnota určena k použití v dalších voláních MQI, ale neměla by mít žádný význam kromě tohoto účelu.
MQOBJ	Popisovač objektu	Datový typ MQHOTBJ představuje popisovač objektu, který poskytuje přístup k objektu. Popisovač objektu musí být zarovnán na 4bajtové hranici. Aplikace se nesmí spoléhat na formát dat uložených uvnitř tohoto popisovače. Je-li tato hodnota platná, bude její hodnota určena k použití v dalších voláních MQI, ale neměla by mít žádný význam kromě tohoto účelu.
MQINT8	8bitové podepsané celé číslo	Datový typ MQINT8 je 8bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -128 až +127, pokud není jinak omezen kontextem.
MQINT16	16bitové podepsané celé číslo	Datový typ MQINT16 je 16bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -32 768 až +32 767, pokud není jinak omezen kontextem. Hodnota MQINT16 musí být zarovnána na 2bajtovou hranici.
MQINT32	32bitové podepsané celé číslo	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, pokud není jinak omezen kontextem. Viz definice MQLONG.

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)

Název elementárního datového typu	Datový typ	Popis
MQINT64	64bitové podepsané celé číslo	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 není jinak omezen kontextem. V případě jazyka COBOL je platný rozsah omezen na -999 999 999 999 999 až +999 999 999 999 999. 64bitové celé číslo musí být zarovnáno na 8bajtovou hranici.
MQLONG	32bitové podepsané celé číslo	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, pokud není jinak omezen kontextem. Pro COBOL je platný rozsah omezen na -999 999 999 až +999 999 999. MQLONG musí být zarovnáno na 4bajtové hranici.
MQPID	Identifikátor procesu	Identifikátor procesu IBM MQ . Jedná se o stejný identifikátor, který je použit v trasování produktu MQ a výpisů paměti produktu FFST™ , ale může být odlišný od identifikátoru procesu operačního systému.
MQPTR	Ukazatel	Datový typ MQPTR je adresa dat libovolného typu. Ukazatel musí být zarovnán na své přirozené hranici, to je 16bajtová hranice na IBM ia 8bajtová hranice na ostatních platformách. Některé programovací jazyky podporují typované ukazatele; rozhraní MQI je také používá v několika případech (například PMQCHAR a PMQLONG v programovacím jazyku C).

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)		
Název elementárního datového typu	**Datový typ**	**Popis**
MQTID	Identifikátor podprocesu	Identifikátor podprocesu IBM MQ . Jedná se o stejný identifikátor, který je použit v trasování produktu MQ a výpisů paměti produktu FFST™ , ale může být odlišný od identifikátoru podprocesu operačního systému.
MQUINT8	8bitové celé číslo bez znaménka	Datový typ MQUINT8 je 8bitové celé číslo bez znaménka, které může mít jakoukoli hodnotu v rozsahu 0 až +255, pokud není jinak omezen kontextem.
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 není jinak omezeno kontextem. MQUINT16 musí být zarovnán na 2bajtovou hranici.
MQUINT32	32bitové celé číslo bez znaménka	Datový typ MQUINT32 je 32bitové, nepodepsané 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, pokud není jinak omezen kontextem. 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	32bitové celé číslo bez znaménka	Datový typ MQULONG je 32bitové binární celé číslo bez znaménka, které může mít jakoukoli hodnotu v rozsahu 0 až + 4 294 967 294, pokud není jinak omezen kontextem. Pro COBOL je platný rozsah omezen na 0 až +999 999 999. Hodnota MQULONG musí být zarovnána na 4bajtové hranici.
PMQACH	Ukazatel	Ukazatel na datovou strukturu typu MQACH

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)

Název elementárního datového typu	Datový typ	Popis
PMQAIR	Ukazatel	Ukazatel na datovou strukturu typu MQAIR
PMQAXC	Ukazatel	Ukazatel na datovou strukturu typu MQAXC
PMQAXP	Ukazatel	Ukazatel na datovou strukturu typu MQAXP
PMQBHO	Ukazatel	Ukazatel na datovou strukturu typu MQBMHO
OBJEKT 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

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)		
Název elementárního datového typu	**Datový typ**	**Popis**
PMQDMHO	Ukazatel	Ukazatel na datovou strukturu typu MQDMHO
PMQDMPO	Ukazatel	Ukazatel na datovou strukturu typu MQDMPO
PMQEP.	Ukazatel	Ukazatel na datovou strukturu typu MQEPH
PMQFLOAT32	Ukazatel	Ukazatel na datovou strukturu typu MQFLOAT32
PMQFLOAT64	Ukazatel	Ukazatel na datovou strukturu typu MQFLOAT64
PMQFUNC	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

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)

Název elementárního datového typu	Datový typ	Popis
PMQPD	Ukazatel	Ukazatel na datovou strukturu typu MQPD
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
ID PMQTID	Ukazatel	Ukazatel na ID vlákna
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
PMQULELONG.	Ukazatel	Ukazatel na datový typ MQULONG
PMQVOID	Ukazatel	
PMQWIH	Ukazatel	Ukazatel na datovou strukturu typu MQWIH

Tabulka 459. Názvy elementárních datových typů, typy a popisy (pokračování)		
Název elementárního datového typu	**Datový typ**	**Popis**
PMQXQH	Ukazatel	Ukazatel na datovou strukturu typu MQXQH

Prohlášení C

Tabulka 460. 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];`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
MQCHAR32	`typedef MQCHAR MQCHAR32[32];`
MQCHAR48	`typedef MQCHAR MQCHAR48[48];`
MQCHAR64	`typedef MQCHAR MQCHAR64[64];`
MQCHAR128	`typedef MQCHAR MQCHAR128[128];`
MQCHAR256	`typedef MQCHAR MQCHAR256[256];`
MQFLOAT32	`typedef float MQFLOAT32;`
MQFLOAT64	`typedef double MQFLOAT64;`
MQHCONFIG	`typedef void MQPOINTER MQHCONFIG;`
MQHCONN	`typedef MQLONG MQHCONN;`
MQHOBJ	`typedef MQLONG MQHOBJ;`
MQINT8	`typedef signed char MQINT8;`
MQINT16	`typedef short MQINT16;`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení

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;`

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;`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení

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;`

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];`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)	
Datový typ	**Zastoupení**
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];`
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;`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
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;`
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;`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)

Datový typ	Zastoupení
PMQOD	`typedef MQOD MQPOINTER PMQOD;`
PMQPMO	`typedef MQPMO MQPOINTER PMQPMO;`
PMQPTR	`typedef MQPTR MQPOINTER PMQPTR;`
PMQRFH	`typedef MQRFH MQPOINTER PMQRFH;`
PMQRFH2	`typedef MQRFH2[2] MQPOINTER PMQRFH2[2];`
PMQRMH	`typedef MQRMH MQPOINTER PMQRMH;`
PMQTM	`typedef MQTM MQPOINTER PMQTM;`
PMQTMC2	`typedef MQTMC2[2] MQPOINTER PMQTMC2[2];`
PMQUINT8	`typedef MQUINT8 MQPOINTER PMQUINT8;`
PMQUINT16	`typedef MQUINT16 MQPOINTER PMQUINT16;`
PMQULONG	`typedef MQULONG MQPOINTER PMQULONG;`
PMQVOID	`typedef void MQPOINTER PMQVOID;`
PMQWIH	`typedef MQWIH MQPOINTER PMQWIH;`
PMQXQH	`typedef MQXQH MQPOINTER PMQXQH;`
PPMQBO	`typedef PMQBO MQPOINTER PPMQBO;`
PPMQBYTE	`typedef PMQBYTE MQPOINTER PPMQBYTE;`

Tabulka 460. Názvy a reprezentace datových typů C (pokračování)	
Datový typ	**Zastoupení**
PPMQCHAR	`typedef PMQCHAR MQPOINTER PPMQCHAR;`
PPMQCNO	`typedef PMQCNO MQPOINTER PPMQCNO;`
PPMQGMO	`typedef PMQGMO MQPOINTER PPMQGMO;`
PPMQHCONN	`typedef PMQHCONN MQPOINTER PPMQHCONN;`
PPMQHOBJ	`typedef PMQHOBJ MQPOINTER PPMQHOBJ;`
PPMQLONG	`typedef PMQLONG MQPOINTER PPMQLONG;`
PPMQMD	`typedef PMQMD MQPOINTER PPMQMD;`
PPMQOD	`typedef PMQOD MQPOINTER PPMQOD;`
PPMQPMO	`typedef PMQPMO MQPOINTER PPMQPMO;`
PPMQULONG	`typedef PMQULONG MQPOINTER PPMQULONG;`
PPMQVOID	`typedef PMQVOID MQPOINTER PPMQVOID;`
Kde `defined(MQ_64_BIT)` znamená 64bitovou platformu.	

Popis proměnné makra MQPOINTER naleznete v části “Datové typy” na stránce 264 .

Deklarace COBOL

Tabulka 461. Jména a reprezentace datových typů COBOL	
Datový typ	**Zastupování**
MQBOOL	`PIC S9(9) BINARY`
MQBYTE	`PIC X`

Tabulka 461. Jména a reprezentace datových typů COBOL (pokračování)

Datový typ	**Zastupování**
MQBYTE8	`PIC X(8)`
MQBYTE16	`PIC X(16)`
MQBYTE24	`PIC X(24)`
MQBYTE32	`PIC X(32)`
MQBYTE40	`PIC X(40)`
MQCHAR	`PIC X`
MQCHAR4	`PIC X(4)`
MQCHAR8	`PIC X(8)`
MQCHAR12	`PIC X(12)`
MQCHAR20	`PIC X(20)`
MQCHAR28	`PIC X(28)`
MQCHAR32	`PIC X(32)`
MQCHAR48	`PIC X(48)`
MQCHAR64	`PIC X(64)`
MQCHAR128	`PIC X(128)`
MQCHAR256	`PIC X(256)`

Tabulka 461. Jména a reprezentace datových typů COBOL (pokračování)

Datový typ	Zastupování
MQFLOAT32	`USAGE COMP-1`
MQFLOAT64	`USAGE COMP-2`

MQHCONN	zapz/OS
	`PIC S9(9) COMP-5`
	Na ostatních platformách
	`PIC S9(9) BINARY`

MQOBJ	`PIC S9(9) BINARY`
MQINT8	`PIC S9(2) BINARY`
MQINT16	`PIC S9(4) BINARY`
MQINT64	`PIC S9(18) BINARY`
MQLONG	`PIC S9(9) BINARY`
MQPTR	`POINTER`
MQUINT8	`PIC 9(2) BINARY`
MQUINT16	`PIC 9(4) BINARY`
MQUINT64	`PIC 9(18) BINARY`
MQULONG	`PIC 9(9) BINARY`

Deklarace PL/I
Produkt PL/I je podporován v systému z/OS.

Tabulka 462. Názvy a reprezentace datových typů PL/I	
Datový typ	**Zastupování**
MQBOOL	`fixed bin(31)`
MQBYTE	`char(1)`
MQBYTE8	`char(8)`
MQBYTE16	`char(16)`
MQBYTE24	`char(24)`
MQBYTE32	`char(32)`
MQBYTE40	`char(40)`
MQCHAR	`char(1)`
MQCHAR4	`char(4)`
MQCHAR8	`char(8)`
MQCHAR12	`char(12)`
MQCHAR20	`char(20)`
MQCHAR28	`char(28)`
MQCHAR32	`char(32)`
MQCHAR48	`char(48)`
MQCHAR64	`char(64)`

Tabulka 462. Názvy a reprezentace datových typů PL/I (pokračování)	
Datový typ	**Zastupování**
MQCHAR128	`char(128)`
MQCHAR256	`char(256)`
MQFLOAT32	`binary float(21) ieee`
MQFLOAT64	`binary float(52) ieee`
MQHCONN	`fixed bin(31)`
MQOBJ	`fixed bin(31)`
MQINT8	`fixed bin(7)`
MQINT16	`fixed bin(15)`
MQINT64	`fixed bin(63)`
MQLONG	`fixed bin(31)`
MQPTR	`pointer`
MQUINT8	`fixed bin(8)`
MQUINT16	`fixed bin(16)`
MQUINT64	`fixed bin(64)`
MQULONG	`fixed bin(32)`

Deklarace asembleru System/390
Asembler System/390 je podporován pouze v systému z/OS .

Tabulka 463. Názvy datových typů a reprezentace dat asembleru System/390	
Datový typ	**Zastupování**
MQBOOL	`DS F`
MQBYTE	`DS XL1`
MQBYTE8	`DS XL8`
MQBYTE16	`DS XL16`
MQBYTE24	`DS XL24`
MQBYTE32	`DS XL32`
MQBYTE40	`DS XL40`
MQCHAR	`DS CL1`
MQCHAR4	`DS CL4`
MQCHAR8	`DS CL8`
MQCHAR12	`DS CL12`
MQCHAR20	`DS CL20`
MQCHAR28	`DS CL28`
MQCHAR32	`DS CL32`
MQCHAR48	`DS CL48`
MQCHAR64	`DS CL64`

Tabulka 463. Názvy datových typů a reprezentace dat asembleru System/390 (pokračování)	
Datový typ	**Zastupování**
MQCHAR128	`DS CL128`
MQCHAR256	`DS CL256`
MQFLOAT32	`DS EB`
MQFLOAT64	`DS DB`
MQHCONN	`DS F`
MQOBJ	`DS F`
MQINT8	`DS XL1`
MQINT16	`DS H`
MQINT64	`DS D`
MQLONG	`DS F`
MQPTR	`DS F`
MQUINT8	`DS XL1`
MQUINT16	`DS H`
MQUINT64	`DS D`
MQULONG	`DS F`

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

Tabulka 464. Datové typy struktury používané pro volání MQI nebo funkce ukončení

Struktura	Popis	Volání, kde se používá
MQACH	Záhlaví řetězce uživatelské procedury rozhraní API	
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

Tabulka 464. Datové typy struktury používané pro volání MQI nebo funkce ukončení (pokračování)		
Struktura	**Popis**	**Volání, kde se používá**
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 465. 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 konverze dat” na stránce 897spolu 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

Soubory záhlaví

Tabulka 466. 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 COBOL

Tento oddíl obsahuje informace, které vám pomohou používat rozhraní MQI z programovacího jazyka COBOL.

Programování High Level Assembler

Informace, které vám pomohou s použitím rozhraní MQI z programovacího jazyka System/390 Assembler.

- "Makra" na stránce 266
- "Struktury" na stránce 267
- "Makro produktu CMQVERA" na stránce 267
- "Konvence notace" na stránce 267

Makra

Existují dvě makra pro pojmenované konstanty a jedno makro pro každou ze struktur. Tyto soubory jsou shrnuty v následující tabulce.

Tabulka 467. 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

Tabulka 467. Makra modulu sestavení (pokračování)	
Soubor	**Obsah**
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 267

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 267
- "Určení tvaru struktury" na stránce 268
- "Řízení verze struktury" na stránce 268
- "Deklarace jedné struktury vložené do jiné struktury" na stránce 268
- "Určení počátečních hodnot pro pole" na stránce 269
- "Řízení výpisu" na stránce 269

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 468. 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

Tabulka 468. Pole v MQAIR (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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
Poznámka: Zbývající pole jsou ignorována, pokud je hodnota *Verze* menší než hodnota MQAIR_VERSION_2.		
OCSPResponderURL (adresa 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)

Hodnota musí být:

ID_STRUKTURY MQIR_CONSTRUCT
Identifikátor pro záznam ověřovacích informací.

Pro programovací jazyk C je také definována konstanta MQAIR_STRUC_ID_ARRAY; má stejnou hodnotu jako MQAIR_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQAIR_STRUC_ID.

Verze (MQLONG)

Číslo verze struktury MQAIR.

Hodnota musí být jedna z následujících:

MQAIR_VERSION_1
Záznam ověřovacích informací Version-1 .

MQAIR_VERSION_2
Version-2 záznam ověřovacích informací.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQIR_CURRENT_VERSION
Aktuální verze záznamu ověřovacích informací.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQAIR_VERSION_1.

Typ AuthInfo(MQLONG)

Jedná se o typ ověřovacích informací obsažených v záznamu.

Hodnota může být jeden z následujících dvou parametrů:

MQAIT_CRL_LDAP
Kontrola odvolání certifikátů pomocí serveru LDAP.

MQACY_OCSP
Kontrola odvolání certifikátů pomocí protokolu OCSP.

Není-li hodnota platná, volání selže s kódem příčiny MQRC_AUTH_TINFO_TYPE_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQAIT_CRL_LDAP.

AuthInfoConnName (MQCHAR264)
Jedná se 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.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Není-li hodnota platná, volání selže s kódem příčiny MQRC_AUTH_REINFO_CONN_NAME_ERROR.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_AUTH_INFO_CONN_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v jazyce C a prázdné znaky v jiných programovacích jazycích.

LDAPUserNamePtr (PMQCHAR)
Jedná se o jméno uživatele LDAP.

Skládá se z rozlišujícího jména uživatele, který se pokouší o přístup k serveru LDAP CRL. Je-li hodnota kratší než délka zadaná parametrem *LDAPUserNameLength*, ukončete ji znakem null nebo jej odpalovat mezerami na délku *LDAPUserNameLength*. Pole je ignorováno, pokud *LDAPUserNameLength* je nula.

Jméno uživatele služby LDAP můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *LDAPUserNamePtr*

  V takovém případě může aplikace deklarovat řetězec, který je oddělen od struktury MQAIR, a nastavit proměnnou *LDAPUserNamePtr* na adresu řetězce.

  Zvažte použití *LDAPUserNamePtr* pro programovací jazyky, které podporují datový typ ukazatele v módě, který je přenosný do různých prostředí (například programovací jazyk C).
- Použití pole offsetu *LDAPUserNameOffset*

  V takovém případě musí aplikace deklarovat složenou strukturu obsahující strukturu MQSCO, za kterou následuje pole záznamů MQAIR, za nimiž následují řetězce názvů uživatelů LDAP, a nastavit proměnnou *LDAPUserNameOffset* na posun příslušného řetězce názvu od začátku struktury MQAIR. Ujistěte se, že je tato hodnota správná a že má hodnotu, která může být umístěna v rámci MQLONG (nejvíce omezující programovací jazyk je COBOL, pro který je platný rozsah -999 999 999 až +999 999 999).

  Zvažte použití *LDAPUserNameOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele v módě, který nemusí být přenosný do různých prostředí (například programovací jazyk COBOL).

Zvolená technika je vybrána, používá se pouze jeden z *LDAPUserNamePtr* a *LDAPUserNameOffset* ; volání selže s kódem příčiny MQRC_LDAP_USER_NAME_ERROR, pokud jsou oba nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null.

Poznámka: Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Offset LDAPUserName(MQLONG)
Jedná se o posun v bajtech jména uživatele LDAP od začátku struktury MQAIR.

Odsazení může být kladné nebo záporné. Pole je ignorováno, pokud *LDAPUserNameLength* je nula.

Můžete použít buď *LDAPUserNamePtr* nebo *LDAPUserNameOffset* , abyste uvedli jméno uživatele LDAP, ale ne obojí; podrobnosti najdete v popisu pole *LDAPUserNamePtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

Délka LDAPUserName(MQLONG)

Toto je délka v bajtech jména uživatele LDAP adresovaného polem *LDAPUserNamePtr* nebo *LDAPUserNameOffset* . Hodnota musí být v rozsahu nula až MQ_DISTINALGUISHED_NAME_LENGTH. Není-li hodnota platná, volání selže s kódem příčiny MQRC_LDAP_USER_NAME_LENGTH_ERR.

Pokud zahrnutý server LDAP nevyžaduje jméno uživatele, nastavte toto pole na nulu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

LDAPPassword (MQCHAR32)

Jedná se o heslo potřebné pro přístup k serveru CRL LDAP. Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole.

Pokud server LDAP nevyžaduje heslo nebo vynechte jméno uživatele LDAP, *LDAPPassword* musí mít hodnotu null nebo být prázdný. Pokud vynecháte jméno uživatele LDAP a *LDAPPassword* nemá hodnotu null nebo je prázdné, 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 řetězec s hodnotou null v jazyce C a prázdné znaky v jiných programovacích jazycích.

OCSPResponderURL (MQCHAR256)

Pro strukturu MQAIR, která představuje podrobnosti připojení pro odpovídací modul OCSP, obsahuje toto pole adresu URL, na které lze kontaktovat odpovídací modul.

Hodnota tohoto pole je adresa URL protokolu HTTP. Toto pole má přednost před adresou URL v rozšíření certifikátu AuthorityInfoAccess (AIA).

Hodnota je ignorována, pokud nejsou pravdivé obě následující příkazy:

- Struktura MQAIR je verze 2 nebo novější (pole verze je nastaveno na hodnotu MQAIR_VERSION_2 nebo vyšší).
- Pole Typ AuthInfoje nastaveno na hodnotu MQAIT_OCSP.

Pokud pole neobsahuje adresu URL protokolu HTTP ve správném formátu (a není ignorována), volání MQCONNX se nezdaří s kódem příčiny MQRC_OCSP_URL_ERROR.

V tomto poli se rozlišují velká a malá písmena. Musí začínat řetězcem http:// malými písmeny. Zbytek adresy URL může být citlivý na velikost písmen, v závislosti na implementaci serveru OCSP.

Toto pole není předmětem konverze 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 469. 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 469. 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)

Struktura vyrovnávací paměti z vyrovnávací paměti-pole StrucId

Jedná se o identifikátor struktury. Hodnota musí být:

MQBMHO_STRUCTION_ID
Identifikátor pro vyrovnávací paměť pro strukturu zpracování vyrovnávací paměti.

Pro programovací jazyk C je také definována konstanta MQBMHO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQBMHO_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBMHO_STRUC_ID.

Verze (MQLONG)
Struktura obsluhy vyrovnávací paměti z vyrovnávací paměti-pole Verze

Jedná se o číslo verze struktury. Hodnota musí být:

MQBMHO_VERSION_1
Číslo verze pro vyrovnávací paměť pro strukturu vyrovnávací paměti zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQBMHO_CURRENT_VERSION
Aktuální verze vyrovnávací paměti pro strukturu zpracování zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBMHO_VERSION_1.

Volby (MQLONG)
Struktura vyrovnávací paměti pro strukturu zpráv-pole Volby

Hodnota může být následující:

VLASTNOSTI MQBMHO_DELETE_PROPERTIES
Vlastnosti, které jsou přidány do popisovače zpráv, jsou z vyrovnávací paměti odstraněny. Pokud se nezdaří volání, nebudou odstraněny žádné vlastnosti.

Výchozí volby: Pokud nepotřebujete uvedenou volbu použít, použijte následující volbu:

MQBMHO_NONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBMHO_DELETE_PROPERTIES.

V 9.2.4 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 470. 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 470. 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 315
Struktura MQCNO umožňuje aplikaci určit volby související s připojením ke správci front. Struktura je vstupní/výstupní parametr volání MQCONNX.

V 9.2.4 *StrucId (MQCHAR4)*

StrucId je vždy vstupní pole. Jeho počáteční hodnota je BNO.

Hodnota musí být

BNO
Identifikátor struktury pro vyvážení struktury.

Pro programovací jazyk C je také definována konstanta MQBNO_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako BNO, ale je to pole znaků místo řetězce.

Je třeba zadat platnou hodnotu pro **StrucId** nebo je vrácena hodnota MQRC_BNO_ERROR.

V 9.2.4 *Verze (MQLONG)*

Verze je vždy vstupní pole. Jeho počáteční hodnota je MQBNO_VERSION_1.

Hodnota musí být:

MQBNO_VERSION_1
Version-1 struktura-volby struktury.

Je třeba zadat platnou hodnotu pro **Version** nebo je vrácena hodnota MQRC_BNO_ERROR.

V 9.2.4 *ApplicationType (MQLONG)*

Typ volby vyvážení nastavený ve struktuře.

Možné hodnoty jsou:

MQBNO_BALTYPE_SIMPLE
Jednoduché vyvážení; kromě těch, které jsou popsány v tématu Ovlivňování vyrovnávání aplikací v uniformách klastrů, se neuplatňují žádná specifická pravidla.

MQBNO_BALTYPE_REQREP
Vyrovnávání požadavek-odpověď; po každém volání MQPUT se očekává odpovídající volání MQGET pro zprávu odezvy. Vyvažování je pozdrženo, dokud taková zpráva nebyla přijata, nebo byla překročena žádost o prodloužení platnosti požadavku.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQBNO_BALTYPE_SIMPLE.

Musíte zadat pouze jednu hodnotu pro pole **ApplicationType** nebo je vrácena hodnota MQRC_BNO_ERROR.

Poznámka: Další hodnota pro toto pole MQBNO_BALTYPE_RA_MANAGED je rezervována pro použití adaptérem prostředků produktu IBM MQ pro prostředí JEE. Ačkoli se jedná o chybu pro aplikaci, která dodává tuto hodnotu přímo, může být například hlášena při dotazování na stav aplikace.

V 9.2.4 *Časový limit (MQLONG)*

Objekt **Timeout** , po jehož uplynutí může vyrovnávání zátěže přerušit aktivitu aplikace.

Možné hodnoty jsou:

MQBNO_TIMEOUT_AS_DEFAULT
Výchozí hodnota časového limitu pro nastavení.

MQBNO_TIMEOUT_IMMEDIATE
Vyskytne se okamžitý časový limit

MQBNO_TIMEOUT_NEVER
Nenastane žádný časový limit.

Počáteční hodnota tohoto pole je MQBNO_TIMEOUT_AS_DEFAULT.

Musíte zadat pouze jednu hodnotu z definovaných hodnot, nebo hodnotu 0-999999999 sekund, pro pole **Timeout** nebo MQRC_BNO_ERROR.

V 9.2.4 *BalanceOptions (MQLONG)*

Volby vyvážení nastavené aplikací vydávání.

Možné hodnoty jsou:

MQBNO_OPTIONS_NONE
Nejsou nastaveny žádné volby

MQBNO_OPTIONS_IGNORE_TRANS
Nastavení této volby umožňuje nové vyvážení aplikací i v případě, že je transakce uprostřed transakce.

Počáteční hodnota tohoto pole je MQBNO_OPTIONS_NONE.

Můžete zadat libovolnou kombinaci definovaných hodnot s použitím logického nebo znakového pole pro pole **BalanceOptions** . Všechny hodnoty, které nejsou platné, způsobí vrácení operace 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 471. 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 471. 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)

Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQBO_STRUC_ID.

Hodnota musí být:

ID_STRUKTURY OBJEKTU MQBO_STRUCT
Identifikátor pro strukturu begin-options.

Pro programovací jazyk C je také definován konstantní MQBO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQBO_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG)

Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQBO_VERSION_1.

Hodnota musí být:

MQBO_VERSION_1
Číslo verze pro strukturu begin-options.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQBO_CURRENT_VERSION
Aktuální verze struktury begin-options.

Volby (MQLONG)

Toto pole je vždy vstupním polem. 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 472. 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)

Hodnota v tomto poli je identifikátor struktury.

Hodnota musí být:

ID_KONSTRUKCE_MQCBC_
Identifikátor pro strukturu kontextu zpětného volání.

Pro programovací jazyk C je také definována konstanta MQCBC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQCBC_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCBC_STRUC_ID.

Verze (MQLONG)
Hodnota v tomto poli je číslo verze struktury.

Hodnota musí být:

MQCBC_VERSION_1
Version-1 -struktura kontextu zpětného volání.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCBC_CURRENT_VERSION
Aktuální verze struktury kontextu zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCBC_VERSION_1.

Funkce zpětného volání je vždy předána nejnovější verzi struktury.

CallType (MQLONG)
Pole obsahující informace o tom, proč byla tato funkce volána; jsou definovány následující hodnoty.

Typy volání doručování zpráv: Tyto typy volání obsahují informace o zprávě. Parametry **DataLength** a **BufferLength** jsou platné pro tyto typy volání.

MQCBCT_MSG_REMOVED
Funkce odběratele zpráv byla vyvolána se zprávou, která byla destruktivně odebrána z manipulátoru objektu.

Je-li hodnota proměnné *CompCode* MQCC_WARNING, hodnota pole *Reason* je MQRC_TRUNCATED_MSG_ACCEPTED nebo jeden z kódů označující problém převodu dat.

MQCBCT_MSG_NOV_REMOVED
Funkce odběratele zpráv byla vyvolána zprávou, která nebyla dosud destruktivně odebrána z manipulátoru objektu. Zpráva může být destruktivně odebrána z popisovače objektu pomocí *MsgToken*.

Je možné, že zpráva nebyla odebrána, protože:

- Volby MQGMO požádaly o operaci procházení, MQGMO_BROWSE_ *
- Zpráva je větší než dostupná vyrovnávací paměť a volby MQGMO neurčují MQGMO_ACCEPT_TRUNCATED_MSG.

Je-li hodnota proměnné *CompCode* MQCC_WARNING, hodnota pole *Reason* je MQRC_TRUNCATED_MSG_FAILED nebo jeden z kódů označující problém převodu dat.

Typy volání ovládacího prvku zpětného volání: Tyto typy volání obsahují informace o kontrole zpětného volání a neobsahují podrobnosti o zprávě. Tyto typy volání jsou vyžádány pomocí volby Volby ve struktuře MQCBD.

Parametry **DataLength** a **BufferLength** nejsou platné pro tyto typy volání.

VOLÁNÍ MQCBCT_REGISTER_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání, aby provedla nějaké 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 to požadováno, je to první vyvolání funkce zpětného volání.

Hodnota pole *Reason* je MQRC_NONE.

MQCBCT_START_CALL
Účelem tohoto typu volání je povolit funkci zpětného volání, aby provedla určité nastavení při spuštění, například obnovení prostředků, které byly vyčištěny, když již bylo dříve zastaveno.

Funkce zpětného volání je vyvolána při spuštění připojení buď 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 povolit funkci zpětného volání, aby provedla určité vyčištění, když je například zastavena, například při čištění dalších prostředků, které byly získány během přijímání zpráv.

Funkce zpětného volání je vyvolána při zadání volání MQCTL s použitím hodnoty pole *Operation* MQOP_STOP.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *Reason* je nastavena na indikování důvodu zastavení.

VOLÁNÍ MQCBCT_DEREGISTER_CALL
Účelem tohoto typu volání je umožnit funkci zpětného volání, aby provedla závěrečný úklid na konci procesu spotřeby. Funkce zpětného volání je vyvolána, když:

- Funkce zpětného volání je deregistrovaná pomocí volání MQCB s MQOP_DEREGISTER.
- Fronta je zavřena, což způsobí implicitní deregistraci. V této instanci je funkce zpětného volání předána MQHO_UNUSABLE_HOBJ jako popisovač objektu.
- Volání MQDISC bylo dokončeno-způsobilo implicitní zavření a proto zrušení registrace. V tomto případě není připojení okamžitě odpojeno a žádná probíhající transakce nebyla dosud potvrzena.

Pokud se některá z těchto akcí provádí uvnitř samotné funkce zpětného volání, akce se vyvolá až po vrácení zpětného volání.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li to požadováno, jedná se o poslední vyvolání funkce zpětného volání.

Hodnota pole *Reason* je nastavena na indikování důvodu zastavení.

VOLÁNÍ 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 odběratele zpráv

Funkce odběratele zpráv byla vyvolána bez zprávy, pokud byla zjištěna chyba (*CompCode* = MQCC_FAILED), která je specifická pro popisovač objektu; například kód *Reason* = MQRC_GET_INHIBITED.

Hodnota pole *Reason* je nastavena na indikování důvodu pro volání.

VOLÁNÍ MQCBCT_MC_EVENT_CALL

Pro události výběrového vysílání byla vyvolána funkce obslužné rutiny událostí; obslužná rutina událostí je odesílána událostí výběrového vysílání IBM MQ místo 'normálních' IBM MQ událostí.

Další informace o proměnné MQCBCT_MC_EVENT_CALL naleznete v tématu Vytváření sestav výjimek výběrového vysílání.

Objekt Hobj (MQHOBJ)
Jedná se o popisovač objektu pro volání spotřebitele zpráv.

Pro obslužnou rutinu událostí je tato hodnota MQHO_NONE.

Aplikace může použít tento popisovač a token zprávy v bloku Volby načtení zprávy k získání zprávy, pokud zpráva nebyla z fronty odebrána.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQHO_UNUSABLE_HOBJ.

CallbackArea (MQPTR)
Toto pole je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole CallbackArea ve struktuře MQCBD, což je parametr volání MQCB, který slouží k definování funkce zpětného volání.

Změny v produktu *CallbackArea* jsou zachovány v rámci vyvolání funkce zpětného volání pro produkt *HObj*. Toto pole není sdíleno s funkcemi zpětného volání pro jiné popisovače.

Jedná se o vstupní/výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

ConnectionArea (MQPTR)
Toto pole je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole ConnectionArea ve struktuře MQCTLO, což je parametr volání MQCTL, který slouží k řízení funkce zpětného volání.

Jakékoli změny provedené v tomto poli pomocí funkcí zpětného volání jsou zachovány v rámci vyvolání funkce zpětného volání. Tato oblast může být použita k předávání informací, které mají být sdíleny všemi funkcemi zpětného volání. Na rozdíl od *CallbackArea*je tato oblast společná pro všechna zpětná volání pro popisovač připojení.

Jedná se o vstupní a výstupní pole. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CompCode (MQLONG)
Toto pole je kód dokončení. Označuje, zda byly při zpracování zprávy nějaké problémy.

Hodnota je jedna z následujících možností:

MQCC_OK
Úspěšné dokončení

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení)

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQCC_OK.

Příčina (MQLONG)
To je kód příčiny, který kvalifikují *CompCode*.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQRC_NONE.

Stav (MQLONG)
Označení stavu aktuálního spotřebitele. Toto pole má největší hodnotu pro aplikaci, pokud je do funkce spotřebitele předán nenulový kód příčiny.

Toto pole můžete použít ke zjednodušení programování aplikací, protože pro každý kód příčiny není třeba chování kódu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQCS_NONE.

Tabulka 473.

Stav	**Akce správce front**	**Hodnota konstanty**
MQCS_NONE Tento kód příčiny představuje běžné volání bez dalších informací o důvodu	Není; jedná se o normální operaci.	0
MQCS_SUSPENDED_TEMPORARY Tyto kódy příčiny představují dočasné podmínky.	Rutina zpětného volání je volána k hlášení podmínky a poté pozastavena. Po určité době se systém může pokusit o provedení operace znovu, což může vést ke stejnému stavu, kdy se znovu objeví podmínka.	1
MQCS_SUSPENDED_USER_ACTION Tyto kódy příčiny představují podmínky, za kterých zpětné volání potřebuje provést akci k vyřešení podmínky.	Spotřebitel je pozastaven a je volána rutina zpětného volání za účelem hlášení podmínky. Rutina zpětného volání by měla vyřešit podmínku, je-li to možné, a buď RESUME, nebo zavřít připojení.	2
MQCS_SUSPENDED Tyto kódy příčiny představují selhání, která brání dalším zpětným voláním zpráv.	Správce front automaticky pozastaví funkci zpětného volání. Je-li funkce zpětného volání obnovena, je pravděpodobné, že bude znovu přijmout stejný kód příčiny.	3
MQCS_STOPPED Tyto kódy příčiny představují konec spotřeby zpráv.	Doručeno do obslužné rutiny výjimek a pro zpětná volání, která byla zadána příkazem MQCBDO_STOP_CALL. Žádné další zprávy nelze spotřebovat.	4

DataLength (MQLONG)

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 nemusí nutně být délka dat zprávy předávaných spotřebiteli. Může se stát, že zpráva byla zkrácena. Použijte pole ReturnedLength v produktu MQGMO k určení toho, kolik dat bylo skutečně předáno spotřebiteli.

Pokud kód příčiny indikuje, že zpráva byla zkrácena, můžete použít pole DataLength k určení, jak velká je skutečná zpráva. To vám umožňuje určit velikost vyrovnávací paměti potřebné k umístění dat zpráv a pak vydat volání MQCB, aby se aktualizovala hodnota MaxMsgLength s odpovídající hodnotou.

Je-li zadána volba MQGMO_CONVERT, může být převedená zpráva větší než vrácená hodnota parametru DataLength. V takových případech pravděpodobně aplikace potřebuje vydat volání MQCB, aby aktualizovala MaxMsgLength , aby byla větší než hodnota vrácená správcem front pro DataLength.

Chcete-li se vyhnout problémům s oříznutím zprávy, zadejte MaxMsgLength jako MQCBD_FULL_MSG_LENGTH. To způsobí, že správce front alokuje vyrovnávací paměť pro úplnou délku zprávy po převodu dat. Uvědomte si však, že i když je tato volba zadána, je stále možné, že není k dispozici dostatek paměti pro správné zpracování požadavku. Aplikace by měly vždy kontrolovat návratový kód příčiny. Není-li například možné přidělit dostatečnou paměť pro převod zprávy, budou zprávy vráceny do nepřevedené aplikace.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

BufferLength (MQLONG)
Toto pole je délka v bajtech vyrovnávací paměti zpráv, která byla předána této funkci.

Vyrovnávací paměť může být větší než hodnota délky MaxMsgdefinovaná pro spotřebitele a hodnota ReturnedLength v produktu MQGMO.

Skutečná délka zprávy se dodává v poli DataLength .

Aplikace může pro své vlastní účely použít celou vyrovnávací paměť po dobu trvání funkce zpětného volání.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny výjimky.

Příznaky (MQLONG)
Příznaky obsahující informace o tomto odběrateli.

Je definována následující volba:

MQCBCF_READA_BUFFER_EMPTY
Tento příznak může být vrácen v případě, že předchozí volání MQCLOSE pomocí volby MQCO_QUIESCE selhalo s kódem příčiny MQRC_READ_AHEAD_MSGS.

Tento kód indikoval, že je vrácena poslední zpráva dopředného čtení a že vyrovnávací paměť je nyní prázdná. Pokud aplikace vydá další volání MQCLOSE s použitím volby MQCO_QUIESCE (MQCO_QUIESCE), uspěje.

Všimněte si, že aplikace není garantována, aby byla poskytnuta zpráva s touto sadou příznaků, protože stále mohou existovat zprávy v vyrovnávací paměti čtení napřed, které neodpovídají aktuálním kritériím výběru. V této instanci je vyvolávána funkce odběratele s kódem příčiny MQRC_HJBJ_QUIESCED.

Je-li vyrovnávací paměť dopředného čtení zcela prázdná, je spotřebitel vyvolán s příznakem MQCBCF_READA_BUFFER_EMPTY a kódem příčiny MQRC_HJBJ_QUIESCED_NO_MSGS.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

ReconnectDelay (MQLONG)
ReconnectDelay označuje, jak dlouho bude správce front čekat, než se znovu pokusí o nové připojení. Toto pole může upravit obslužnou rutinou událostí, aby došlo ke změně prodlevy nebo zastavení opakovaného připojení.

Pole `ReconnectDelay` použijte pouze v případě, že hodnota pole `Příčina` v kontextu zpětného volání je `MQRC_RECONNECTING`.

Při vstupu do obslužné rutiny událostí je hodnota parametru `ReconnectDelay` počet milisekund, po které bude správce front čekat, než provede pokus o opětovné připojení. V produktu Tabulka 474 na stránce 287 jsou uvedeny hodnoty, které lze nastavit k úpravě chování správce front při návratu z obslužné rutiny událostí.

Tabulka 474. Hodnoty `ReconnectDelay`

Název	Hodnota	Popis
`MQRD_NO_RECONNECT`	`-1`	Neprovádět další pokusy o opětovné připojení. Do aplikace se vrátí chyba.
`MQRD_NO_DELAY`	`0`	Zkuste se okamžitě znovu připojit.
`Milliseconds`	`>0`	Než zopakujete 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 475. 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

Tabulka 475. Pole v MQCBD (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 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)

Struktura deskriptoru zpětného volání-pole StrucId

Jedná se o identifikátor struktury; hodnota musí být:

MQCBD_STRUCTION_ID
Identifikátor pro strukturu deskriptoru zpětného volání.

Pro programovací jazyk C je také definována konstanta MQCBD_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQCBD_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCBD_STRUC_ID.

Verze (MQLONG)

Struktura deskriptoru zpětného volání-pole Verze

Jedná se o číslo verze struktury; hodnota musí být:

MQCBD_VERSION_1
Struktura deskriptoru zpětného volání Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCBD_
Aktuální verze struktury deskriptoru zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCBD_VERSION_1.

CallbackType (MQLONG)

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 se volá tehdy, je-li zpráva splňující zadaná kritéria výběru k dispozici na manipulátoru objektu a připojení je spuštěno.

OBSLUŽNÁ RUTINA MQCBT_EVENT_HANDLER
Definuje toto zpětné volání jako rutinu asynchronních událostí; neřídí se spotřebovávat zprávy pro manipulátor.

Příkaz *Hobj* není vyžadován při volání MQCB, který definuje obslužnou rutinu událostí, a je-li zadán, je ignorován.

Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí spotřebitele zpráv. Funkce odběratele je vyvolána bez zprávy při výskytu události, například zastavení správce front nebo zastavení připojení nebo uvedení do klidového stavu. Nevolá se pro podmínky, které jsou specifické pro jednotlivého spotřebitele zpráv, například MQRC_GET_INHIBITED.

Události jsou doručovány 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)
Struktura deskriptoru zpětného volání-pole Volby

Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

FUNKCE MQCBDO_FAIL_IF_QUIESCING
Volání MQCB selže, 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) ve stavu uvedení do klidového stavu.

Určete MQGMO_FAIL_IF_QUIESCING, v rámci voleb MQGMO předaných volání MQCB, aby bylo oznámení uživatelům oznámeno, když je uváděno do klidového stavu.

Volby ovládacího prvku: Následující volby řídí, zda je funkce zpětného volání volána bez zprávy, když se změní stav spotřebitele:

MQCBDO_REGISTER_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_REGISTER_CALL.

VOLÁNÍ 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.

VOLÁNÍ MQCBDO_DEREGISTER_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_DEREGISTER_CALL.

VOLÁNÍ MQCBDO_EVENT_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT_EVENT_CALL.

VOLÁNÍ MQCBDO_MC_EVENT_CALL
Funkce zpětného volání je vyvolána s typem volání MQCBCT__MC_EVENT_CALL.

Viz CallType , kde získáte další podrobnosti o těchto typech volání.

Výchozí volba: Pokud nepotřebujete žádné 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.

Funkce MQCBDO_NONE je definována pro dokumentaci programu podpory; není určena k tomu, aby byla tato volba použita spolu s jinou hodnotou, ale její hodnota je nulová, protože takové použití nelze zjistit.

Toto je vstupní pole. Počáteční hodnota pole *Options* je MQCBDO_NONE.

CallbackArea (MQPTR)
Struktura deskriptoru zpětného volání-pole CallbackArea

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front neprovádí žádná rozhodnutí založená na obsahu tohoto pole a je beze změny z pole CallbackArea ve struktuře MQCBC, což je parametr v deklaraci funkce zpětného volání.

Hodnota se používá pouze u *Operation* s hodnotou MQOP_REGISTER, bez aktuálně definovaného zpětného volání, a nenahradí předchozí definici.

Jedná se o vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CallbackFunction (MQPTR)
Struktura deskriptoru zpětného volání-pole CallbackFunction

Funkce zpětného volání je vyvolána jako volání funkce.

Toto pole slouží k zadání ukazatele na funkci zpětného volání.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Pokud uvedete obojí, vrátí se kód příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Není-li parametr *CallbackName* ani *CallbackFunction* nastaven, volání selže 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. Např. v programovacím jazyce C zadejte:

```
#pragma linkage(MQCB_FUNCTION,OS)
```

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

Poznámka: Pokud používáte produkt CICS s produktem IBM WebSphere MQ 7.0.1, je asynchronní spotřeba podporována, pokud:

- Apar PK66866 se použije na CICS TS 3.2
- Apar PK89844 se použije na CICS TS 4.1

CallbackName (MQCHAR128)

Struktura deskriptoru zpětného volání-pole CallbackName

Funkce zpětného volání je vyvolána jako dynamicky propojený program.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Pokud uvedete obojí, vrátí se kód příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Není-li parametr *CallbackName* ani *CallbackFunction* nastaven, volání selže s kódem příčiny MQRC_CALLBACK_ROUTINE_ERROR.

Modul je načten při registraci první rutiny zpětného volání a uvolnění při posledním volání rutiny zpětného volání, která má být použita pro zrušení registrace.

Není-li uvedeno v následujícím textu, název je zarovnán vlevo v poli bez vložených mezer; název samotný je doplněn mezerami do délky pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

IBM i

Název zpětného volání může být jeden z následujících formátů:

- Knihovna "/" Program
- Knihovna "/" ServiceProgram "("FunctionName")"

Například `MyLibrary/MyProgram(MyFunction)`.

Název knihovny může být *LIBL. Názvy knihoven a programů jsou omezeny na maximálně 10 znaků.

AIX and Linux

Název zpětného volání je název dynamicky zaveditelného modulu nebo knihovny s příponou s názvem funkce umístěné v této knihovně. Název funkce musí být uzavřen do závorek. Název knihovny může být volitelně s předponou cesty k adresáři:

```
[path]library(function)
```

Není-li cesta zadána, použije se systémová cesta pro vyhledávání.

Název je omezen na maximálně 128 znaků.

Windows
Název zpětného volání je název knihovny s dynamicky propojovacím odkazem s příponou s názvem funkce umístěné v dané knihovně. Název funkce musí být uzavřen v závorkách. Název knihovny může být volitelně s předponou cesty k adresáři a jednotka:

```
[d:][path]library(function)
```

Není-li cesta a cesta uvedena, použije se systémová cesta pro vyhledávání.

Název je omezen na maximálně 128 znaků.

z/OS
Název zpětného volání je název načítaného modulu, který je platný pro specifikaci v parametru EP makra LINK nebo LOAD.

Název je omezen na maximálně 8 znaků.

z/OS CICS
Název zpětného volání je název načítaného modulu, který je platný pro specifikaci v parametru PROGRAM příkazu EXEC CICS LINK.

Název je omezen na maximálně 8 znaků.

Program může být definován jako vzdálený pomocí volby REMOVESYTEM nainstalované definice PROGRAMU nebo pomocí dynamického programu směrování.

Vzdálená oblast CICS musí být připojena k serveru IBM MQ , pokud má program používat volání rozhraní API produktu IBM MQ . Všimněte si však, že pole Hobj ve struktuře MQCBC není platné ve vzdáleném systému.

Dojde-li k selhání při pokusu o načtení *CallbackName*, vrátí se do aplikace jeden z následujících kódů chyby:

- MQRC_MODULE_NOT_FOUND
- MQRC_MODULE_INVALID
- MQRC_MODULE_ENTRY_NOT_FOUND

Do protokolu chyb se zapíše také zpráva obsahující název modulu, pro který byl pokus o načtení proveden, a kód příčiny selhání z operačního systému.

Toto je vstupní pole. Počáteční hodnota tohoto pole je prázdný řetězec nebo prázdný řetězec.

MaxMsgDélka (MQLONG)
Toto je délka v bajtech nejdelší zprávy, kterou lze přečíst z popisovače a poskytnuta pro rutinu zpětného volání. Struktura deskriptoru zpětného volání-pole MaxMsgLength

Má-li zpráva delší délku, přijímá rutina zpětného volání *MaxMsgLength* bajtů zprávy a kód příčiny:

- Objekt MQRC_TRUNCATED_MSG_FAILED nebo
- Objekt MQRC_TRUNCATED_MSG_ACCEPTED, pokud jste zadali hodnotu 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 přizpůsobena systémem pro vrácení zpráv bez oříznutí.

Je-li k dispozici dostatek paměti pro přidělení vyrovnávací paměti k přijetí zprávy, systém zavolá 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, nekonvertované zprávy se předají 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).

Pouľití

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 476. 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

Tabulka 476. Pole v MQCHARV (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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)
Jedná se o ukazatel na řetězec proměnné délky.

Můžete použít buď pole VSPR, nebo VSOffset k uvedení řetězce proměnné délky, ale ne obojí.

Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

VSOffset (MQLONG)
Odsazení může být kladné nebo záporné. Můžete použít buď pole VSPR, nebo VSOffset k uvedení řetězce proměnné délky, ale ne obojí. Posunutí v bajtech proměnné délky proměnné od začátku MQCHARV nebo z struktury obsahující tento řetězec.

Je-li struktura MQCHARV vložena do jiné struktury, bude tato hodnota posunem v bajtech proměnné délky proměnné od začátku struktury, která obsahuje tuto strukturu MQCHARV. Není-li struktura MQCHARV vložena do jiné struktury, například pokud je zadána jako parametr ve volání funkce, posunutí je relativní vzhledem ke spuštění struktury MQCHARV.

Počáteční hodnota tohoto pole je 0.

VSBufSize (MQLONG)
Jedná se o velikost vyrovnávací paměti adresovaná v poli VSPtr nebo VSOffset.

Je-li struktura MQCHARV použita jako výstupní pole ve funkci volání funkce, musí být toto pole inicializováno s délkou poskytnuté vyrovnávací paměti. Je-li hodnota VSLlength větší než hodnota VSBufSize , vrátí se volajícímu do vyrovnávací paměti pouze bajty dat VSBufSize .

Tato hodnota musí být větší než nula nebo rovna nule nebo následující speciální hodnotu, která je rozpoznána:

DÉLKA MQVS_USE_VSLENGTH
Je-li tato hodnota určena, je délka vyrovnávací paměti převzata z pole VSLlength ve struktuře MQCHARV. Nepoužívejte tuto hodnotu, používáte-li strukturu jako výstupní pole a je poskytnuta vyrovnávací paměť.

Toto je počáteční hodnota tohoto pole.

Délka VSLlength (MQLONG)
Délka řetězce proměnné délky adresovaná polem VSPtr nebo VSOffset (v bajtech).

Počáteční hodnota tohoto pole je 0. Hodnota musí být buď větší než nebo rovna nule, nebo následující speciální hodnotu, která je rozeznána:

MQVS_NULL_TERMINATED

Není-li parametr MQVS_NULL_TERMINATED zadán, jsou bajty VSLlength zahrnuty jako součást řetězce. Pokud jsou přítomny znaky null, neoddělují řetězec.

Je-li zadána hodnota MQVS_NULL_TERMINATED, bude řetězec oddělen první hodnotou null, zjištěnou v řetězci. Samotná hodnota null není zahrnuta jako součást tohoto řetězce.

Poznámka: Nulový znak použitý k ukončení řetězce, je-li zadán parametr MQVS_NULL_TERMINATED, má hodnotu null z kódové sady určené VSCCSID.

Například v UTF-16 (CCSID 1200, 13488 a 17584) se jedná o dvoubajtové kódování Unicode, kde hodnota null je představována 16bitovým číslem všech nul. V UTF-16 je běžné najít jednotlivé bajty nastavené na všechny nuly, které jsou součástí znaků (7-bitové ASCII znaky pro instanci), ale řetězce budou ukončeny pouze tehdy, když se dva 'nula' bajtů nacházejí na rovnoměrné hranici bajtů. Je možné získat dva 'nula' bajtů na liché hranici, když jsou každá část platných znaků. Například x '01' x '00 x' 00 'x' 30 ' představuje dva platné znaky Unicode a tento řetězec neukončí null.

VSCCSID (MQLONG)

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.

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
RemoteTransID (CICS RTRANSID, který se má použít)	Není	Mezery
TransactionId (transakce, která se má připojit)	Není	Mezery
FacilityLike (atributy emulované terminálem)	Není	Mezery

Tabulka 477. Pole v MQCIH pro MQCIH (pokračování)

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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
```

Použití

Pokud aplikace vyžaduje hodnoty, které jsou stejné jako počáteční hodnoty zobrazené v souboru Tabulka 477 na stránce 299, 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 478. 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

StrucId (MQCHAR4)

Toto pole je polem požadavku s počáteční hodnotou proměnné MQCIH_STRUC_ID.

Hodnota musí být:

ID_KONSTRUKCE_MQCIH_
Identifikátor pro strukturu záhlaví informací produktu CICS .

Pro programovací jazyk C je také definována konstanta MQCIH_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQCIH_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQLONG)

Toto pole je pole požadavku. Jeho počáteční hodnota je MQCIH_VERSION_2.

Hodnota musí být jedna z následujících:

MQCIH_VERSION_1
Struktura informačního záhlaví Version-1 CICS .

MQCIH_VERSION_2
Struktura informačního záhlaví Version-2 CICS .

Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCIH_CURRENT_VERSION
Aktuální verze struktury záhlaví informací produktu CICS .

StrucLength (MQLONG)
Toto pole je pole požadavku s počáteční hodnotou proměnné MQCIH_LENGTH_2.

Hodnota musí být jedna z následujících:

MQCIH_LENGTH_1
Délka struktury záhlaví informačního obsahu version-1 CICS .

MQCIH_LENGTH_2
Délka struktury záhlaví informačního obsahu version-2 CICS .

Následující konstanta uvádí délku aktuální verze:

AKTUÁLNÍ_DÉLKA MQCIH_CURRENT_LENGTH
Délka aktuální verze struktury záhlaví informačního obsahu produktu CICS .

Kódování (MQLONG)
Toto pole je vyhrazené pole; jeho hodnota není významná. Jeho počáteční hodnota je 0.

Kódování pro podporované struktury, které postupují podle struktury MQCIH, je stejné jako kódování struktury MQCIH samotné a převzaté z jakéhokoli předchozího záhlaví IBM MQ .

CodedCharSetId (MQLONG)
CodedCharSetId je vyhrazené pole; jeho hodnota je nevýznamná. Počáteční hodnota tohoto pole je 0.

ID znakové sady pro podporované struktury, které postupují podle struktury MQCIH, je stejné jako ID znakové sady struktury MQCIH samotné a pochází z jakéhokoli předchozího záhlaví IBM MQ .

Formát (MQCHAR8)
V tomto poli je zobrazen název formátu produktu IBM MQ pro data, která následují za strukturou MQCIH.

Na základě 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 produktu MQMD.

Tento název formátu je také použit pro zprávu odpovědi, pokud má pole *ReplyToFormat* hodnotu MQFMT_NONE.

- V případě požadavků DPL musí být *Format* název formátu COMMAREA.
- Pro požadavky 3270 musí být *Format* CSQCBDCIa funkce mostu nastavuje formát na CSQCBDC0 pro zprávy odpovědi.

Uživatelské procedury pro převod dat pro tyto formáty musí být instalovány ve správci front, ve kterém mají být spuštěny.

Pokud zpráva požadavku generuje chybovou zprávu odpovědi, zpráva s chybovou zprávou má formát názvu 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)
Toto pole je pole požadavku. Počáteční hodnota tohoto pole je MQCIH_NONE.

Hodnota musí být:

MQCIH_NONE
Žádné vlajky.

MQCIH_PASS_EXPIRATION
Zpráva odpovědi obsahuje:

- Stejné volby sestavy vypršení platnosti jako zpráva požadavku.
- Zbývající doba vypršení platnosti ze zprávy požadavku bez úpravy provedené v době zpracování mostu.

Pokud tuto hodnotu vynecháte, doba vypršení platnosti se nastaví na *unlimited*(neomezeno).

MQCIH_REPLYY_WITHUT_NULL
Délka zprávy odpovědi na požadavek programu CICS DPL je upravena tak, aby vyloučila koncové hodnoty null (X'00 ') na konci COMMAREA, kterou vrátil program DPL. Není-li tato hodnota nastavena, hodnoty null mohou být značné a bude vrácena celá oblast COMMAREA.

MQCIH_SYNC_ON_RETURN
Odkaz CICS pro požadavky DPL používá volbu SYNCONRETURN, což způsobí, že produkt CICS vezme synchronizační bod, když se program dokončí, pokud je dodán do jiné oblasti CICS . Most neurčuje, do kterého regionu produktu CICS má být požadavek odeslán; to je řízeno definicí programu CICS nebo zařízením pro vyrovnávání pracovní zátěže.

ReturnCode (MQLONG)
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 478 na stránce 304). 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)
Toto pole je pole odpovědi. Jeho počáteční hodnota je MQCC_OK

Hodnota vrácená v tomto poli závisí na *ReturnCode* ; viz Tabulka 478 na stránce 304.

Příčina (MQLONG)
Toto pole je pole odpovědi. Jeho počáteční hodnota je MQRC_NONE.

Hodnota vrácená v tomto poli závisí na *ReturnCode* ; viz Tabulka 478 na stránce 304.

UOWControl (MQLONG)

Toto pole je pole požadavku, které řídí zpracování jednotky práce provedené serverem CICS bridge. Počáteční hodnota tohoto pole je MQCUOWC_ONLY.

Můžete požadovat, aby most spustil jednu transakci, nebo jeden nebo více programů v rámci transakce. Pole uvádí, zda produkt CICS bridge spustí jednotku práce, provede požadovanou funkci v rámci aktuální jednotky práce nebo ukončí jednotku práce tím, že ji potvrdí nebo ji zálohuje. Jsou podporovány různé kombinace, aby se optimalizovalo toky přenosu dat.

Hodnota musí být jedna z následujících:

POUZE MQCUOWC_ONLY
Spuštění pracovní jednotky, provedení funkce a pak potvrzení jednotky práce.

MQCUOWC_CONTINUE
Další data pro aktuální jednotku práce (pouze 3270).

NEJPRVE MQCUOWC_FIRST
Spustit jednotku práce a provést funkci.

MQCUOWC_MIDDLE
Provést funkci v rámci aktuální jednotky práce

MQCUOWC_LAST
Provést funkci a pak potvrdit jednotku práce.

MQCUOWC_COMMIT
Potvrdit jednotku práce (pouze DPL).

MQCUOWC_BACKOUT.
Zálohovat pouze jednotku práce (pouze DPL).

Interval GetWait(MQLONG)

Toto pole je pole požadavku. Jeho počáteční hodnota je MQCGWI_DEFAULT.

Toto pole je použito pouze v případě, že hodnota *UOWControl* má hodnotu MQCUOWC_FIRST. Umožňuje odesílající aplikaci určit přibližný čas v milisekundách, po který bude volání MQGET vydaná mostem čekat na druhé a následující zprávy požadavků pro jednotku práce spuštěnou touto zprávou. Toto zařízení přepíše výchozí čekací interval použitý mostem. Můžete použít následující speciální hodnoty:

MQCGWI_DEFAULT
Předvolený interval čekání.

Tato hodnota způsobí, že CICS bridge čeká na čas uvedený při spuštění mostu.

MQWI_UNLIMITED
Neomezený interval čekání.

LinkType (MQLONG)

Toto pole je pole požadavku. Jeho počáteční hodnota je MQCLT_PROGRAM.

Tato hodnota určuje typ objektu, který se most pokouší propojit. Musí se jednat o jednu z následujících hodnot:

MQCLT_PROGRAM
Program DPL.

TRANSAKCE MQCLT_TRANSACTION
transakce 3270.

Délka OutputData(MQLONG)

Toto pole je pole požadavku použité pouze pro programy DPL. Jeho počáteční hodnota je MQCODL_AS_INPUT.

Tato hodnota představuje délku uživatelských dat, která má být vrácena klientovi ve zprávě s odpovědí. Tato délka zahrnuje 8bajtový název programu. Délka oblasti COMMAREA předaná k propojenému programu je maximum tohoto pole a délka uživatelských dat ve zprávě požadavku minus 8.

Poznámka: Délka uživatelských dat ve zprávě je délka zprávy kromě struktury MQCIH.

Je-li délka uživatelských dat ve zprávě požadavku menší než hodnota *OutputDataLength*, použije se volba DATALENGTH příkazu LINK , což umožňuje efektivní funkci produktu LINK v jiné oblasti CICS .

Můžete použít následující speciální hodnotu:

MQCODL_AS_INPUT
Délka výstupu je stejná jako vstupní délka.

Tato hodnota může být potřebná i v případě, že není požadována žádná odpověď, aby se zajistilo, že COMMAREA předaná do propojeného programu má dostatečnou velikost.

FacilityKeep-čas (MQLONG)
FacilityKeepČas je doba v sekundách, po kterou je prostředek mostu udržován po ukončení uživatelské transakce.

Pro pseudo-konverzační transakce zadejte hodnotu, která odpovídá očekávané době trvání pseudo-konverzace; zadejte nulu pro poslední transakci pseudo-konverzace a pro jiné typy transakcí uveďte nulu.

Toto pole je polem požadavku, které se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0.

ADSDescriptor (MQLONG)
Toto pole je indikátorem určujícím, zda odesílat deskriptory ADS na požadavky SEND a RECEIVE BMS.

Jsou definovány tyto hodnoty:

MQCADSD_NONE
Neodesílat nebo přijímat deskriptory ADS.

MQCADSD_SEND
Odeslat deskriptory ADS.

MQCADSD_RECV
Přijímat deskriptory ADS.

FORMÁT ZPRÁVY MQCADSD_MSGFORMAT
Použít formát zpráv pro deskriptory ADS.

Tím se odešle nebo přijímá deskriptory ADS pomocí dlouhé formy deskriptoru ADS. Dlouhá forma má pole, která jsou zarovnána na 4bajtové hranice.

Nastavte pole *ADSDescriptor* následujícím způsobem:

- Pokud nepoužíváte deskriptory ADS, nastavte pole na hodnotu MQCADSD_NONE.
- Používáte-li deskriptory ADS se *stejným* CCSID v každém prostředí, nastavte pole na součet příkazů MQCADSD_SEND a MQCADSD_RECV.
- Používáte-li deskriptory ADS s *různými* CCSID v každém prostředí, nastavte pole na součet příkazů MQCADSD_SEND, MQCADSD_RECV a MQCADSD_MSGFORMAT.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je MQCADSD_NONE.

ConversationalTask (MQLONG)
Toto pole je indikátorem, který uvádí, zda povolit úloze vydávat požadavky na další informace, nebo zastavit úlohu a vydat neaktuální zprávu.

Hodnota musí být jedna z následujících voleb:

MQCKT_YES
Úloha je dialogová.

MQCCT_NO
Úloha není dialogová.

Toto pole je polem požadavku, které se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je MQCCT_NO.

Stav TaskEndStav (MQLONG)
Toto pole je pole odezvy, které zobrazuje stav transakce uživatele na konci úlohy. Pole se používá pouze pro transakce 3270 a jeho počáteční hodnota je MQCTES_NOSYNC.

Je vrácena jedna z následujících hodnot:

MQCTES_NOSYNC
Nesynchronizováno.

Transakce uživatele nebyla dosud dokončena a nebyla synchronizovaná. Pole *MsgType* v MQMD je MQMT_REQUEST v tomto případě.

MQCTES_COMMIT
Potvrdit jednotku práce.

Transakce uživatele se ještě nedokončila, ale syncpointa první transakce byla synchronizována. Pole *MsgType* v MQMD je MQMT_DATAGRAM v tomto případě.

MQCTES_BACKOUT
Zazálohujte jednotku práce.

Transakce uživatele nebyla dosud dokončena. Aktuální pracovní jednotka je vrácena zpět. Pole *MsgType* v MQMD je MQMT_DATAGRAM v tomto případě.

ÚLOHA MQCTES_ENDTASK
Ukončit úlohu.

Transakce uživatele byla ukončena (nebo ukončena). Pole *MsgType* v MQMD je MQMT_REPLY v tomto případě.

Zařízení (MQBYTE8)
V tomto poli je zobrazen osmibajtový token mechanizmu mostu.

Token funkce mostu umožňuje více transakcí v pseudokonverzaci pro použití stejné funkce mostu (virtuální terminál 3270). V první nebo jediné zprávě v pseudo konverzaci nastavte hodnotu MQCFAC_NONE. Tato hodnota sděluje příkazu CICS , že má přidělit nové zařízení mostu pro tuto zprávu. Token prostředku mostu je vrácen ve zprávách odezvy, je-li na vstupní zprávě uveden nenulový *FacilityKeepTime* . Následné vstupní zprávy v rámci pseudokonverzace musí poté používat stejný token prostředku mostu.

Je definována následující speciální hodnota:

MQCFAC_NONE
Nebyl zadán token zařízení.

Pro programovací jazyk C je také definována konstanta MQCFAC_NONE_ARRAY a má stejnou hodnotu jako MQCFAC_NONE, ale je to pole znaků namísto řetězce.

Toto pole je pole požadavku i pole odezvy použité pouze pro transakce 3270. Délka tohoto pole je dána proměnnou MQ_FACILITY_LENGTH. Počáteční hodnota tohoto pole je MQCFAC_NONE.

Funkce (MQCHAR4)
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 478 na stránce 304. 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)
Hodnota AbendCode je pole odezvy. 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 ABCODE CICS .

Ověřovatel (MQCHAR8)
Hodnota tohoto pole je heslo nebo přístupový lístek.

Pokud je ověření identifikátoru uživatele aktivní pro CICS bridge, použije se *Authenticator* spolu s identifikátorem uživatele v kontextu identity MQMD k ověření odesílatele zprávy.

Toto je pole požadavku. Délka tohoto pole je dána hodnotou MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota tohoto pole je 8 mezer.

Reserved1 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

Formát ReplyTo(MQCHAR8)
Hodnota tohoto pole je název formátu IBM MQ zprávy odpovědi, která se odesílá jako odpověď na aktuální zprávu.

Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro kódování pole *Format* v produktu 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.

ID RemoteSys(MQCHAR4)
Toto pole uvádí identifikátor systému CICS systému CICS , který zpracovává požadavek.

Je-li toto pole prázdné, je systémový požadavek CICS zpracován na stejném systému CICS jako monitor mostu. Použité SYSID je vráceno ve zprávě odpovědi.

Pro pseudo-konverzaci 3270 musí všechny následné zprávy v rámci konverzace uvádět vzdálené SYSID vrácené v počáteční odpovědi. Je-li uveden, SYSID musí:

- Buďte aktivní.
- Mít přístup ke frontě požadavků IBM MQ .
- být přístupný prostřednictvím odkazů na konzolu CICS ISC ze systému CICS monitoru mostu.

ID RemoteTrans(MQCHAR4)
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)
Toto pole je pole požadavku. Jeho délka je dána hodnotou MQ_TRANSACTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

Má-li *LinkType* hodnotu MQCLT_TRANSACTION, *TransactionId* je identifikátor transakce uživatelské transakce, která se má spustit; v tomto případě zadejte neprázdnou hodnotu.

Má-li *LinkType* hodnotu MQCLT_PROGRAM, *TransactionId* je kód transakce, pod kterým mají být spuštěny všechny programy v jednotce práce. Zadáte-li prázdnou hodnotu, použije se výchozí kód transakce mostu CICS DPL (CKBP). Je-li hodnota neprázdná, musíte ji definovat jako lokální transakci CICS s počátečním programem, který je CSQCBP00. Toto pole je použito pouze v případě, že hodnota *UOWControl* má hodnotu MQCUOWC_FIRST nebo MQCUOWC_ONLY.

FacilityLike (MQCHAR4)
FacilityLike je název instalovaného terminálu, který má být použit jako model pro zařízení mostu.

Hodnota mezer znamená, že produkt *FacilityLike* je převzat z definice profilu transakce mostu, nebo se použije výchozí hodnota.

Toto pole je polem požadavku, které se používá pouze pro transakce 3270. Délka tohoto pole je dána proměnnou MQ_FACILITY_LIKE_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

AttentionId (MQCHAR4)
Hodnota v tomto poli určuje počáteční hodnotu klíče AID, když je transakce spuštěna. Je to 1bajtová hodnota, zarovnaná vlevo.

AttentionId je pole požadavku použité pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_ATTENTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

StartCode (MQCHAR4)
Hodnota tohoto pole je indikátorem určujícím, zda most emuluje transakci terminálu nebo transakci iniciovanou pomocí příkazu START.

Hodnota musí být jedna z následujících:

MQCSC_START
Spustit.

POČÁTEČNÍ_DATA MQCSC_STARTDATA
Spustit data.

MQCSC_TERMINPUT
Vstup terminálu.

MQCSC_NONE
Není.

Ve všech případech jsou pro programovací jazyk C také definovány konstanty MQCSC_ * _ARRAY; tyto konstanty mají stejné hodnoty jako odpovídající konstanty MQCSC_ *, ale jsou to pole znaků místo řetězců.

V odpovědi na můstek je toto pole nastaveno na počáteční kód odpovídající dalšímu ID transakce, které je obsaženo v poli *NextTransactionId* . V odezvě jsou možné následující spouštěcí kódy:

- MQCSC_START
- POČÁTEČNÍ_DATA MQCSC_STARTDATA
- MQCSC_TERMINPUT

Pro CICS Transaction Server 1.2je toto pole pouze pole požadavku; jeho hodnota v odpovědi není definována.

Pro produkt CICS Transaction Server 1.3 a následující vydání je toto pole jak pole požadavku, tak pole odezvy.

Toto pole se používá pouze pro transakce 3270. Délka tohoto pole je dána proměnnou MQ_START_CODE_LENGTH. Počáteční hodnota tohoto pole je MQCSC_NONE.

CancelCode (MQCHAR4)
Hodnota v tomto poli je kód nestandardního ukončení, který má být použit k ukončení transakce (obvykle konverzační transakce, která požaduje více dat). Jinak je toto pole nastaveno na mezery.

Toto pole je polem požadavku, které se používá pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_CANCEL_CODE_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

ID NextTransaction(MQCHAR4)
Tato hodnota je název další transakce vrácené uživatelskou transakcí (obvykle EXEC CICS RETURN TRANSID). Pokud žádná další transakce neexistuje, je toto pole nastaveno na mezery.

Toto pole je pole odezvy použité pouze pro transakce 3270. Délka tohoto pole je dána hodnotou MQ_TRANSACTION_ID_LENGTH. Počáteční hodnota tohoto pole je čtyři mezery.

Reserved2 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

Reserved3 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

CursorPosition (MQLONG)
Hodnota v tomto poli ukazuje počáteční pozici kurzoru, když je transakce spuštěna. V případě konverzačních transakcí je pozice kurzoru v vektoru PŘÍJMU.

Toto pole je polem požadavku, které se používá pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud *Version* je menší než MQCIH_VERSION_2.

ErrorOffset (MQLONG)
Pole ErrorOffset zobrazuje pozici neplatných dat zjištěných uživatelskou procedurou mostu. Toto pole poskytuje posun od začátku zprávy do umístění neplatných dat.

ErrorOffset je pole odezvy použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud *Version* je menší než MQCIH_VERSION_2.

InputItem (MQLONG)
Toto pole je rezervované pole. Hodnota musí být 0.

Toto pole není přítomno, pokud *Version* je menší než MQCIH_VERSION_2.

Reserved4 (MQLONG)
Toto pole je rezervované pole. Hodnota musí být 0.

Toto pole není přítomno, pokud *Version* je 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 s klienty IBM MQ .

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 479. 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)

Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQCMHO_STRUC_ID.

Jedná se o identifikátor struktury; hodnota musí být:

MQCMHO_STRUCTION_ID
Identifikátor pro strukturu voleb pro vytváření zpracování zpráv.

Pro programovací jazyk C je také definována konstanta **MQCMHO_STRUC_ID_ARRAY** ; tato hodnota má stejnou hodnotu jako **MQCMHO_STRUC_ID**, ale je pole znaků místo řetězce.

Verze (MQLONG)

Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQCMHO_VERSION_1.

Jedná se o číslo verze struktury; hodnota musí být:

MQCMHO_VERSION_1
Version-1 vytvoří strukturu voleb zpracování zpráv.

Následující konstanta uvádí číslo verze aktuální verze:

MQCMHO_CURRENT_VERSION
Aktuální verze struktury voleb popisovače vytvoření zprávy.

Volby (MQLONG)

Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQCMHO_DEFAULT_VALIDATION.

Je možné zadat jednu z následujících možností:

MQCMHO_VALIDATE

Je-li volána funkce **MQSETMP** k nastavení vlastnosti v tomto popisovači zprávy, je název vlastnosti ověřen, aby bylo zajištěno, že:

- neobsahuje neplatné znaky.
- nezačíná JMS nebo usr.JMS s výjimkou následujících:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType.
  - JMSXGroupID
  - JMSXGroupSeq

  Tyto názvy jsou vyhrazeny pro vlastnosti produktu JMS .
- není jedním z následujících klíčových slov, v libovolné směsi s velkými nebo malými písmeny:
  - AND
  - BETWEEN
  - Esc
  - NEPRAVDA
  - IN
  - IS
  - Jako
  - NE
  - NULL
  - OR
  - PRAVDA
- se nezačíná textem Body. nebo Kořen. (kromě Root.MQMD.).

Je-li vlastnost MQdefinovaná uživatelem (mq. *) a název je rozpoznán, pole deskriptoru vlastností jsou nastavena na správné hodnoty pro vlastnost. Není-li vlastnost rozpoznána, je pole *Support* deskriptoru vlastností nastaveno na hodnotu **MQPD_OPTIONAL**.

MQCMHO_DEFAULT_VALIDATION

Tato hodnota určuje, že se má provést výchozí úroveň ověření názvů vlastností.

Výchozí úroveň ověření je ekvivalentní úrovni určené parametrem **MQCMHO_VALIDATE**.

Tato hodnota je výchozí hodnotou.

MQCMHO_NO_VALIDATION

Nedojde k ověření platnosti názvu vlastnosti. Viz popis **MQCMHO_VALIDATE**.

Výchozí volba: Pokud není vyžadována žádná z uvedených předchozích voleb, lze použít následující volbu:

MQCMHO_NONE

Všechny volby předpokládají jejich výchozí hodnoty. Použijte tuto hodnotu, chcete-li označit, že nebyly zadány žádné další volby. Produkt **MQCMHO_NONE** pomáhá dokumentaci programu; není určeno, že tato volba je použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze 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

Všechny verze struktury MQCNO s výjimkou MQCNO_VERSION_4jsou 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 480. Pole v 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.		

Tabulka 480. Pole v MQCNO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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.
CCDTUrlLength (délka adresy URL CCDT)	Není	Délka řetězce identifikovaného pomocí *CCDTUrlPtr* nebo *CCDTUrlOffset*
CCDTUrlPtr (ukazatel na adresu URL CCDT)	Není	Ukazatel na řetězec, který obsahuje adresu URL, pro identifikaci umístění tabulky kanálů připojení klienta, která se má použít pro připojení.
CCDTUrlOffset (offset adresy URL CCDT)	Není	Posun v bajtech od řetězce, který obsahuje adresu URL identifikující umístění tabulky kanálu připojení klienta, která se má použít pro připojení.
V 9.2.0 V 9.2.0 **Poznámka:** Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než MQCNO_VERSION_7.		
V 9.2.0 ApplName (název nastavený aplikací)	Není	Název nastavený aplikací pro identifikaci připojení ke správci front
V 9.2.0 Reserved2 (vyhrazené pole)	Není	Vyhrazené pole pro vyplnění struktury na 64bitovou hranici.

Tabulka 480. Pole v MQCNO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
V 9.2.4 V 9.2.4 **Poznámka:** Zbývající pole jsou ignorována, pokud je hodnota *Version* menší než hodnota MQCNO_VERSION_8.		
V 9.2.4 BalanceParms| Offset	Není	Posun v bajtech ke struktuře MQBNO
V 9.2.4 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

C prohlášení pro MQCNO

V 9.2.0

```
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 to 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 */
};
```

V 9.2.4

```
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

V 9.2.0

```
**   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 CCDTUrlPtr or CCDTUrlOffset
   15 MQCNO-CCDTURLLENGTH
**    Pointer to a string which contains a URL, to identify the location of the client
connection channel
   15 MQCNO-CCDTURLPTR
**    Offset in bytes from a 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
```

V 9.2.4

```
**   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

V 9.2.0

```
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
*/
```

V 9.2.4

```
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

V 9.2.0

```
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
```

V 9.2.4

```
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

V 9.2.0

```
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'
  End Type
```

V 9.2.4

```
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)

StrucId je vždy vstupní pole. Jeho počáteční hodnota je MQCNO_STRUC_ID.

Hodnota musí být:

MQCNO_STRUCTION_ID
Identifikátor pro strukturu voleb připojení.

Pro programovací jazyk C je také definována konstanta MQCNO_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako MQCNO_STRUC_ID, ale je to pole znaků namísto řetězce.

Verze (MQLONG)
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 9.2.4 **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)
Volby, které řídí akci MQCONNX.

Volby účtování

Následující volby určují typ evidence, je-li atribut správce front produktu **AccountingConnOverride** nastaven na hodnotu MQMON_ENABLED:

MQCNO_ACCOUNTING_MQI_ENABLED
Je-li shromažďování dat monitorování zakázáno v definici správce front nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, je nastavení tohoto příznaku umožněno shromažďování dat evidence MQI.

MQCNO_ACCOUNTING_MQI_DISABLED
Pokud je shromažďování dat monitorování zakázáno v definici správce front nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku zastaví shromažďování dat evidence MQI.

MQCNO_ACCOUNTING_Q_ENABLED
Je-li shromažďování dat evidence front v definici správce front zakázáno nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, povoluje nastavení tohoto příznaku shromažďování dat evidence pro tyto fronty, které specifikují správce front v poli *MQIAccounting* příslušné definice fronty.

MQCNO_ACCOUNTING_Q_DISABLED
Je-li shromažďování dat evidence front v definici správce front zakázáno nastavením atributu **MQIAccounting** na hodnotu MQMON_OFF, nastavení tohoto příznaku vypíná shromažďování dat evidence pro fronty, které určují správce front v poli *MQIAccounting* příslušné definice fronty.

Pokud není definován žádný z těchto parametrů, je evidence připojení tak, jak je definováno v atributech správce front.

Volby vazeb

Následující volby řídí typ vazby IBM MQ , která má být použita. Uveďte pouze jednu z těchto voleb:

VAZBA MQCNO_STANDARD_BINDING
Aplikace a lokální agent správce front (komponenta, která spravuje operace front) běží v samostatných jednotkách provedení (obvykle v samostatných procesech). Toto uspořádání udržuje integritu správce front; to znamená, že chrání správce front před chybnými programy.

Pokud správce front podporuje více typů vazeb a nastavíte parametr MQCNO_STANDARD_BINDING, použije správce front atribut **DefaultBindType** ze stanzy Connection v souboru qm.ini , aby vybral skutečný typ vazby. Není-li tato stanza definována nebo ji nelze použít nebo není vhodná pro danou aplikaci, správce front vybere vhodný typ vazby. Správce front nastaví skutečný typ vazby použitý ve volbách připojení.

Použijte MQCNO_STANDARD_BINDING v situacích, kdy aplikace možná nebyla plně otestována, nebo může být nespolehlivá nebo nedůvěryhodná. MQCNO_STANDARD_BINDING je výchozí nastavení.

Tato volba je podporována ve všech prostředích.

Pokud odkazujete na knihovnu produktu mqm , je nejprve proveden pokus o použití standardního připojení k serveru s použitím výchozího typu vazby. Pokud se základní knihovna serveru nepodařilo načíst, bude namísto toho proveden pokus o připojení klienta.

- Chcete-li změnit chování MQCONN (nebo MQCONNX, je-li zadáno MQCNO_STANDARD_BINDING), nastavte proměnnou prostředí MQ_CONNECT_TYPE na jednu z následujících voleb. Všimněte si, že je zde výjimka: Je-li MQCNO_FASTPATH_BINDING uvedeno s hodnotou MQ_CONNECT_TYPE nastaveným na LOCAL nebo STANDARD, může administrátor snížit úroveň zkrácenou připojení bez související změny na aplikaci.

Tabulka 481. Hodnoty pro MQ_CONNECT_TYPE, které mění chování MQCONN nebo MQCONNX

Hodnota	**Význam**
CLIENT	Pokus o připojení klienta se provede pouze.
Rychlý	Tato hodnota byla v předchozích verzích podporována, ale je nyní ignorována, pokud byla zadána.
LOKÁLNÍ	Pokus o připojení k serveru se pouze pokusil. Připojení zrychleného přístupu se sníží na standardní připojení k serveru.
STANDARD	Podporováno z důvodu kompatibility s předchozími verzemi. Tato hodnota je nyní považována za LOCAL.

- Není-li proměnná prostředí MQ_CONNECT_TYPE nastavena při volání MQCONNX, je proveden pokus o standardní připojení k serveru s použitím výchozího typu vazby. Pokud se knihovna serveru nepodaří načíst, dojde k pokusu o připojení klienta.

VAZBA MQCNO_FASTPATH_BINDING
Aplikace a lokální agent správce front jsou součástí stejné jednotky provedení. Na rozdíl od typické metody vázání, kde se aplikace a lokální správce front spouštějí v samostatných jednotkách provádění.

MQCNO_FASTPATH_BINDING je ignorována, pokud správce front nepodporuje tento typ vazby; zpracování pokračuje, jako by tato volba nebyla uvedena.

MQCNO_FASTPATH_BINDING může být výhodné v situacích, kdy více procesů spotřebovává více prostředků než celkový prostředek používaný aplikací.Aplikace, která používá vazbu zkrácené cesty, je známá jako *důvěryhodná aplikace*.

Při rozhodování, zda použít vazbu se zkrácenou cestou, zvažte následující důležité body:

- Použití volby MQCNO_FASTPATH_BINDING nezabrání změně nebo poškození zpráv a dalších datových oblastí náležejících ke správci front. Tuto volbu používejte pouze v situacích, kdy jste tyto problémy plně vyhodnotili.
- Aplikace nesmí používat asynchronní signály nebo přerušení časovače (např. `sigkill`) s MQCNO_FASTPATH_BINDING. Existují také omezení týkající se použití segmentů sdílené paměti.
- Aplikace musí používat volání MQDISC k odpojení od správce front.
- Aplikace musí dokončit práci před ukončením správce front příkazem `endmqm` .
- IBM i V systému IBM imusí být úloha spuštěna pod profilem uživatele, který patří do skupiny `QMQMADM` . Program také nesmí být nestandardně ukončen, jinak může dojít k nepředvídatelným výsledkům.
- Linux AIX V systému AIX and Linuxmusí být identifikátor uživatele `mqm` efektivním identifikátorem uživatele a identifikátor skupiny `mqm` musí být efektivní identifikátor skupiny. Chcete-li, aby aplikace bělanila tímto způsobem, nakonfigurujte program tak, aby byl vlastněn identifikátorem uživatele `mqm` a identifikátorem skupiny `mqm` a pak nastavte bity oprávnění `setuid` a `setgid` v programu.

  Produkt IBM MQ Object Authority Manager (OAM) stále používá skutečné ID uživatele pro kontrolu oprávnění.
- Windows V systému Windowsmusí být tento program členem skupiny `mqm` . Vázání zrychleného přístupu není podporováno pro 64bitové aplikace.

Volba MQCNO_FASTPATH_BINDING je podporována v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows

z/OS V systému z/OSje volba přijata, ale je ignorována.

Další informace o důsledcích použití důvěryhodných aplikací naleznete v tématu Omezení pro důvěryhodné aplikace.

CQCNO_SHARED_BINDING

S použitím proměnné MQCNO_SHARED_BINDING se aplikace a lokální agent správce front sdílejí s některými prostředky. Objekt MQCNO_SHARED_BINDING je ignorován, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by tato volba nebyla uvedena.

VAZBA MQCNO_ISOLATED_BINDING

V tomto případě jsou procesy aplikace a lokální agent správce front izolovány od sebe navzájem, protože nesdílejí prostředky. Objekt MQCNO_ISOLATED_BINDING je ignorován, pokud správce front nepodporuje tento typ vazby. Zpracování pokračuje, jako by tato volba nebyla uvedena.

VÁZÁNÍ MQCNO_CLIENT_BINDING

Tuto volbu uveďte, chcete-li aplikaci pokusit pouze o připojení klienta. Tato volba má následující omezení:

- z/OS Parametr MQCNO_CLIENT_BINDING je v produktu z/OSignorován.
- Hodnota MQCNO_CLIENT_BINDING byla odmítnuta s chybou MQRC_OPTIONS_ERROR, je-li zadána s jinou volbou vazby MQCNO, než je MQCNO_STANDARD_BINDING.
- MQCNO_CLIENT_BINDING není k dispozici pro produkt 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 také zadán buď MQCNO_FASTPATH_BINDING, MQCNO_ISOLATED_BINDING nebo MQCNO_SHARED_BINDING, bude místo toho připojení tohoto typu a v této sekci je zdokumentováno. Jinak se provede pokus o standardní připojení k serveru s použitím výchozího typu vazby. Objekt MQCNO_LOCAL_BINDING má následující omezení:

- z/OS Parametr MQCNO_LOCAL_BINDING je v produktu z/OSignorován.
- MQCNO_LOCAL_BINDING byl odmítnut s chybou MQRC_OPTIONS_ERROR, pokud je zadán spolu s jinou volbou MQCNO reconnect jiným než MQCNO_RECONNECT_AS_DEF.
- MQCNO_LOCAL_BINDING není k dispozici pro produkt Java nebo .NET , protože mají vlastní mechanizmy pro výběr typu vazby.

Na následujících platformách můžete použít proměnnou prostředí `MQ_CONNECT_TYPE` s typem vazby určeným polem `Options` , abyste mohli řídit typ použité vazby.

- AIX AIX
- Linux Linux
- Windows Windows

Určíte-li tuto proměnnou prostředí, musí mít hodnotu FASTPATH nebo STANDARD . má-li jinou hodnotu, je ignorována. Hodnota proměnné prostředí je citlivá na velikost písmen; další informace viz proměnná prostředí MQCONNX .

Proměnná prostředí a pole *`Options`* pracují následujícím způsobem:

- Pokud vynecháte proměnnou prostředí nebo jí poskytnete hodnotu, která není podporována, je použití vazby zkrácené cesty určeno výhradně polem `Options` .
- Zadáte-li do proměnné prostředí podporovanou hodnotu, použije se vazba fastpath pouze v případě, že proměnná prostředí i pole `Options` určují vazbu se zkrácenou cestou.

Volby značek připojení

LTS 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.

V 9.2.0 Přesná implementace značek připojení se liší od IBM MQ for z/OS a IBM MQ for Multiplatforms:

- z/OS Následující volby, kromě položky *MQCNO_GENERATE_CONN_TAG*, jsou podporovány pouze při připojování ke správci front produktu z/OS a řídí použití značky připojení. Můžete určit pouze jednu z podporovaných voleb.
- ALW Objekt *MQCNO_GENERATE_CONN_TAG* je podporován pouze na platformách jiných než z/OS.

V 9.2.0 ALW **PODSOUBOR MQCNO_GENERATE_CONN_TAG**

Vrací značku připojení, kterou správce front přidružuje k tomuto připojení, ve výstupní struktuře MQCNO.

Vrácená značka připojení bude identická pro všechna připojení, která správce front považuje za jednu instanci aplikace.

z/OS **MQCNO_SERIALIZE_CONN_TAG_Q_MGR**

Tato volba vyžaduje výlučné použití značky připojení v rámci lokálního správce front. Je-li 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 níž lokální správce front patří.

z/OS **MQCNO_SERIALIZE_CONN_TAG_QSG**

Tato volba vyžaduje výlučné použití značky připojení v rámci skupiny sdílení front, do níž patří lokální správce front. Je-li značka připojení již používána 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 rámci lokálního správce front. Je-li značka připojení již používána v lokálním správci front, volání MQCONNX může být úspěšné, pokud je žádající 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 na jiném místě ve skupině sdílení front, do níž patří lokální správce front.

- Aplikace musí běžet v rámci stejného adresního prostoru MVS, aby sdílely značku připojení. Je-li aplikace používající značku připojení aplikací klienta, MQCNO_RESTRICT_CONN_TAG_Q_MGR není povolen.

z/OS **MQCNO_RESTRICT_CONN_TAG_QSG**

Tato volba vyžaduje sdílené použití značky připojení v rámci skupiny sdílení front, do níž patří lokální správce front. Je-li značka připojení již používána ve skupině sdílení front, volání MQCONNX může být úspěšné za předpokladu, že žádající aplikace bude spuštěna ve stejném rozsahu zpracování a je připojena ke stejnému správci front jako stávající uživatel značky.

Nejsou-li tyto podmínky splněny, volání MQCONNX selže s kódem příčiny MQRC_CONN_TAG_IN_USE.

- Aplikace musí běžet v rámci stejného adresního prostoru MVS, aby sdílely značku připojení. Je-li aplikace, která používá značku připojení, aplikací klienta, MQCNO_RESTRICT_CONN_TAG_QSG není povoleno.

Není-li zadána žádná z těchto voleb, `ConnTag` se nepoužije. Tyto volby nejsou platné, pokud `Version` je menší než MQCNO_VERSION_3.

Volby sdílení manipulátoru

Multi

Tyto volby jsou podporovány v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows

Ukontrolují sdílení manipulátorů mezi různými podprocesy (jednotky paralelního zpracování) v rámci stejného procesu. Můžete uvést pouze jednu z těchto voleb:

MQCNO_HANDLE_SHARE_NONE

Tato volba označuje, že připojení a popisovače objektů mohou být použity pouze podprocesem, který způsobil alokaci manipulátoru (tj. podprocesu, který vydal volání MQCONN, MQCONNX nebo MQOPEN). Popisovače nemohou být použity jinými podprocesy náležícími ke stejnému procesu.

MQCNO_HANDLE_SHARE_BLOCK

Tato volba označuje, že připojení a popisovače objektů přidělené jedním vláknem procesu mohou být použity jinými podprocesy náležícími ke stejnému procesu. Avšak pouze jedno vlákno v daném okamžiku může použít jakýkoli konkrétní popisovač; to znamená, že je povoleno pouze sériové použití ovladače. Pokud se podproces pokusí použít popisovač, který je již používán jiným podprocesem, zavolají bloky (waits), dokud nebude manipulátor dostupný.

MQCNO_HANDLE_SHARE_NO_BLOCK

Je to stejné jako MQCNO_HANDLE_SHARE_BLOCK, kromě toho, že pokud je popisovač používán jiným podprocesem, volání se dokončí okamžitě s MQCC_FAILED a MQRC_CALL_IN_PROGRESS místo blokování, dokud nebude k dispozici popisovač.

Vlákno může mít nula nebo jeden nesdílený popisovače:

- Každé volání MQCONN nebo MQCONNX, které určuje volání MQCNO_HANDLE_SHARE_NONE, vrátí nový nesdílený popisovač při prvním volání a stejný nesdílený popisovač při druhém a pozdějším volání (za předpokladu, že nezavolá žádné volání MQDISC). Kód příčiny je MQRC_ALREADY_CONNECTED pro druhou a pozdější volání.
- Každé volání MQCONNX, které určuje volání MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrací při každém volání nový sdílený popisovač.

Obslužné rutiny objektu dědí stejné vlastnosti sdílení jako manipulátor připojení určený v rámci volání MQOPEN, který vytvořil popisovač objektu. Také jednotky práce zdědí stejné vlastnosti sdílení jako popisovač připojení používaný ke spuštění jednotky práce; pokud se jednotka práce spustí v jednom podprocesu pomocí sdílené obslužné rutiny, může být pracovní jednotka aktualizována v jiném podprocesu s použitím stejného popisovače.

Nezadáte-li volbu sdílení manipulátoru, bude výchozí hodnota určena prostředím:

- Windows V prostředí MTS (Microsoft Transaction Server) 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 je připojení opětovně připojitelné. Pouze připojení klienta jsou znovu připojitelná.

MQCNO_RECONNECT_AS_DEF

Volba opětovného připojení je interpretována jako výchozí hodnota. Není-li nastavena žádná výchozí hodnota, je hodnota této volby interpretována jako DISABLED. Hodnota volby je předána na server a může být dotazována pomocí PCF a MQSC.

MQCNO_RECONNECT

Aplikace může být znovu připojena k libovolnému správci front, který je konzistentní s hodnotou parametru **QmgrName** příkazu MQCONNX. Volbu MQCNO_RECONNECT použijte pouze v případě, že neexistuje žádná afinita mezi aplikací klienta a správcem front, se kterým na počátku navázaly spojení. Hodnota volby je předána na server a může být dotazována pomocí PCF a MQSC.

FUNKCE MQCNO_RECONNECT_DISABLED

Aplikaci nelze znovu připojit. Hodnota volby není předána do serveru.

FUNKCE 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 klientskou aplikací a správcem front, s nímž původně navázala spojení. 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 na server a může být dotazována pomocí PCF a MQSC.

Použijte volby MQCNO_RECONNECT, MQCNO_RECONNECT_DISABLED a MQCNO_RECONNECT_Q_MGR pouze pro připojení klienta. Pokud se volby používají pro vázané připojení, MQCONNX selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_OPTIONS_ERROR. Automatické opětovné připojení klienta není podporováno produktem IBM MQ classes for Java

Volby sdílení konverzace

Následující volby se vztahují pouze na 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.

Můžete použít MQCNO_NO_CONV_SHARING v situacích, kdy jsou konverzace silně zatížené, a proto, je-li soupeření možností na konci spojení mezi serverem a instancí kanálu, na které se sdílení konverzací nachází. MQCNO_NO_CNV_SHARING se chová jako služba sharecnv (1) při připojení ke kanálu, který podporuje sdílení konverzace, a sharecnv (0) při připojení k kanálu, který nepodporuje sdílení konverzace.

MQCNO_ALL_CONVS_SHARE

Tato volba povoluje sdílení konverzace; aplikace nezadává žádné omezení počtu připojení na instanci kanálu. Tato volba je výchozí hodnotou.

Pokud aplikace označuje, že instance kanálu může sdílet, ale definice *SharingConversations* (SHARECNV) na konci kanálu serveru je nastavena na hodnotu jedna, nedojde ke sdílení a pro aplikaci není poskytnuta žádná výstraha.

Podobně, pokud aplikace indikuje, že je povoleno sdílení, ale definice připojení serveru *SharingConversations* je nastavena na nulu, žádné varování se neuvede a aplikace vykazuje stejné chování jako klient ve verzích starších než IBM WebSphere MQ 7.0; nastavení aplikace související se sdílením konverzací je ignorováno.

MQCNO_NO_CONV_SHARE a MQCNO_ALL_CONVS_SHARE se vzájemně vylučují. Jsou-li obě volby zadány v konkrétním připojení, je 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 umožňuje použití struktury definice kanálu v objektu MQCNO pouze k vrácení názvu kanálu použitého pro úspěšné volání MQCONNX.

Není-li poskytnuta platná struktura definice kanálu, volání selže s kódem příčiny MQRC_CD_ERROR.

Pokud aplikace není spuštěna jako klient, je tato volba ignorována.

Vrácený název kanálu lze použít při následném volání MQCONNX s použitím volby MQCNO_USE_CD_SELECTION k opětovnému připojení s použitím stejné definice kanálu. To může být užitečné v případě, že v tabulce kanálů klienta existuje více použitelných definic kanálů.

VÝBĚR MQCNO_USE_CD_SELECTION

Tato volba povoluje volání MQCONNX pro připojení s použitím názvu kanálu obsaženého ve struktuře definice kanálu předané v MQCNO.

Je-li nastavena proměnná prostředí MQSERVER, použije se definice kanálu definovaná tímto způsobem. Není-li parametr MQSERVER nastaven, použije se tabulka kanálů klienta.

Není-li nalezena definice kanálu s odpovídajícím názvem kanálu a názvem správce front, volání selže s kódem příčiny MQRC_Q_MGR_NAME_ERROR.

Není-li poskytnuta platná struktura definice kanálu, volání selže s kódem příčiny MQRC_CD_ERROR.

Pokud aplikace není spuštěna jako klient, je tato volba ignorována.

Výchozí volba

Pokud žádnou z výše uvedených voleb nevyžadujete, můžete použít následující volbu:

MQCNO_NONE

Nejsou zadány žádné volby.

Použijte MQCNO_NONE k podpoře dokumentace programu. Není určeno, že je tato volba použita s jinou volbou MQCNO_ *, ale protože její hodnota je nula, takové použití nelze zjistit.

Offset ClientConn(MQLONG)

Posunutí ClientConnje relativní ukazatel v bajtech struktury definice kanálu MQCD od začátku struktury MQCNO. Odsazení může být kladné nebo záporné. Toto pole je vstupní pole, jehož počáteční hodnota je 0.

Volbu *ClientConnOffset* používejte pouze v případě, že je aplikace, která vydala volání MQCONNX, spuštěna jako IBM MQ MQI client. Další informace o tom, jak používat toto pole, najdete v popisu pole *ClientConnPtr* .

Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_2.

ClientConnPtr (MQPTR)

ClientConnPtr je vstupní pole. Jeho počáteční hodnota je ukazatel null v těchto programovacích jazycích, které podporují ukazatele a jinak nulový bajtový řetězec s hodnotou null.

Používejte příkazy *ClientConnOffset* a *ClientConnPtr* pouze v případě, že aplikace, která volala volání MQCONNX, běží jako IBM MQ MQI client. Uvedením jednoho nebo druhého z těchto polí aplikace může řídit definici kanálu připojení klienta poskytnutím struktury definice kanálu MQCD, která obsahuje požadované hodnoty.

Je-li 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, jsou *ClientConnOffset* a *ClientConnPtr* ignorovány.

Pokud aplikace poskytuje strukturu MQCD, nastavte pole uvedená na požadované hodnoty; ostatní pole v aplikaci MQCD se budou ignorovat. Můžete vyplnit řetězce znaků s mezerami až do délky pole nebo je ukončovat znakem null. Další informace o polích ve struktuře MQCD viz “Pole” na stránce 1465 .

Tabulka 482. Pole na MQCD

Pole v MQCD	Hodnota
ChannelName	Název kanálu.
Version	Číslo verze struktury. Nesmí být menší než MQCD_VERSION_7.
TransportType	Libovolný podporovaný typ transportu.

Tabulka 482. Pole na MQCD (pokračování)

Pole v MQCD	Hodnota
ModeName	Název režimu LU 6.2 .
TpName	Název transakčního programu LU 6.2 .
SecurityExit	Název uživatelské procedury zabezpečení kanálu.
SendExit	Název uživatelské procedury odeslání kanálu.
ReceiveExit	Název uživatelské procedury příjmu kanálu.
MaxMsgLength	Maximální délka (v bajtech) zpráv, které lze odeslat přes kanál připojení klienta.
SecurityUserData	Uživatelská data pro ukončení zabezpečení.
SendUserData	Uživatelská data pro ukončení odeslání.
ReceiveUserData	Uživatelská data pro ukončení příjmu.
UserIdentifier	Identifikátor uživatele, který má být použit k vytvoření relace LU 6.2 .
Password	Heslo, které má být použito k vytvoření relace LU 6.2 .
ConnectionName	Název připojení.
HeartbeatInterval	Doba v sekundách mezi toky synchronizačních signálů.
StrucLength	Délka struktury MQCD.
ExitNameLength	Délka výstupních názvů adresovaných *SendExitPtr* a *ReceiveExitPtr*. Musí být větší než nula, je-li *SendExitPtr* nebo *ReceiveExitPtr* nastaven na hodnotu, která není ukazatelem Null.
ExitDataLength	Délka výstupních dat adresovaných *SendUserDataPtr* a *ReceiveUserDataPtr*. Musí být větší než nula, je-li *SendUserDataPtr* nebo *ReceiveUserDataPtr* nastaven na hodnotu, která není ukazatelem Null.
SendExitsDefined	Počet ukončených uživatelských procedur adresovaných produktem *SendExitPtr*. Pokud jsou nulové, *SendExit* a *SendUserData* poskytují jméno ukončení a data. Je-li větší než nula, *SendExitPtr* a *SendUserDataPtr* poskytují názvy ukončení a data, a *SendExit* a *SendUserData* musí být prázdné.
ReceiveExitsDefined	Počet uživatelských procedur pro příjem adresovaných produktem *ReceiveExitPtr*. Pokud jsou nulové, *ReceiveExit* a *ReceiveUserData* poskytují jméno ukončení a data. Je-li větší než nula, *ReceiveExitPtr* a *ReceiveUserDataPtr* poskytují názvy ukončení a data, a *ReceiveExit* a *ReceiveUserData* musí být prázdné.
SendExitPtr	Adresa jména prvního ukončení odeslání.
SendUserDataPtr	Adresa dat pro první ukončení odeslání.
ReceiveExitPtr	Adresa jména prvního ukončení příjmu.
ReceiveUserDataPtr	Adresa dat pro první uživatelskou proceduru pro příjem dat.
LongRemoteUserIdLength	Délka identifikátoru dlouhého vzdáleného uživatele.
LongRemoteUserIdPtr	Adresa dlouhého vzdáleného identifikátoru uživatele.
RemoteSecurityId	Identifikátor vzdáleného zabezpečení.
SSLCipherSpec	TLS CipherSpec.

Tabulka 482. Pole na MQCD (pokračování)

Pole v MQCD	Hodnota
SSLPeerNamePtr	Adresa názvu partnera TLS.
SSLPeerNameLength	Délka názvu partnera TLS.
KeepAliveInterval	Hodnota předaná do komunikačního zásobníku pro časování udržení aktivity pro kanál
LocalAddress	Lokální komunikační adresa, včetně adresy IP lokálního síťového adaptéru, který má být použit, a rozsah portů, které se mají použít pro odchozí připojení.

Zadejte strukturu definice kanálu jedním ze dvou způsobů:

- Použití pole offsetu *ClientConnOffset*

  V takovém případě musí aplikace deklarovat složenou strukturu obsahující MQCNO následovanou strukturou definice kanálu MQCD a nastavit hodnotu *ClientConnOffset* na posun struktury definice kanálu od začátku objektu MQCNO. Ujistěte se, že je tento posun správný. Hodnota *ClientConnPtr* musí být nastavena na nulový ukazatel nebo na null bajtů.

  Použijte *ClientConnOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele takovým způsobem, který není přenosný do různých prostředí (například programovací jazyk COBOL).

  Pro programovací jazyk Visual Basic se volá složená strukturaMQCNOCD je k dispozici v záhlaví souboru CMQXB.BAS; tato struktura obsahuje strukturu MQCNO, za kterou následuje struktura MQCD. Inicializujte MQCNOCD vyvoláním subrutiny MQCNOCD_DEFAULTS. MQCNOCD se používá spolu sMQCONNXAny pro volání MQCONNX; další podrobnosti naleznete v popisu volání MQCONNX.
- Pomocí pole ukazatele *ClientConnPtr*

  V takovém případě může aplikace deklarovat strukturu definice kanálu odděleně od struktury MQCNO a nastavit hodnotu *ClientConnPtr* na adresu struktury definice kanálu. Nastavte *ClientConnOffset* na nulu.

  Použijte *ClientConnPtr* pro programovací jazyky, které podporují datový typ ukazatele, a to způsobem, který je přenosný do různých prostředí (například programovací jazyk C).

  V programovacím jazyce C lze pomocí proměnné makra MQCD_CLIENT_CONN_DEFAULT zadat počáteční hodnoty struktury, které jsou vhodnější pro použití v rámci volání MQCONNX než počáteční hodnoty poskytnuté parametrem MQCD_DEFAULT.

Vámi zvolený způsob, jak vybrat, můžete použít pouze jeden z *ClientConnOffset* a *ClientConnPtr* ; volání selže s kódem příčiny MQRC_CLIENT_CONN_ERROR, pokud jsou obě tyto hodnoty nenulová.

Po dokončení volání MQCONNX se na strukturu MQCD již znovu neodkazuje.

Toto pole je ignorováno, pokud *Version* je 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 hodnotou je řetězec bajtů se všemi znaky null.

V 9.2.0 *ConnTag (MQBYTE128) on Multiplatforms*

Značka připojení je koncepčně podobná identifikátoru připojení, ale může zahrnovat více souvisejících připojení a identifikovat je jako jedinou instanci aplikace. Na platformách 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 je lze zobrazit a filtrovat v prostředí MQSC, jako by řetězce v lokální znakové sadě. Připojení, která jsou známa produktem IBM MQ

jako související, mají automaticky přiřazenou stejnou značku připojení. Toto přiřazení je obzvláště důležité pro vyvažování aplikací.

Generovaná značka připojení je viditelná třemi způsoby:

- Ve výstupní struktuře MQCNO v rámci volání MQCONNX, je-li zadán parametr PODSOUBOR MQCNO_GENERATE_CONN_TAG .
- Na výstupu příkazu DISPLAY CONN (nebo programové ekvivalenty).
- Ve výstupu příkazu DISPLAY APSTATUS (nebo ekvivalenty).

Tato značka přestane být platná při ukončení aplikace nebo při vyvolání volání MQDISC.

Související odkazy

"ConnTag (MQBYTE128) na systému 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) na systému 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) on Multiplatforms" na stránce 332 .

SSLConfigPtr (PMQSCO)

SSLConfigPtr je vstupní pole. Jeho počáteční hodnota je ukazatel null v těchto programovacích jazycích, které podporují ukazatele a jinak nulový bajtový řetězec s hodnotou null.

Používejte příkazy *SSLConfigPtr* a *SSLConfigOffset* pouze v případě, že je spuštěna aplikace, která vydala volání MQCONNX, jako IBM MQ MQI client a protokol kanálu je TCP/IP. Pokud aplikace není spuštěna jako klient produktu IBM MQ , nebo pokud není protokol kanálu TCP/IP, jsou ignorovány parametry *SSLConfigPtr* a *SSLConfigOffset* .

Uvedením *SSLConfigPtr* nebo *SSLConfigOffset*, plus buď *ClientConnPtr* nebo *ClientConnOffset*, aplikace může řídit použití TLS pro připojení klienta. Když jsou informace TLS

uvedeny tímto způsobem, proměnné prostředí MQSSLKEYR a MQSSLCRYP se ignorují; všechny informace související s TLS v tabulce definic kanálů klienta (CCDT) se také ignorují.

Informace TLS mohou být uvedeny pouze na:

- První volání MQCONNX procesu typu klient, nebo
- Následný volání MQCONNX, když byla všechna předchozí připojení TLS ke správci front uzavřena pomocí MQDISC.

Toto jsou jediné stavy, v nichž lze inicializovat prostředí TLS v rámci celého procesu. Je-li vydáno volání MQCONNX s uvedením informací TLS, když prostředí TLS již existuje, informace TLS na volání se budou ignorovat a spojení se provede s použitím existujícího prostředí TLS; volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SSL_ALREADY_INITIALIZED v tomto případě.

Strukturu MQSCO můžete zadat stejným způsobem jako strukturu MQCD, a to buď zadáním adresy v produktu *SSLConfigPtr*, nebo zadáním posunu v souboru *SSLConfigOffset* . Podrobnosti o tom, jak to provést, naleznete v popisu *ClientConnPtr* . Avšak nemůžete použít více než jeden z *SSLConfigPtr* a *SSLConfigOffset* ; volání selže s kódem příčiny MQRC_SSL_CONFIG_ERROR. jsou-li nenulová hodnota.

Jakmile je volání MQCONNX dokončeno, struktura MQSCO již není znovu odkazována.

Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_4.

Poznámka: Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

SSLConfigOffset (MQLONG)

SSLConfigOffset je posun v bajtech struktury MQSCO od začátku struktury MQCNO. Odsazení může být kladné nebo záporné. Toto pole je vstupní pole, s počáteční hodnotou 0.

Volbu *SSLConfigOffset* používejte pouze v případě, že je aplikace, která vydala volání MQCONNX, spuštěna jako IBM MQ MQI client. Další informace o tom, jak používat toto pole, najdete v popisu pole *SSLConfigPtr* .

Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_4.

ConnectionId (MQBYTE24)

ConnectionId je jedinečný 24bajtový identifikátor, který umožňuje IBM MQ spolehlivě identifikovat aplikaci. Aplikace může použít tento identifikátor pro korelaci v voláních PUT a GET. Tento výstupní parametr má počáteční hodnotu 24 bajtů s hodnotou null ve všech programovacích jazycích.

Správce front přiřadí jedinečné ID ke všem připojením, jsou však ustanovená. Pokud MQCONNX vytvoří připojení s verzí 5 MQCNO, může aplikace určit vlastnost ConnectionId z vráceného objektu MQCNO. Přiřazenému identifikátoru je zaručeno, že je jedinečný mezi všemi ostatními identifikátory, které generuje produkt IBM MQ , jako např. CorrelId, MsgIDa GroupId.

Použijte ConnectionId k identifikaci dlouho běžících jednotek práce pomocí příkazu PCF pro zjišťování spojení nebo příkazu MQSC DISPLAY CONN. Hodnota ConnectionId použitá příkazy MQSC (CONN) je odvozena z hodnoty ConnectionId vrácené zde. Příkazy PCF Inquire a Stop Connection mohou používat identifikátor ConnectionId , který je zde bez úprav vrácen.

Pomocí ConnectionId můžete vynutit ukončení přerušitelné jednotky práce uvedením ConnectionId pomocí příkazu k zastavení připojení příkazu PCF nebo příkazu MQSC STOP CONN. Další informace o použití těchto příkazů najdete v tématech Zastavit připojení a STOP CONN .

Toto pole není vráceno, pokud je verze nižší než MQCNO_VERSION_5.

Délka tohoto pole je dána hodnotou MQ_CONNECTION_ID_LENGTH.

Offset SecurityParms(MQLONG)
SecurityParmsPosunutí je relativní ukazatel v bajtech struktury MQCSP od začátku struktury MQCNO. Odsazení může být kladné nebo záporné. Toto pole je vstupní pole, s počáteční hodnotou 0.

Toto pole se ignoruje, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v produktu "MQCSP-parametry zabezpečení" na stránce 336.

SecurityParmsPtr (PMQCSP)
SecurityParmsPtr je adresa struktury MQCSP, která se používá k zadání ID uživatele a hesla pro ověření pomocí autorizační služby. Toto pole je vstupním polem a jeho počáteční hodnotou je ukazatel null nebo null bajtů.

Toto pole se ignoruje, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v produktu "MQCSP-parametry zabezpečení" na stránce 336.

Rezervováno (MQBYTE4)
Vyhrazené pole pro vyplnění struktury na 64-bitovou hranici. Počáteční hodnota pole je binární nula pro délku pole.

Toto pole je ignorováno, pokud `Version` je menší než MQCNO_VERSION_6.

CCDTUrlLength (MQLONG)
CCDTUrlLength je délka řetězce identifikovaného pomocí příkazu CCDTUrlPtr nebo CCDTUrlOffset , který obsahuje adresu URL identifikující umístění tabulky kanálů 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)
CCDTUrlPtr je volitelný ukazatel na řetězec, který obsahuje adresu URL pro 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)
CCDTUrlOffset je posun v bajtech od začátku struktury MQCNO k řetězci, který obsahuje adresu URL identifikující umístění tabulky kanálu 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.

V 9.2.0 *ApplName (MQCHAR28)*

Název nastavený aplikací pro identifikaci připojení ke správci front. Počáteční hodnota pole je MQAN_NONE_ARRAY (prázdné znaky).

Toto pole je ignorováno, pokud `Version` je menší než MQCNO_VERSION_7, nebo pokud je hodnota nastavena na mezery.

z/OS Toto pole nemůžete nastavit na z/OS. Pokusíte-li se tak učinit, obdržíte zpět kód příčiny MQRC_CNO_ERROR.

V 9.2.0 *Reserved2 (MQBYTE4)*

Vyhrazené pole pro vyplnění struktury na 64-bitovou hranici. Počáteční hodnota pole je binární nula pro délku pole.

Toto pole je ignorováno, pokud `Version` je menší než MQCNO_VERSION_7.

V 9.2.4 *Offset BalanceParms(MQLONG)*

Umístění paměti pro strukturu typu MQBNO, které obsahuje informace o vyvažování chování aplikace. Struktura je zcela ignorována, pokud se aplikace nepřipojuje přes kanál klienta.

Toto pole je ignorováno, pokud `Version` je menší než MQCNO_VERSION_8.

Další informace viz MQBNO .

Zadáte-li toto pole, nebude možné zadat pole “BalanceParmsPtr (MQPTR)” na stránce 336 . Pokusíte-li se zadat obě pole, obdržíte chybovou zprávu MQRC_CNO_ERROR. Jelikož je toto pole relevantní pouze pro klientská připojení, zadání tohoto pole na libovolném jiném typu připojení má za následek také funkci MQRC_CNO_ERROR.

V 9.2.4 *BalanceParmsPtr (MQPTR)*

Ukazatel na umístění paměti pro strukturu typu MQBNO, která obsahuje informace o vyvažování chování aplikace. Struktura je zcela ignorována, pokud se aplikace nepřipojuje přes kanál klienta.

Toto pole je ignorováno, pokud `Version` je menší než MQCNO_VERSION_8.

Další informace viz MQBNO .

Zadáte-li toto pole, nebude možné zadat pole “Offset BalanceParms(MQLONG)” na stránce 336 . Pokusíte-li se zadat obě pole, obdržíte chybovou zprávu MQRC_CNO_ERROR. Jelikož je toto pole relevantní pouze pro klientská připojení, zadání tohoto pole na libovolném jiném typu připojení má za následek také funkci MQRC_CNO_ERROR.

MQCSP-parametry zabezpečení

Struktura MQCSP umožňuje autorizační službě ověřit ID uživatele a heslo. Strukturu parametrů zabezpečení připojení MQCSP zadáváte ve volání MQCONNX.

Varování: V některých případech je heslo ve struktuře MQCSP pro klientskou aplikaci odesláno po síti jako prostý text. Chcete-li zajistit odpovídající ochranu hesel klientských aplikací, přečtěte si téma Ochrana hesel MQCSP.

Dostupnost

Struktura MQCSP je k dispozici na všech podporovaných platformách IBM MQ .

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 483. Pole v 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
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

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

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
  MQCHAR4    StrucId;            /* Structure identifier */
  MQLONG     Version;            /* Structure version number */
  MQLONG     AuthenticationType; /* Type of authentication */
  MQBYTE4    Reserved1;          /* Required for IBM i pointer
                                    alignment */
  MQPTR      CSPUserIdPtr;       /* Address of user ID */
  MQLONG     CSPUserIdOffset;    /* Offset of user ID */
```

```
  MQLONG     CSPUserIdLength;    /* Length of user ID */
  MQBYTE8    Reserved2;          /* Required for IBM i pointer
                                    alignment */
                                 alignment */
  MQPTR      CSPPasswordPtr;     /* Address of password */
  MQLONG     CSPPasswordOffset;  /* Offset of password */
  MQLONG     CSPPasswordLength;  /* Length of password */
  };
```

Deklarace jazyka COBOL pro MQCSP

```
**   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.
```

Deklarace PL/I pro MQCSP

```
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 */
```

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
```

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota musí být:

ID_STRUKTURY MQCSP_STRUCT
Identifikátor struktury parametrů zabezpečení.

Pro programovací jazyk C je také definován konstantní MQCSP_STRUC_ID_ARRAY; má stejnou hodnotu jako MQCSP_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCSPSTRUC_ID.

Verze (MQLONG)
Číslo verze struktury.

Hodnota musí být:

MQCSP_VERSION_1
Struktura parametrů zabezpečení Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCSP_CURRENT_VERSION
Aktuální verze struktury parametrů zabezpečení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCSP_VERSION_1.

AuthenticationType (MQLONG)
AuthenticationType je vstupní pole. Jeho počáteční hodnota je MQCS_AUTH_NONE.

Jedná se o typ ověření, které se má provést. Platné jsou tyto hodnoty:

MQCSP_AUTH_NONE
Nepoužívejte pole ID uživatele a heslo.

MQCSP_AUTH_USER_ID_AND_PWD
Ověřte ID uživatele a pole hesel.

Výchozí hodnota je MQCS_AUTH_NONE. Při výchozím nastavení není ochrana heslem provedena.

Pokud vyžadujete ověření, musíte nastavit **MQCSP**.**AuthenticationType** pro MQCSP_AUTH_USER_ID_AND_PWD.

Další informace naleznete v tématu Ochrana heslem MQCSP .

Reserved1 (MQBYTE4)
Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

CSPUserIdPtr (MQPTR)
Jedná se o adresu v bajtech ID uživatele, které má být použito pro ověření.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_5.

Toto pole může obsahovat ID uživatele operačního systému, je-li **AUTHTYPE** *IDPWOS* zadáno v poli CONNAUTH správce front.

V systému Windows může jít o plně kvalifikované ID uživatele domény.

Toto pole může obsahovat ID uživatele LDAP, je-li **AUTHTYPE** *IDPWLDAP* pojmenováno v poli CONNAUTH správce front.

Offset CSPUserId(MQLONG)
Jedná se o ofset v bajtech ID uživatele, které se má použít při ověření. Odsazení může být kladné nebo záporné.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSPUserIdDélka (MQLONG)
Toto pole je délka ID uživatele, které se má použít při ověření.

Maximální délka ID uživatele je závislá na platformě, viz ID uživatelů. Je-li délka ID uživatele větší než maximální povolená délka, požadavek na ověření selže s hodnotou MQRC_NOT_AUTHORIZED.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je 0.

Reserved2 (MQBYTE8)
Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

CSPPasswordPtr (MQPTR)
Jedná se o adresu v bajtech hesla, které má být použito v ověření.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_5.

Toto pole může obsahovat prázdné heslo, které bylo zamítnuto operačním systémem nebo kontrolou hesla LDAP, v závislosti na nastavení, ale IBM MQ ji před předáním metody ověření odmítne.

CSPPasswordOffset (MQLONG)
Toto je posun v bajtech hesla, které má být použito při ověření. Odsazení může být kladné nebo záporné.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSPPasswordLength (MQLONG)
Toto pole je délka hesla, které se má použít při ověření.

Maximální délka hesla je MQ_CSP_PASSWORD_LENGTH, což je 256 znaků. Je-li délka hesla větší než maximální povolená délka, požadavek na ověření selže s hodnotou MQRC_NOT_AUTHORIZED.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je 0.

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 484. 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)

Struktura voleb ovládacích prvků-pole StrucId

Jedná se o identifikátor struktury; hodnota musí být:

MQCTLO_STRUCTION_ID
Identifikátor pro strukturu voleb ovládacích prvků.

Pro programovací jazyk C je také definován konstantní MQCTLO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQCTLO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCTLO_STRUC_ID.

Verze (MQLONG)

Struktura voleb ovládacího prvku-pole Verze

Jedná se o číslo verze struktury; hodnota musí být:

MQCTLO_VERSION_1
Version-1 Struktura voleb řízení.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ VERZE MQCTLO_VERSION
Aktuální verze struktury voleb řízení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQCTLO_VERSION_1.

Volby (MQLONG)

Struktura voleb ovládacího prvku-pole Volby

Volby, které řídí akci MQCTL.

UVÁDĚNÍ MQCTLO_FAIL_IF_QUIESCING
Vynuťte selhání volání funkce MQCTL, je-li správce front nebo připojení ve stavu uvedení do klidového stavu.

Určete MQGMO_FAIL_IF_QUIESCING, v rámci voleb MQGMO předaných volání MQCB, aby bylo oznámení uživatelům oznámeno, když je uváděno do klidového stavu.

MQCTLO_THREAD_AFFINITY
Tato volba informuje systém o tom, že aplikace vyžaduje, aby všichni spotřebitelé zpráv, pro stejné připojení, byli volání na stejném podprocesu. Tento podproces bude použit pro všechna vyvolání spotřebitelů, dokud nebude připojení zastaveno.

Výchozí volba: Pokud nepotřebujete žádné 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 pro dokumentaci programu podpory; není určeno, aby byla tato volba použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze zjistit.

Toto je vstupní pole. Počáteční hodnota pole *Options* je MQCTLO_NONE.

Rezervováno (MQLONG)

Jedná se o vyhrazené pole. Hodnota musí být nula.

ConnectionArea (MQPTR)

Struktura voleb ovládacího prvku-pole ConnectionArea

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front neprovádí žádná rozhodnutí založená na obsahu tohoto pole a je beze změny do pole ConnectionArea ve struktuře MQCBC, což je vstupní parametr pro zpětné volání.

Toto pole je ignorováno pro všechny operace jiné než MQOP_START a MQOP_START_WAIT.

Jedná se o vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null 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).

Použití

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 827. 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 485. 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
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

Tabulka 485. Pole v MQDH pro MQDH (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.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)

Hodnota musí být:

ID_STRUKTURY MQDH_
Identifikátor pro strukturu záhlaví distribuce.

Pro programovací jazyk C je také definována konstanta MQDH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQDH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQDH_STRUC_ID.

Verze (MQLONG)

Hodnota musí být:

MQDH_VERSION_1
Číslo verze pro strukturu záhlaví distribuce.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQDH_CURRENT_VERSION
Aktuální verze struktury záhlaví distribuce.

Počáteční hodnota tohoto pole je MQDH_VERSION_1.

StrucLength (MQLONG)

Jedná se o počet bajtů od začátku struktury MQDH do začátku dat zprávy za pole záznamů MQOR a MQPMR. Data se objevují 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 jedním nebo více strukturou MQDH, poli záznamů a daty zprávy, tyto nepoužívané bajty musí být zahrnuty do hodnoty *StrucLength*, ale obsah těchto bajtů není správcem front zachován. Je platný pro pole záznamů MQPMR, aby bylo před polem záznamů MQOR předcházet.

Počáteční hodnota tohoto pole je 0.

Kódování (MQLONG)

Jedná se o číselné kódování dat, která jsou uvedena v polích záznamů MQOR a MQPMR; nevztahuje se na číselná data ve struktuře MQDH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG)

Jedná se o identifikátor znakové sady dat, která jsou uvedena v polích záznamů MQOR a MQPMR; nevztahuje se na znaková data ve struktuře MQDH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Můžete použít následující speciální hodnotu:

MQCSI_INHERIT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Pokud se nevyskytne žádná chyba, volání MQGET nevrátí hodnotu MQCCSI_INHERIT.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru 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)

Jedná se o název formátu dat, která následují za pole záznamů MQOD a MQPMR (podle toho, co nastane dříve).

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole *Format* v produktu MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

Můžete zadat následující příznak:

MQDHF_NEW_MSG_ID
Generujte nový identifikátor zprávy pro každé místo určení v rozdělovníku. Nastavte jej pouze v případě, že nejsou přítomny žádné záznamy vložení zpráv, nebo jsou-li záznamy přítomné, ale neobsahují pole *MsgId* .

Použití tohoto parametru deferuje generování identifikátorů zpráv až do chvíle, kdy je zpráva distribučního seznamu konečně rozdělena na jednotlivé zprávy. Tím se minimalizuje množství řídicích informací, které musí tok obsahovat zprávu distribuční seznam.

Když aplikace vloží zprávu do distribučního seznamu, správce front nastaví MQDHF_NEW_MSG_IDS v objektu MQDH, který vygeneruje, když jsou obě následující příkazy pravdivé:

- K dispozici nejsou žádné záznamy vložení zpráv poskytnuté aplikací nebo zadané záznamy neobsahují pole *MsgId* .
- Pole *MsgId* v MQMD je MQMI_NONE, nebo pole *Options* v MQPMO zahrnuje MQPMO_NEW_MSG_ID

Nejsou-li vyžadovány žádné příznaky, zadejte následující:

MQDHF_NONE
Nebyly zadány žádné parametry. Objekt MQDHF_NONE je definován v dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je MQDHF_NONE.

PutMsgRecFields (MQLONG)

Můžete určit žádný nebo více z následujících příznaků:

MQPMRF_ID_ZPRÁVY
Zobrazí se pole identifikátoru zprávy.

MQPMRF_CORREL_ID
Pole identifikátoru korelace je přítomno.

ID SKUPINY MQPMRF_GROUP_ID
Pole identifikátoru skupiny je přítomno.

ZPĚTNÁ VAZBA MQPMRF_FEEDBACK
Je přítomno pole zpětné vazby.

MQPMRF_ACCOUNTING_TOKEN
Pole Účetní-token je přítomno.

Pokud nejsou přítomna žádná pole MQPMR, zadejte následující:

MQPMRF_NONE
Nejsou přítomna žádná pole záznamu vložení zprávy. Funkce MQPMRF_NONE je definována pro dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je MQPMRF_NONE.

RecsPresent (MQLONG)

Toto je počet míst určení. Rozdělovník musí vždy obsahovat alespoň jedno místo určení, takže *RecsPresent* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

Posunutí ObjectRec(MQLONG)

To dává offsetu v bajtech prvního záznamu v poli záznamů objektů MQOR, které obsahují 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 zadané v poli *StrucLength* .

Rozdělovník musí vždy obsahovat alespoň jedno místo určení, takže *ObjectRecOffset* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

PutMsgRecOffset (MQLONG)

To dává odchylku v bajtech prvního záznamu v poli záznamů zpráv MQPMR, které obsahují vlastnosti zprávy. Je-li přítomen, v tomto poli jsou záznamy *RecsPresent* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem vložení zprávy a předchozím polem) jsou zahrnuty do délky zadané v poli *StrucLength* .

Záznamy vložení 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 .

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 pro vložení do fronty nedoručených zpráv, aplikace musí provést jednu z následujících možností:

- Ořízněte 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 486. 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

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)

StrucId je identifikátor struktury.

Hodnota musí být:

ID_STRUKTURY MQDLH_STRUCTION_ID
Identifikátor pro strukturu záhlaví s dead-letter.

Pro programovací jazyk C je také definována konstanta MQDLH_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQDLH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQDLH_STRUC_ID.

Verze (MQLONG)
Verze je číslo verze struktury.

Hodnota musí být:

MQDLH_VERSION_1
Číslo verze pro strukturu záhlaví dead-letter.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQDLH_CURRENT_VERSION
Aktuální verze struktury záhlaví dead-letter.

Počáteční hodnota tohoto pole je MQDLH_VERSION_1.

Příčina (MQLONG)
Pole Příčina identifikuje důvod, proč byla zpráva umístěna do fronty nedoručených zpráv místo na původní cílové frontě.

To identifikuje důvod, proč byla zpráva umístěna do fronty nedoručených zpráv místo na původní cílové frontě. Mělo by se jednat o jednu z hodnot MQFB_ * nebo MQRC_ * (například MQRC_Q_FULL). Podrobné informace o obecných hodnotách MQFB_ *, které se mohou vyskytnout, najdete v popisu pole *Feedback* v příručce “MQMD-Deskriptor zpráv” na stránce 419 .

Je-li hodnota v rozsahu MQFB_IMS_FIRST až MQFB_IMS_LAST, skutečný kód chyby IMS může být určen odečtením MQFB_IMS_ERROR od hodnoty pole *Reason* .

Některé hodnoty MQFB_ * se vyskytují pouze v tomto poli. Souvisí s zprávami úložiště, spouštěcími zprávami nebo zprávami přenosové fronty, které byly přeneseny do fronty nedoručených zpráv. Patří mezi ně:

Objekt MQFB_APPL_CANNOT_BE_STARTED (X'00000109')
Zpracování aplikace, které zpracovává spouštěcí zprávu, nemůže spustit aplikaci uvedenou v poli *ApplId* zprávy spouštěče (viz “MQTM-zpráva spouštěče” na stránce 594).

V systému z/OSje transakce CKTI CICS příkladem aplikace, která zpracovává zprávy spouštěče.

MQFB_APPL_TYPE_ERROR (X'0000010B')
Zpracování žádosti o spouštěcí zprávu aplikace nemůže spustit aplikaci, protože pole *ApplType* zprávy spouštěče je neplatné (viz “MQTM-zpráva spouštěče” na stránce 594).

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 určená pro frontu klastru, která byla otevřena pomocí volby MQOO_BIND_ON_OPEN, ale vzdálený kanál příjemce klastru, který má být použit k přenosu zprávy do cílové fronty, byl odstraněn dříve, než bylo možné zprávu odeslat. Protože byla zadána hodnota 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, bude zpráva umístěna do fronty nedoručených zpráv.

MQFB_NOT_A_REPOSITORY_MSG (X'00000118')
Zpráva není zprávou úložiště.

Funkce 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 zprávy kanálu.

MQFB_TM_ERROR (X'0000010A')
Pole *Format* v MQMD určuje MQFMT_TRIGGER, ale zpráva nezačíná platnou strukturou MQTM. Například mnemonika *StrucId* může být neplatná, *Version* nemusí být rozpoznána, nebo může být délka zprávy spouštěče nedostatečná k tomu, aby mohla obsahovat 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 oznamovacího kanálu umístí zprávu do fronty nedoručených zpráv pomocí tohoto kódu zpětné vazby.

Jedna společná příčina 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)
DestQName je název fronty zpráv, která byla původním cílem pro zprávu.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název DestQMgr(MQCHAR48)
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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Kódování (MQLONG)
Kódování je číselné kódování dat, která se řídí strukturou MQDLH (obvykle data z původní zprávy). Nevztahuje se na číselná data ve struktuře MQDLH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG)
CodedCharSetId je identifikátor znakové sady dat, která teče přes strukturu MQDLH (obvykle data z původní zprávy). Nepoužívá se na znaková data ve struktuře MQDLH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech po této struktuře jsou ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru 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)
Formát je název formátu dat, která následují za strukturou MQDLH (obvykle data z původní zprávy).

Na základě 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 produktu MQMD.

Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Typ PutAppl(MQLONG)
PutApplTyp je typ aplikace, která vložila zprávu do fronty nedoručených zpráv (undelivered-message).

Toto pole má stejný význam jako pole *PutApplType* v deskriptoru zpráv MQMD (podrobnosti viz "MQMD-Deskriptor zpráv" na stránce 419).

Pokud správce front přesměrovává zprávu do fronty nedoručených zpráv, bude mít parametr *PutApplType* hodnotu MQAT_QMGR.

Počáteční hodnota tohoto pole je 0.

Název funkce PutAppl(MQCHAR28)
PutApplName je název aplikace, která vložila zprávu do fronty nedoručených zpráv (undelivered-message).

Formát názvu závisí na poli *PutApplType* . Formát se může lišit od verze k vydání. Viz popis pole *PutApplName* v "MQMD-Deskriptor zpráv" na stránce 419.

Pokud správce front přesměrovává zprávu do fronty nedoručených zpráv, obsahuje *PutApplName* prvních 28 znaků názvu správce front a je-li to nutné, doplní se mezerami.

Délka tohoto pole je dána hodnotou MQ_PUT_APPL_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 28 prázdných znaků v jiných programovacích jazycích.

PutDate (MQCHAR8)
PutDate je datum, kdy byla zpráva vložena do fronty nedoručených zpráv (undelivered-message).

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- YYYYMMDD

kde znaky představují:

yyyy
rok (čtyři číselné číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Čas GMT (Greenwich Mean Time) se používá pro pole *PutDate* a *PutTime* , přičemž se použijí systémové hodiny přesně nastavené na GMT.

Délka tohoto pole je dána hodnotou MQ_PUT_DATE_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v jazyce C a osm prázdných znaků v jiných programovacích jazycích.

PutTime (MQCHAR8)
PutTime je čas, kdy byla zpráva vložena do fronty nedoručených zpráv (undelivered-message).

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSTH

kde znaky představují:

HH
hodin (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: Je-li časová základna systému synchronizována s velmi přesným časovým standardem, je možné ve vzácných případech vrátit hodnotu 60 nebo 61 pro sekundy v produktu *PutTime*. To se stane, když se do globálního časového standardu vloží přestupné sekundy.

Čas GMT (Greenwich Mean Time) se používá pro pole *PutDate* a *PutTime* , přičemž se použijí systémové hodiny přesně nastavené na GMT.

Délka tohoto pole je dána hodnotou MQ_PUT_TIME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null 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 487. 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)

Jedná se o identifikátor struktury; hodnota musí být:

MQDMHO_STRUCTION_ID

Identifikátor pro strukturu voleb pro zpracování odstranění zprávy.

Pro programovací jazyk C je také definována konstanta **MQDMHO_STRUC_ID_ARRAY** ; tato hodnota má stejnou hodnotu jako **MQDMHO_STRUC_ID**, ale je pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQDMHO_STRUC_ID**.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQDMHO_VERSION_1
Version-1 -odstranění struktury voleb zpracování zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQDMHO_CURRENT_VERSION
Aktuální verze struktury voleb pro zpracování odstranění zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQDMHO_VERSION_1**.

Volby (MQLONG)

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 488. 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)

Struktura voleb pro odstranění vlastností zprávy-pole StrucId

Jedná se o identifikátor struktury. Hodnota musí být:

ID_KONSTRUKCE_MQDMPO_
Identifikátor pro strukturu voleb vlastností odstranění zprávy.

Pro programovací jazyk C je také definován konstantní MQDMPO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQDMPO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQDMPO_STRUC_ID.

Verze (MQLONG)

Struktura volby odstranění vlastností zprávy-pole Verze

Jedná se o číslo verze struktury. Hodnota musí být:

MQDMPO_VERSION_1
Číslo verze pro strukturu voleb vlastností odstranění zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

MQDMPO_AKTUÁLNÍ_VERZE
Aktuální verze struktury voleb pro odstranění vlastností zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQDMPO_VERSION_1.

Volby (MQLONG)

Struktura voleb odstranění vlastností zprávy-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 uvedenému názvu.

MQDMPO_DEL_PROP_UNDER_CURSOR
Odstraní vlastnost, na kterou ukazuje kurzor vlastností. Jedná se o vlastnost, která byla naposledy dotazovaná pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy. Je také resetováno, je-li popisovač zprávy určen v poli *MsgHandle* struktury MQGMO na volání MQGET nebo MQPMO na volání MQPUT.

Je-li tato volba použita v situaci, kdy kurzor vlastnosti ještě nebyl vytvořen, volání se nezdaří s kódem dokončení MQCC_FAILED a příčinou je MQRC_PROPERTY_NOT_AVAILABLE. Pokud byla vlastnost, na kterou ukazuje kurzor vlastnosti, již odstraněna, volání také selže s kódem dokončení MQCC_FAILED a příčinou je MQRC_PROPERTY_NOT_AVAILABLE.

Není-li požadována žádná z voleb thees, lze použít následující volbu:

MQDMPO_NONE
Nejsou uvedeny žádné volby.

Toto pole je vždy vstupním polem. 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).

Pouľití

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 489. Pole v MQEPH pro MQEPH

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	MQEPH_STRUC_ID	'EPH¬'

Tabulka 489. Pole v MQEPH pro MQEPH (pokračování)

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
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 490 na stránce 365	0

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)

Hodnota musí být:

ID_KONSTRUKCE_MQEPH_
Identifikátor pro strukturu záhlaví distribuce.

Pro programovací jazyk C je také definována konstanta MQEPH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQDH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQEPH_STRUC_ID.

Verze (MQLONG)

Hodnota musí být:

MQEPH_VERSION_1
Číslo verze pro vloženou strukturu záhlaví PCF.

Následující konstanta uvádí číslo verze aktuální verze:

MQCFH_VERSION_3
Aktuální verze vestavěné struktury záhlaví PCF.

Počáteční hodnota tohoto pole je MQEPH_VERSION_1.

StrucLength (MQLONG)

Jedná se o množství dat, která předchází další struktuře záhlaví. Zahrnuje:

- Délka záhlaví MQEPH
- Délka všech parametrů PCF za záhlavím
- Jakákoli prázdná výplň za těmito parametry

Hodnota StrucLength musí být násobkem 4.

Část struktury pevné délky je definována proměnnou MQEPH_STRUC_LENGTH_FIXED.

Počáteční hodnota tohoto pole je 68.

Kódování (MQLONG)

Jedná se o číselné kódování dat, která se řídí strukturou MQEPH a s přiřazovanými parametry PCF; nepoužívá se pro znaková data ve struktuře MQEPH.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG)

Jedná se o identifikátor znakové sady dat, která následuje strukturu MQEPH a přidružené parametry PCF; nepoužívá se pro znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8)

Jedná se o název formátu dat, která se řídí strukturou MQEPH a s přidruženými parametry PCF.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

K dispozici jsou tyto hodnoty:

MQEPH_NONE
Nebyly zadány žádné parametry. Funkce MQEPH_NONE je definována pro dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

VLOŽKA MQEPH_CCSID_EMBEDDED
Znaková sada parametrů, které obsahují znaková data, se zadává jednotlivě v poli CodedCharSetId v každé struktuře. Znaková sada polí StrucId a Format je definována polem CodedCharSetId ve struktuře záhlaví, která předchází struktuře MQEPH, nebo pole CodedCharSetId v MQMD, pokud je MQEPH na začátku zprávy.

Počáteční hodnota tohoto pole je MQEPH_NONE.

Záhlaví PCFHeader (MQCFH)

Jedná se o záhlaví PCF (Programmable command format) definující parametry PCF, které se řídí strukturou MQEPH. To vám umožní sledovat data zprávy PCF s ostatními záhlavími.

Hlavička PCF je na počátku definována s následujícími hodnotami:

Tabulka 490. Počáteční hodnoty polí v MQCFH

Název pole	Název konstanty	Hodnota konstanty
Type	MQCFT_NONE	0
StrucLength	DÉLKA OBJEKTU MQCFH_STRU_LENGTH	36
Version	MQCFH_VERSION_3	3
StrucLength	Není	0
Command	MQCMD_NONE	0
MsgSeqNumber	Není	1
Control	MQCFC_LAST	1

Tabulka 490. Počáteční hodnoty polí v MQCFH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	Není	0

Aplikace musí změnit Type z MQCFT_NONE na platný typ struktury pro použití 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 366 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 491. 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

Tabulka 491. Pole v MQGMO pro MQGMO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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
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 9.2 partnerskému správci front ze starší verze produktu IBM MQ.

Tabulka 492. 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 9.2 . 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 9.2 . 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 9.2 a dřívější verzí s použitím kanálu. Nové aplikace IBM MQ 9.2 používající vlastnosti zpráv mohou spolupracovat s aplikacemi ze starší verze v závislosti na způsobu, jakým aplikace starší verze zpracovává záhlaví MQRFH2 .

Tabulka 492. Nastavení atributu vlastnosti zprávy kanálu (pokračování)

PROPCTL	**Popis**
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 9.2 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 9.2 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.
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 9.2 , 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 493. 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 493. 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 9.2. 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 9.2 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 9.2 . 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 9.2 . 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 9.2 . 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 493. 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 9.2 , 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 9.2, 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 “Data NameValueData (MQCHARn)” na stránce 530.

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 494. 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 494. 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)

Jedná se o identifikátor struktury. Hodnota musí být:

MQGMO_STRUCTION_ID
Identifikátor pro strukturu voleb get-message.

Pro programovací jazyk C je také definována konstanta MQGMO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQGMO_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQGMO_STRUC_ID.

Verze (MQLONG)

Verze je číslo verze struktury.

Hodnota musí být jedna z následujících:

MQGMO_VERSION_1
Struktura volby get-message pro objekt Version-1 .

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_2
Struktura volby get-message pro objekt Version-2 .

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_3
Struktura volby get-message pro objekt Version-3 .

Tato verze je podporována ve všech prostředích.

MQGMO_VERSION_4
Struktura volby get-message pro objekt Version-4 .

Tato verze je podporována ve všech prostředích.

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQGMO_VERSION
Aktuální verze struktury voleb získání zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQGMO_VERSION_1.

Volby (MQLONG) pro MQGMO

Volby produktu **MQGMO** řídí akci produktu MQGET. Můžete uvést nulu nebo více voleb. Potřebujete-li 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 zaznamenány; všechny ostatní kombinace jsou platné.

Volby čekání

Následující volby se vztahují k čekání na příchod zpráv do fronty:

MQGMO_WAIT
Aplikace čeká, dokud nepřijde vhodná zpráva. Maximální doba, po kterou aplikace čeká, je určena v produktu *WaitInterval* .

Důležité: Pokud je okamžitě k dispozici vhodná zpráva, není zde žádná čekací doba nebo prodleva.

Pokud jsou požadavky MQGET blokovány nebo požadavky MQGET přestanou být při čekání blokovány, čekání je zrušeno. Volání je dokončeno s MQCC_FAILED a kódem příčiny MQRC_GET_INHIBITED, bez ohledu na to, zda jsou ve frontě vhodné zprávy.

Příkaz MQGMO_WAIT můžete použít s volbami MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT .

Pokud ve stejné sdílené frontě čeká několik aplikací, následující pravidla vyberou, která aplikace se aktivuje, když přijde vhodná zpráva:

Tabulka 495. Pravidla pro aktivaci volání MQGET ve sdílené frontě.

Počet volání příkazu 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	Jsou aktivována všechna volání MQGET s volbou BROWSE .
Jedna a více	Jedna a více	Je aktivováno jedno volání MQGET bez volby BROWSE . Počet volání příkazu MQGET s aktivovanou volbou BROWSE je nepředvídatelný.

Pokud ve stejné frontě čeká více než jedno volání MQGET bez volby BROWSE , aktivuje se pouze jeden z nich. Správce front se pokusí o prioritu čekání na volání v následujícím pořadí:

1. Specifické požadavky typu get-wait, které mohou být uspokojeny pouze určitými zprávami, například s určitými zprávami, které mají specifický MsgId nebo CorrelId (nebo obojí).
2. Obecné požadavky typu get-wait, které mohou být uspokojeny jakoukoli zprávou.

Poznámka:

[2] Volání MQGET s uvedením volby MQGMO_LOCK je považováno za volání bez procházení.

- V první kategorii není poskytnuta žádná další priorita pro více konkrétních požadavků na získání čekání. Například požadavky, které uvádějí jak MsgId , tak CorrelId .
- V jedné z kategorií nelze předpovědět, která aplikace je vybrána. Zvláště čekání na aplikaci není nutně tím, co je vybráno.
- Délka cesty a aspekty plánování priority operačního systému mohou znamenat, že čeká se aplikace nižší priority operačního systému, než se očekává, že tato zpráva načte zprávu.
- Může se také stát, že aplikace, která nečeká, načte zprávu v preferovaném pořadí na takový, který je.

z/OS V systému z/OSplatí následující body:

- Pokud chcete, aby aplikace pokračovala v práci s jinou prací při čekání na příchod zprávy, zvažte raději použití volby signálu (MQGMO_SET_SIGNAL). Avšak volba signálu je specifická pro prostředí; aplikace, které musíte do portu mezi různými prostředími, nesmí používat.
- Pokud existuje více než jedno volání MQGET čeká se na stejnou zprávu se směsí voleb čekání a signálu, každá čekající volání se považuje za stejně. Jde o chybu při specifikaci MQGMO_SET_SIGNAL s MQGMO_WAIT. Je to také chyba pro uvedení této volby s manipulátorem fronty, pro který je signál nevyřízený.
- Uvedete-li MQGMO_WAIT nebo MQGMO_SET_SIGNAL pro frontu, která má IndexType z MQIT_MSG_TOKEN, nejsou přípustná žádná kritéria výběru. To znamená, že:
  - Používáte-li version-1 MQGMO, nastavte pole MsgId a CorrelId v MQMD uvedeném na volání MQGET pro MQMI_NONE a MQCI_NONE.
  - Používáte-li version-2 nebo pozdější 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 neshodují, po 200 milisekund je signál ECB vyslán MQEC_MSG_ARRILED.

  K tomu dochází, i když nebyla do fronty doručena vhodná zpráva, dokud nevyprší čekací interval, když je fronta publikována s MQEC_WAIT_INTERVAL_EXPIRED. Je-li odeslán příkaz MQEC_MSG_ARRILED, je třeba znovu zadat druhé volání MQGET , aby bylo možné zprávu načíst, je-li k dispozici.

  Tato technika se používá k ujištění, že jste včas informováni o příchodu zprávy, ale ve srovnání s podobnou posloupností volání u nesdílené fronty se může objevit neočekávaná režie zpracování.

MQGMO_WAIT je ignorován, pokud je zadán s MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_MSG_UNDER_CURSOR ; žádná chyba se nevyskytne.

MQGMO_NO_WAIT
Aplikace nebude čekat, pokud není k dispozici žádná vhodná zpráva. MQGMO_NO_WAIT je opakem MQGMO_WAIT. MQGMO_NO_WAIT je definován v dokumentaci programu podpory. Je-li uveden žádný, je to výchozí nastavení.

SIGNÁL MQGMO_SET_DATA
Tuto volbu použijte s poli Signal1 a Signal2 . Umožňuje aplikacím pokračovat s jinou prací a čekat na příchod zprávy. Umožňuje také (jsou-li k dispozici vhodná zařízení operačního systému) čekat na zprávy přicházející do více než jedné fronty.

Poznámka: Volba MQGMO_SET_SIGNAL je specifická pro prostředí; nepoužívat ji pro aplikace, které chcete portovat.

Za dvou okolností se volání dokončí stejným způsobem, jako by tato volba nebyla zadána:

1. Pokud momentálně dostupná zpráva splňuje kritéria uvedená v deskriptoru zpráv.
2. Je-li zjištěna chyba parametru nebo jiná synchronní chyba.

Pokud není v současné době k dispozici žádná zpráva splňující kritéria uvedená v deskriptoru zprávy, vrátí se řízení do aplikace bez čekání na příchod zprávy. Parametry **CompCode** a **Reason** jsou nastaveny na MQCC_WARNING a MQRC_SIGNAL_REQUEST_ACCEPTED. Ostatní výstupní pole

v deskriptoru zpráv a výstupní parametry volání MQGET nejsou nastaveny. Je-li vhodná zpráva doručena později, signál se doručí zveřejněním ECB.

Volající musí poté znovu zadat volání MQGET , aby bylo možné zprávu načíst. Aplikace může čekat na tento signál pomocí funkcí poskytovaných operačním systémem.

Pokud operační systém poskytuje vícenásobný mechanismus čekání, můžete jej použít k čekání na příchod zprávy do libovolného z několika front.

Je-li zadán nenulový WaitInterval , signál se doručí po vypršení čekací doby. Správce front může také zrušit čekání. V takovém případě bude signál doručen.

Více než jedno volání MQGET může nastavit signál pro stejnou zprávu. Pořadí, ve kterém jsou aplikace aktivovány, je stejné, jak je popsáno v tématu MQGMO_WAIT.

Pokud více než jedno volání MQGET čeká na stejnou zprávu, každá čekající volání se považuje za stejně. Volání mohou zahrnovat kombinaci voleb čekání a signálu.

Za určitých podmínek může volání MQGET načíst zprávu a může být doručena signál, který je výsledkem příchodu stejné zprávy. Když je dodán signál, musí být připravena aplikace, aby nebyla k dispozici žádná zpráva.

Popisovač fronty nemůže mít více než jeden nevyřízený 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, signální ECB uživatele je vystavena MQEC_MSG_ARRIVED po 200 milisekund.

K tomu dochází, i když ve frontě nebyla doručena vhodná zpráva, dokud nevyprší čekací interval, když je fronta publikována s MQEC_WAIT_INTERVAL_EXPIRED. Je-li produkt MQEC_MSG_ARRIVED uveřejněn, je třeba znovu zadat druhé volání MQGET , aby bylo možné zprávu načíst, je-li k dispozici.

Tato technika se používá k ujištění, že jste včas informováni o příchodu zprávy, ale ve srovnání s podobnou posloupností volání u nesdílené fronty se může objevit neočekávaná režie zpracování.

To není efektivní metoda načítání zpráv, když se zprávy přidávají zřídka. Chcete-li se vyhnout této režii pro případ procházení, zadejte MsgId (pokud není indexováno nebo indexováno produktem MsgId) nebo *CorrelId* (pokud je indexováno produktem CorrelId) shodující se s voláním MQGET .

z/OS Tato volba je podporována pouze v systému z/OS .

FUNKCE MQGMO_FAIL_IF_QUIESCING

Pokud je správce front ve stavu uvedení do klidového stavu, vynutí selhání volání MQGET .

z/OS V systému z/OStato volba také vynutí selhání volání MQGET , pokud se připojení (pro aplikaci CICS nebo IMS) nachází ve stavu uvedení do klidového stavu.

Je-li tato volba zadána s MQGMO_WAIT nebo MQGMO_SET_SIGNALa čekání nebo signál jsou nevyřízené v době, kdy správce front vstoupí do klidového stavu, postupujte takto:

- Čekání je zrušeno a volání vrátí kód dokončení MQCC_FAILED s kódem příčiny MQRC_Q_MGR_QUIESCING nebo MQRC_CONNECTION_QUIESCING.
- Signál je zrušený s kódem dokončení signálu specifickým pro prostředí.

  z/OS V systému z/OSje signál dokončen s kódem dokončení události MQEC_Q_MGR_QUIESCING nebo MQEC_CONNECTION_QUIESCING.

Není-li parametr MQGMO_FAIL_IF_QUIESCING zadán a správce front nebo připojení přejde do klidového stavu, nebude volba čekat nebo signál zrušena.

Volby bodu synchronizace

Následující volby se vztahují k účasti volání MQGET v rámci pracovní jednotky:

MQGMO_SYNCPOINT
Požadavek má fungovat v rámci běžných protokolů jednotky práce. Zpráva je označena jako nedostupná pro jiné aplikace, ale je vymazána z fronty pouze tehdy, když je potvrzena transakce. Zpráva je znovu zpřístupněna, pokud je jednotka práce zálohována.

Můžete ponechat MQGMO_SYNCPOINT a MQGMO_NO_SYNCPOINT nenastaveno. V takovém případě je zahrnutí požadavku get v protokolech pracovní jednotky určeno prostředím, které spouští správce front. Není určen prostředím, kde je spuštěna aplikace.

- z/OS V systému z/OSse požadavek na získání nachází v rámci transakce.
- Ve všech prostředích kromě z/OSse požadavek na získání nenachází v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nesmí aplikace, kterou chcete nastavit na port, tuto volbu povolit, aby byla výchozí; zadejte explicitně MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT .

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 má fungovat v rámci normálních protokolů jednotky práce, ale pouze tehdy, je-li zpráva načtená, trvalá. Trvalá zpráva má hodnotu MQPER_PERSISTENT v poli Persistence v MQMD.

- Je-li zpráva trvalá, bude správce front zpracovávat volání, jako by aplikace byla zadána MQGMO_SYNCPOINT.
- Pokud zpráva není trvalá, bude správce front zpracovávat volání, jako by aplikace byla zadána 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 IBM MQ MQI clients připojené k těmto systémům.

MQGMO_NO_SYNCPOINT
Požadavek má fungovat mimo běžné protokoly jednotek práce. Pokud obdržíte zprávu bez volby procházení, je vymazána z fronty okamžitě. Zprávu nelze znovu zpřístupnit tak, že zazálohujete jednotku práce.

Tato volba se předpokládá, pokud zadáte MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT.

Můžete ponechat MQGMO_SYNCPOINT a MQGMO_NO_SYNCPOINT nenastaveno. V takovém případě je zahrnutí požadavku get v protokolech pracovní jednotky určeno prostředím, které spouští správce front. Není určen prostředím, kde je spuštěna aplikace.

- z/OS V systému z/OSse požadavek na získání nachází v rámci transakce.
- Ve všech prostředích kromě z/OSse požadavek na získání nenachází v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nemusí aplikace, kterou chcete nastavit na port, tuto volbu povolit, aby byla explicitně nastavena; zadejte explicitně MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT .

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**
Zálohovat jednotku práce bez opětovného uvedení do fronty, která byla označena touto volbou.

Tato volba je podporována pouze v systému z/OS.

Je-li tato volba zadána, musí být zadán také příznak MQGMO_SYNCPOINT . 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 CICSmůže být nutné vydat volání extra IBM MQ po provedení zálohy pracovní jednotky obsahující zprávu označenou MQGMO_MARK_SKIP_BACKOUT. Chcete-li potvrdit novou jednotku práce obsahující označenou zprávu, musíte vydat výzvu IBM MQ . Volání může být libovolné volání IBM MQ .

1. Pokud jste v systému IMSneaplikovali IMS APAR PN60855 a spouštíte aplikaci MPP nebo BMP produktu IMS .
2. Pokud v produktu CICSspouštíte nějakou aplikaci.

V obou případech vydejte jakékoli volání IBM MQ před potvrzením nové jednotky práce obsahující zazálohovanou zprávu.

Poznámka: V rámci pracovní jednotky může existovat pouze jeden požadavek get označený jako přeskočení odvolání, stejně jako žádný nebo několik neoznačených požadavků get.

Pokud se aplikace zálohuje z pracovní jednotky, zpráva, která byla načtena pomocí MQGMO_MARK_SKIP_BACKOUT , není obnovena do předchozího stavu. Další aktualizace prostředků se zálohují. S touto zprávou se zachází tak, jako kdyby byla načtena v nové pracovní jednotce spuštěné požadavkem vrácení. Zpráva se načte bez volby MQGMO_MARK_SKIP_BACKOUT .

Produkt MQGMO_MARK_SKIP_BACKOUT je užitečný v případě, že po změně některých prostředků je zřejmé, že se jednotka práce nemůže úspěšně dokončit. Vynecháte-li tuto volbu, zálohování jednotky práce znovu obnoví zprávu ve frontě. Stejná posloupnost událostí se vyskytne znovu, když se zpráva příště načte.

Avšak pokud uvedete MQGMO_MARK_SKIP_BACKOUT na původním volání MQGET , zálohování jednotky práce zálohuje aktualizace ostatních prostředků. S touto zprávou se zachází, jako kdyby byla načtena pod novou pracovní jednotkou. Aplikace může provádět odpovídající ošetření chyb. Může odeslat zprávu o zprávě odesílateli původní zprávy nebo původní zprávu do fronty nedoručených zpráv. Poté může potvrdit novou jednotku práce. Potvrzení nové jednotky práce odstraní zprávu trvale z původní fronty.

MQGMO_MARK_SKIP_BACKOUT označuje jednotlivou fyzickou zprávu. Pokud zpráva patří do skupiny zpráv, další zprávy ve skupině nejsou označeny. Podobně platí, že je-li označená zpráva segmentem logické zprávy, ostatní segmenty v logické zprávě nejsou označeny.

Jakákoli zpráva ve skupině může být označena, ale pokud se zprávy načtou pomocí MQGMO_LOGICAL_ORDER, je výhodné označit první zprávu ve skupině. Je-li jednotka práce vrácena, první (označená) zpráva se přesune na novou pracovní jednotku. Druhá a pozdější zpráva ve skupině byla znovu zavedena do fronty. Zprávy ponechané ve frontě nemohou být načteny jinou aplikací pomocí produktu MQGMO_LOGICAL_ORDER. První zpráva ve skupině již není ve frontě. Avšak aplikace, která zálohoval jednotku práce, může načíst druhou a pozdější zprávu do nové jednotky práce pomocí volby MQGMO_LOGICAL_ORDER . První zpráva již byla načtena.

Někdy může být zapotřebí vrátit novou pracovní jednotku. Například, protože fronta nedoručených zpráv je plná a zpráva nesmí být vyřazena. Zálohování nové jednotky práce znovu obnoví zprávu v původní frontě, která zabrání ztrátě zprávy. Avšak v tomto zpracování situace nemůže pokračovat. Po dokončení nové pracovní jednotky musí aplikace informovat operátora nebo administrátora, že došlo k neopravitelné chybě, a poté ji dokončit.

MQGMO_MARK_SKIP_BACKOUT funguje pouze v případě, že je jednotka práce obsahující požadavek na získání přerušena aplikací, která ji zálohuje. Je-li jednotka práce obsahující požadavek na získání vrácena, protože transakce nebo systém selže, je MQGMO_MARK_SKIP_BACKOUT ignorován. Jakákoli zpráva načtená pomocí této volby se obnoví do fronty stejným způsobem jako zprávy načtené bez této volby.

Volby procházení

Následující volby se vztahují k procházení zpráv ve frontě:

NEJPRVE MQGMO_BROWSE_FIRST
Je-li fronta otevřena s volbou MQOO_BROWSE, je umístěn kurzor procházení, umístěný logicky před první zprávou ve frontě. Pak můžete použít volání MQGET s určením volby MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXTnebo MQGMO_BROWSE_MSG_UNDER_CURSOR k získání zpráv z fronty nedestruktivně. Kurzor pro procházení označuje umístění ve zprávách ve frontě, od kterého další volání příkazu MQGET s produktem MQGMO_BROWSE_NEXT vyhledá vhodnou zprávu.

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

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení.

Volání MQGET s parametrem MQGMO_BROWSE_FIRST ignoruje předchozí pozici kurzoru pro procházení. Načítá se první zpráva ve frontě, která splňuje podmínky uvedené v deskriptoru zpráv. Zpráva zůstává ve frontě a kurzor procházení je umístěn na této zprávě.

Po tomto volání je kurzor procházení umístěn ve zprávě, která byla vrácena. Zpráva může být odebrána z fronty před tím, než bude vydáno další volání MQGET s MQGMO_BROWSE_NEXT . V takovém případě zůstane kurzor procházení na pozici ve frontě, kterou zpráva obsazená, i když je tato pozice nyní prázdná.

Chcete-li zprávu odebrat z fronty, použijte volbu MQGMO_MSG_UNDER_CURSOR s voláním MQGET bez procházení procházení.

Kurzor procházení není přesunut pomocí volání MQGET bez procházení procházení, a to i v případě použití stejného manipulátoru *Hobj* . Nepřesunuje se ani po 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 zprávu, která je procházena.

Můžete zadat MQGMO_BROWSE_FIRST s libovolnou platnou kombinací voleb MQGMO_* a MQMO_* , které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Zadáte-li MQGMO_LOGICAL_ORDER, budou zprávy procházeny v logickém pořadí. Vynecháte-li tuto volbu, budou zprávy zkontrolovány ve fyzickém pořadí. Uvedete-li MQGMO_BROWSE_FIRST, můžete přepínat mezi logickým pořadím a fyzickým příkazem. Následná volání MQGET pomocí produktu MQGMO_BROWSE_NEXT prohledá frontu ve stejném pořadí jako poslední volání, které bylo zadáno 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. Uvedete-li MQGMO_BROWSE_FIRST, správce front ignoruje informace o skupině a segmentu pro procházení. Skenuje frontu, jako by neexistovala žádná aktuální skupina a žádná aktuální logická zpráva. Je-li volání MQGET úspěšné, kód dokončení MQCC_OK nebo MQCC_WARNING, informace o skupině a segmentu pro procházení jsou nastaveny na informace vrácené zprávy. Pokud se volání nezdaří, zůstanou informace o skupině a segmentu stejné jako před voláním.

PŘÍŠTĚ MQGMO_BROWSE_NEXT

Postoupit kurzor procházení na další zprávu ve frontě, která odpovídá kritériím výběru zadaným ve volání MQGET . Zpráva se vrátí do aplikace, ale zůstane ve frontě.

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

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení.

MQGMO_BROWSE_NEXT se chová stejně jako MQGMO_BROWSE_FIRST, pokud se jedná o první volání k procházení fronty, poté, co byla fronta otevřena pro procházení.

Zpráva pod kurzorem může být odebrána z fronty před tím, než bude vydáno další volání MQGET s MQGMO_BROWSE_NEXT . Kurzor procházení logicky zůstává na pozici ve frontě, ve které byla zpráva obsazena, i když je tato pozice nyní prázdná.

Zprávy jsou ukládány do fronty 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 827).

Fronta může mít MsgDeliverySequence z MQMDS_PRIORITY. Zpráva dorazí do fronty, která má vyšší prioritu než ta, na kterou v současné době ukazuje kurzor procházení. V takovém případě se zpráva s vyšší prioritou nenachází během aktuálního procházení fronty pomocí produktu MQGMO_BROWSE_NEXT. Lze ji nalézt až poté, co byl kurzor procházení obnoven s parametrem MQGMO_BROWSE_FIRSTnebo opětovným otevřením fronty.

Volbu MQGMO_MSG_UNDER_CURSOR lze použít s neprohlížečovým voláním MQGET , je-li to nutné, aby byla zpráva odebrána z fronty.

Kurzor procházení se nepřesunuje mezi voláními MQGET , které nepoužívají procházení, pomocí stejného popisovače Hobj .

Uveďte volbu MQGMO_LOCK s touto volbou pro zamknutí zprávy, která je procházena.

Můžete zadat MQGMO_BROWSE_NEXT s libovolnou platnou kombinací voleb MQGMO_* a MQMO_* , které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Zadáte-li MQGMO_LOGICAL_ORDER, budou zprávy procházeny v logickém pořadí. Vynecháte-li tuto volbu, budou zprávy zkontrolovány ve fyzickém pořadí. Uvedete-li MQGMO_BROWSE_FIRST, můžete přepínat mezi logickým pořadím a fyzickým příkazem. Následná volání MQGET pomocí produktu MQGMO_BROWSE_NEXT prohledá frontu ve stejném pořadí jako poslední volání, které bylo zadáno MQGMO_BROWSE_FIRST pro popisovač fronty. Volání se nezdaří s kódem příčiny MQRC_INCONSISTENT_BROWSE , pokud tato podmínka není splněna.

Poznámka: Zvláštní opatrnosti při použití volání MQGET je zapotřebí při procházení za koncem skupiny zpráv, 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, uvedení MQMO_MATCH_MSG_SEQ_NUMBER s MsgSeqNumber nastaveným na 1 vrátí první zprávu ve skupině již prohlédnuto. K tomuto výsledku může dojít okamžitě nebo k několika MQGET voláním později, pokud dojde k zasahující skupiny. Stejná úvaha platí i 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
Získejte zprávu, na kterou ukazuje kurzor bez destruktivního nastavení, bez ohledu na volby MQMO_* zadané v poli MatchOptions v MQGMO.

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

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení.

Zpráva, na kterou ukazuje procházení kurzorem, je ta, která byla naposledy načtena pomocí volby MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT . Volání se nezdaří, pokud ani jedno z těchto volání nebylo pro tuto frontu vydáno, protože bylo otevřeno. Volání se také nezdaří, pokud byla zpráva, která byla pod kurzorem procházení, od té doby destruktivně načtena.

Poloha kurzoru procházení se při tomto volání nezmění.

Volbu MQGMO_MSG_UNDER_CURSOR lze použít s neprohlížečovým voláním MQGET , aby se zpráva odebrala z fronty.

Kurzor procházení není přesunut pomocí volání MQGET bez procházení procházení, a to i v případě použití stejného manipulátoru Hobj . Nepřesunuje se ani po volání procházení MQGET , které vrací kód dokončení MQCC_FAILEDnebo kód příčiny MQRC_TRUNCATED_MSG_FAILED.

Pokud je MQGMO_BROWSE_MSG_UNDER_CURSOR zadán s MQGMO_LOCK:

- Pokud je již zpráva uzamčena, musí být pod kurzorem, takže se vrátí bez odemčení a zamknutí znovu. Zpráva zůstane uzamknuta.
- Pokud zde není žádná zamčená zpráva a je zde zpráva pod kurzorem procházení, je uzamčena a vrácena do aplikace. Pokud pod kurzorem procházení není žádná zpráva, volání selže.

Je-li MQGMO_BROWSE_MSG_UNDER_CURSOR zadán bez MQGMO_LOCK:

- Je-li již zpráva uzamknuta, musí být pod kurzorem. Zpráva se vrátí do aplikace a poté odemknuta. Vzhledem k tomu, že zpráva je nyní odemknuta, neexistuje žádná záruka, že ji lze znovu prohlížet, nebo ji lze destruktivně načíst stejnou aplikací. Může být načten destruktivně jinou aplikací získávajícím zprávy z fronty.
- Pokud zde není žádná zamčená zpráva a je zde zpráva pod kurzorem procházení, vrátí se do aplikace. Pokud pod kurzorem procházení není žádná zpráva, volání selže.

Je-li MQGMO_COMPLETE_MSG zadán s MQGMO_BROWSE_MSG_UNDER_CURSOR, kurzor procházení musí identifikovat zprávu, jejíž pole Offset v MQMD je nula. Není-li tato podmínka splněna, volání selže 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
Načítá zprávu, na kterou ukazuje kurzor procházení, bez ohledu na volby MQMO_* zadané v poli MatchOptions v MQGMO. Zpráva se odebere z fronty.

Zpráva, na kterou ukazuje procházení kurzorem, je ta, která byla naposledy načtena pomocí volby MQGMO_BROWSE_FIRST nebo MQGMO_BROWSE_NEXT .

Je-li MQGMO_COMPLETE_MSG zadán s MQGMO_MSG_UNDER_CURSOR, kurzor procházení musí identifikovat zprávu, jejíž pole Offset v MQMD je nula. Není-li tato podmínka splněna, volání selže 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

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení i pro vstup. Pokud kurzor procházení momentálně neukazuje na zprávu, kterou lze načíst, je vrácena chyba voláním funkce MQGET .

POPISOVAČ MQGMO_MARK_ BROWSE_HANDLE
Je označena zpráva vrácená úspěšným serverem MQGETnebo identifikovaná vrácenou hodnotou MsgToken. Značka je specifická pro popisovač objektu použitý ve volání.

Zpráva se neodebere z fronty.

MQGMO_MARK_BROWSE_HANDLE je platný pouze tehdy, je-li zadána jedna z následujících voleb:

- MQGMO_BROWSE_FIRST
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_BROWSE_NEXT

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 zůstane v tomto stavu, dokud nenastane jedna z následujících událostí:

- Dotčená obsluha objektu je uzavřena, buď normálně, nebo jinak.
- Zpráva je neoznačena pro tento popisovač voláním MQGET s volbou MQGMO_UNMARK_BROWSE_HANDLE.
- Zpráva je vrácena z volání destruktivního objektu MQGET, které je dokončeno s MQCC_OK nebo MQCC_WARNING. Stav zprávy zůstane změněn, i když je MQGET později odvolaný.
- Platnost zprávy vyprší.

MQGMO_MARKER_BROWSE_CO_OP
Zpráva, která je vrácena úspěšným produktem MQGETnebo identifikovaná vrácenou hodnotou *MsgToken*, je označena pro všechny popisovače v rámci spolupracující sady.

Značka na úrovni spolupráce je navíc k libovolné značce popisovače, která mohla být nastavena.

Zpráva se neodebere z fronty.

Volba MQGMO_MARK_BROWSE_CO_OP je platná pouze v případě, že použitý popisovač objektu byl vrácen voláním příkazu MQOPEN , které bylo zadáno MQOO_CO_OP. Musíte také zadat jeden 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

Je-li zpráva již označena a volba MQGMO_UNMARKED_BROWSE_MSG není uvedena, volání selže s MQCC_FAILED a kódem příčiny MQRC_MSG_MARKED_BROWSE_CO_OP.

Zpráva zůstane v tomto stavu, dokud nenastane jedna z následujících událostí:

- Všechny manipulátory objektů v spolupracující sadě jsou zavřeny.
- Zpráva je neoznačena pro spolupracující prohlížeče voláním MQGET s volbou MQGMO_UNMARK_BROWSE_CO_OP.
- Zpráva je automaticky neoznačena správcem front.
- Zpráva se vrátí z volání na neprocházení MQGET. Stav zprávy zůstane změněn, i když je MQGET později odvolaný.
- Platnost zprávy vyprší.

MQGMO_UNMARKED_BROWSE_MSG
Volání MQGET , které uvádí MQGMO_UNMARKED_BROWSE_MSG , vrací zprávu, která má být považována za neoznačenou pro její obsluhu. Nevrátí se zpráva, pokud byla zpráva označena pro její popisovač. Nevrátí ji také, pokud byla fronta otevřena voláním do produktu MQOPEN, s 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 tuto volbu uvádí, zpráva již není považována za otevřenou manipulátory v sadě spolupracujících popisovačů, které mají být označeny pro spolupracující sadu. Tato zpráva je stále považována za označenou na úrovni obsluhy, pokud byla před tímto voláním označena na úrovni obsluhy.

Použití MQGMO_UNMARK_BROWSE_CO_OP je platné pouze s popisovačem vráceným úspěšným voláním příkazu MQOPEN s volbou MQOO_CO_OP. Produkt MQGET uspěje i v případě, že zpráva není považována za označenou spolupracujícím souborem manipulátorů.

MQGMO_UNMARK_BROWSE_CO_OP není platný při volání MQGET , který není procházení, nebo s libovolnou z následujících možností:

- 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

POPISOVAČ MQGMO_UNMARK_ BROWSE_HANDLE
Po volání příkazu MQGET , který tuto volbu uvádí, nebude již tato zpráva považována za označenou daným popisovačem.

Volání se zdaří i v případě, že zpráva není označena pro tento popisovač.

Tato volba není platná pro volání MQGET bez procházení, nebo 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 uzamčení

Následující volby se vztahují k zamykání zpráv ve frontě:

MQGMOVÝ_ZÁMEK
Zamkni zprávu, která je procházena, takže se zpráva stane neviditelnou pro všechny ostatní ovladače otevřené 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 může být uzamčena pouze jedna zpráva. Může se jednat o logickou zprávu nebo o fyzickou zprávu:

- Uvedete-li MQGMO_COMPLETE_MSG, všechny segmenty zprávy, které tvoří logickou zprávu, jsou uzamčeny pro obsluhu fronty. Všechny zprávy musí být přítomné ve frontě a jsou k dispozici pro načtení.
- Vynecháte-li MQGMO_COMPLETE_MSG, uzamkne se pouze jedna fyzická zpráva s manipulátorem fronty. Pokud se tato zpráva stane segmentem logické zprávy, zamčený segment zabraňuje ostatním aplikacím, které používají produkt MQGMO_COMPLETE_MSG k načtení nebo procházení logické zprávy.

Zamknutá zpráva je vždy ta pod kurzorem procházení. Zprávu lze z fronty odstranit pomocí pozdějšího volání příkazu MQGET , které určuje volbu MQGMO_MSG_UNDER_CURSOR . Ostatní volání příkazu MQGET používající manipulátor fronty mohou také zprávu odebrat (například volání, které určuje identifikátor zprávy zamčené 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:

- Vydáním dalšího volání MQGET pro tento popisovač uveďte buď MQGMO_BROWSE_FIRST , nebo MQGMO_BROWSE_NEXT. Zámek se uvolní, je-li volání dokončeno s MQCC_OK nebo MQCC_WARNING. Zpráva zůstane zamknutá, je-li volání dokončeno s MQCC_FAILED. Platí však následující výjimky:
  - Zpráva se nezamkne, pokud je MQCC_WARNING vrácen s MQRC_TRUNCATED_MSG_FAILED.
  - Zpráva je odemknuta, pokud je MQCC_FAILED vrácen s MQRC_NO_MSG_AVAILABLE.

  Pokud také uvedete MQGMO_LOCK, vrácená zpráva je zamčená. Pokud vynecháte MQGMO_LOCK, po volání se nezamkne žádná zamčená zpráva.

  Uvedete-li MQGMO_WAITa žádná zpráva není okamžitě k dispozici, původní zpráva se odemkne před začátkem čekání.
- Vydáním dalšího volání MQGET pro tento popisovač, s MQGMO_BROWSE_MSG_UNDER_CURSOR, bez MQGMO_LOCK. Zámek se uvolní, je-li volání dokončeno s MQCC_OK nebo MQCC_WARNING. Zpráva zůstane zamknutá, je-li volání dokončeno s MQCC_FAILED. Platí však následující výjimka:
  - Zpráva se nezamkne, pokud je MQCC_WARNING vrácen s MQRC_TRUNCATED_MSG_FAILED.
- Vydáním dalšího volání MQGET pro tento popisovač s MQGMO_UNLOCK.
- Vyvolání volání MQCLOSE pomocí manipulátoru. Hodnota MQCLOSE může být implicitní, způsobená ukončením aplikace.

Není potřeba žádná speciální volba MQOPEN pro zadání MQGMO_LOCK, jiného než MQOO_BROWSE, který je potřebný k uvedení doprovodného příkazu pro procházení.

MQGMO_LOCK není platný s žádnou z následujících voleb:

- MQGMO_MARK_SKIP_BACKOUT
- MQGMO_SYNCPOINT
- MQGMO_SYNCPOINT_IF_PERSISTENT
- MQGMO_UNLOCK

MQGMO_ODEMKNOUT
Zpráva, která má být odemknuta, musí být dříve zamčena voláním MQGET s volbou MQGMO_LOCK . Pokud pro tento popisovač není k dispozici žádná zpráva, bude volání dokončeno s MQCC_WARNING a MQRC_NO_MSG_LOCKED.

Parametry **MsgDesc**, **BufferLength**, **Buffer**a **DataLength** se nekontrolují ani nemění, pokud zadáte MQGMO_UNLOCK. V produktu *Buffer*není vrácena žádná zpráva.

K zadání MQGMO_UNLOCK není zapotřebí žádná speciální volba otevření (ačkoli MQOO_BROWSE je potřeba k vydání požadavku na uzamčení na prvním místě).

Tato volba není platná s žádnými volbami kromě následujících:

- MQGMO_NO_WAIT
- MQGMO_NO_SYNCPOINT

Obě tyto možnosti se předpokládají bez ohledu na to, zda jsou zadány nebo ne.

Volby zpráv-data

Následující volby se vztahují ke zpracování dat zprávy, když je zpráva přečtena z fronty:

SOUBOR MQGMO_ACCEPT_TRUNCATED_MSG
Je-li vyrovnávací paměť zpráv příliš malá, aby mohla obsahovat úplnou zprávu, umožněte volání MQGET vyrovnávací paměť. Produkt MQGET zaplní vyrovnávací paměť tak velkou část zprávy, kterou dokáže. Vydá kód dokončení varování a dokončí své zpracování. To znamená, že:

- Při procházení zpráv je kurzor procházení pro vracenou zprávu rozšířený.
- Při odebírání zpráv je vrácená zpráva odebrána z fronty.
- Kód příčiny MQRC_TRUNCATED_MSG_ACCEPTED je vrácen, pokud se nevyskytne jiná chyba.

Bez této volby je vyrovnávací paměť stále zaplněna jako velká část zprávy, jak ji lze zadržet. Je vydán kód dokončení varování, ale zpracování není dokončeno. To znamená, že:

- Při procházení zpráv není kurzor procházení pokročilý.
- Při odebírání zpráv se zpráva neodebere z fronty.
- Kód příčiny MQRC_TRUNCATED_MSG_FAILED je vrácen, pokud se nevyskytne jiná chyba.

MQGMO_CONVERT
Tato volba převede data aplikace ve zprávě tak, aby odpovídala hodnotám CodedCharSetId a Encoding uvedeným v parametru **MsgDesc** na volání MQGET . Data se převedou před zkopírováním do parametru **Buffer** .

Pole Format zadané při vložení zprávy je předpokládáno procesem převodu za účelem identifikace povahy dat ve zprávě. Data zprávy jsou převedena správcem front pro vestavěné formáty a uživatelem napsanou uživatelskou procedurou pro jiné formáty. Podrobné informace o ukončení konverze dat naleznete v části “Uživatelská procedura konverze dat” na stránce 897 .

- Je-li konverze úspěšná, pole CodedCharSetId a Encoding uvedená v parametru **MsgDesc** se nezmění při návratu z volání MQGET .
- Pokud pouze konverze selže, data zprávy se vrátí nekonvertované pole CodedCharSetId a Encoding v MsgDesc jsou nastaveny na hodnoty pro nepřekonvertované zprávy. Kód dokončení je MQCC_WARNING v tomto případě.

V obou případech tato pole popisují identifikátor znakové sady a kódování dat zprávy, která jsou vrácena v argumentu **Buffer** .

Seznam názvů formátů, pro které správce front provádí převod, naleznete v poli *Format* , které je popsáno v “MQMD-Deskriptor zpráv” na stránce 419 .

Volby skupin a segmentů

Následující volby se vztahují ke zpracování zpráv ve skupinách a segmentech logických zpráv. Před popisy možností jsou zde uvedeny některé definice důležitých výrazů:

Fyzická zpráva
Fyzická zpráva je nejmenší jednotka informací, které lze umístit do fronty nebo z ní odstranit. Často odpovídá informacím zadaným nebo načteným na jediném volání MQPUT, MQPUT1nebo MQGET . Každá

fyzická zpráva má svůj vlastní deskriptor zprávy MQMD. Obvykle se fyzické zprávy rozlišují podle lišících hodnot identifikátoru zprávy, pole MsgId v MQMD. Správce front nevynucuje různé hodnoty.

Logická zpráva
Logická zpráva je jediná jednotka informace o aplikaci. Pokud nejsou k dispozici omezení systému, je logická zpráva stejná jako fyzická zpráva. Jsou-li logické zprávy velké, omezení systému by mohla učinit vhodné nebo nezbytné k rozdělení logické zprávy do dvou nebo více fyzických zpráv, nazývaných segmenty.

Logická zpráva, která byla segmentována, se skládá ze dvou nebo více fyzických zpráv, které mají stejný identifikátor skupiny bez hodnoty null, pole GroupId v MQMD. Mají stejné pořadové číslo zprávy, pole MsgSeqNumber v MQMD. Segmenty jsou rozlišeny odlišnými hodnotami pro offset segmentu, pole Offset v MQMD. Offset segmentu je posun dat ve fyzické zprávě od začátku dat v logické zprávě. Vzhledem k tomu, že každý segment je fyzická zpráva, segmenty v logické zprávě mají obvykle různé identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale pro kterou byla segmentace povolena odesílající aplikací, má také identifikátor skupiny bez hodnoty null. V tomto případě existuje pouze jedna fyzická zpráva s tímto identifikátorem skupiny, pokud tato logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které byla segmentace zablokována odesílající aplikací, mají identifikátor skupiny s hodnotou null, MQGI_NONE, pokud tato logická zpráva nepatří do skupiny zpráv.

Skupina zpráv
Skupina zpráv je sada jedné nebo více logických zpráv, které mají stejný neprázdný identifikátor skupiny. Logické zprávy ve skupině jsou odlišeny různými hodnotami pro pořadové číslo zprávy. Pořadové číslo je celé číslo v rozsahu od 1 do n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentovaná, ve skupině jsou více než n fyzických zpráv.

MQGMO_LOGICAL_ORDER
MQGMO_LOGICAL_ORDER řídí pořadí, ve kterém jsou zprávy vraceny po sobě jdoucími voláními MQGET pro popisovač fronty. Volba musí být uvedena u každého volání.

Je-li MQGMO_LOGICAL_ORDER zadán pro následné volání MQGET pro stejný popisovač fronty, jsou zprávy ve skupinách vraceny v pořadí pořadových čísel zpráv. Segmenty logických zpráv jsou vraceny v pořadí poskytnutému jejich segmentem segmentu. Toto pořadí se může lišit od pořadí, ve kterém se tyto zprávy a segmenty vyskytují ve frontě.

Poznámka: Zadání MQGMO_LOGICAL_ORDER nemá žádné nepříznivé důsledky pro zprávy, které nepatří do skupin a které nejsou segmenty. V důsledku toho se s takovými zprávami zachází, jako by každá patřila do skupiny zpráv skládající se pouze z jedné zprávy. Je bezpečné zadat MQGMO_LOGICAL_ORDER při načítání zpráv z front, které obsahují směs zpráv ve skupinách, segmentech zpráv a nesegmentované zprávy, které nejsou ve skupinách.

Chcete-li zprávy vrátit v požadovaném pořadí, zachová správce front informace o skupině a segmentu mezi následujícími voláními MQGET . Informace o skupině a segmentu slouží k identifikaci aktuální skupiny zpráv a aktuální logické zprávy pro manipulátor fronty. Identifikuje aktuální pozici ve skupině a logické zprávě a zda jsou zprávy načítány v rámci jednotky práce. Vzhledem k tomu, že správce front uchovává tyto informace, nemusí aplikace před každým voláním MQGET nastavovat informace o skupině a segmentu. Konkrétně to znamená, že aplikace nemusí nastavovat pole GroupId, MsgSeqNumbera Offset v MQMD. Aplikace však musí u každého 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 se zavolá zpráva, která má parametr MQMF_MSG_IN_GROUP , je vrácena voláním MQGET . Je-li MQGMO_LOGICAL_ORDER uvedeno v následných voláních, tato skupina zůstane 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).

Je-li taková zpráva vrácena, skupina zpráv je ukončena a při úspěšném dokončení volání MQGET již není aktuální skupina. Podobně se logická zpráva stane aktuální logickou zprávou, jakmile se pomocí příkazu MQGET vrátí zpráva, která má parametr MQMF_SEGMENT . Logická zpráva je ukončena, když je vrácena zpráva, která má příznak MQMF_LAST_SEGMENT .

Pokud nejsou zadána žádná kritéria výběru, za sebou následují volání MQGET ve správném pořadí, zprávy pro první skupinu zpráv ve frontě. Pak vrátí zprávy pro druhou skupinu zpráv, a tak dále, dokud nejsou k dispozici žádné další zprávy. Konkrétní skupiny zpráv lze vybrat zadáním jedné nebo více z následujících voleb do pole 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. Viz pole MatchOptions popsané v části "MQGMO-Volby získání zprávy" na stránce 366 , kde jsou uvedeny další podrobnosti.

Tabulka 496 na stránce 391 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 volba MQGMO_LOGICAL_ORDER zadána při volání.

Cur grp
Indikuje, zda před voláním existuje aktuální skupina zpráv.

Cur log msg
Indikuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazení hodnot, které správce front hledá. Předchozí označuje hodnotu vrácenou pro pole v předchozí zprávě pro popisovač fronty.

Tabulka 496. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv

Volby, které uvedete	**Stav skupiny a protokolu-zpráva 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 korelační identifikátor	Předchozí identifikátor skupiny	1	Předchozí odchylka + předchozí délka segmentu
Ano	Ano	Ne	Libovolný identifikátor zprávy	Libovolný korelační identifikátor	Předchozí identifikátor skupiny	Předchozí pořadové číslo + 1	0

Tabulka 496. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv (pokračování)

Volby, které uvedete	**Stav skupiny a protokolu-zpráva před voláním**		**Hodnoty, které správce front hledá**				
Ano	Ano	Ano	Libovolný identifikátor zprávy	Libovolný korelační identifikátor	Předchozí identifikátor skupiny	Předchozí pořadové číslo	Předchozí odchylka + předchozí délka segmentu
Ne	buď	buď	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*	řízený subjektem *MatchOptions*

Je-li ve frontě přítomno více skupin zpráv a je možné je vrátit, skupiny se vrátí v pořadí určeném pozicí ve frontě prvního segmentu první logické zprávy v každé skupině. To znamená, že fyzické zprávy, které mají pořadová čísla zpráv 1, a posuny 0, určují pořadí, ve kterém jsou vráceny způsobilé skupiny.

Volba MQGMO_LOGICAL_ORDER ovlivňuje jednotky práce následujícím způsobem:

- Je-li 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, je-li použita stejná obsluha fronty. Avšak nemusí být načteny v rámci stejné jednotky práce. To umožňuje skupině zpráv skládající se z mnoha fyzických zpráv, které mají být rozděleny do dvou nebo více po sobě jdoucích jednotek práce pro manipulátor fronty.
- Pokud se první logická zpráva nebo segment ve skupině nenačte v rámci pracovní jednotky a použije se stejný ovladač fronty, nelze v rámci jednotky práce načíst žádnou z ostatních logických zpráv a segmentů ve skupině.

Nejsou-li tyto podmínky splněny, volání MQGET selže s kódem příčiny MQRC_INCONSISTENT_UOW.

Je-li zadán parametr MQGMO_LOGICAL_ORDER , nesmí být MQGMO dodaný na volání MQGET menší než MQGMO_VERSION_2a MQMD nesmí být menší než MQMD_VERSION_2. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_WRONG_GMO_VERSION nebo MQRC_WRONG_MD_VERSIONpodle potřeby.

Pokud MQGMO_LOGICAL_ORDER není zadán pro následné volání MQGET pro obsluhu fronty, jsou zprávy vraceny 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 intermingované 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 vrácené po sobě jdoucími voláními MQGET řízeny volbami MQMO_* zadanými na těchto voláních (viz pole *MatchOptions* popsané v části “MQGMO-Volby získání zprávy” na stránce 366 , kde jsou uvedeny podrobnosti o těchto volbách).

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy ve středu, po selhání systému. Když se systém restartuje, může aplikace nastavit pole GroupId, MsgSeqNumber, Offseta MatchOptions na odpovídající hodnoty a pak vydat volání MQGET s parametrem MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT , ale bez uvedení MQGMO_LOGICAL_ORDER. Je-li toto volání úspěšné, uchovává správce front informace o skupině a segmentech a následná volání MQGET používající tento manipulátor fronty mohou určit MQGMO_LOGICAL_ORDER jako normální.

Informace o skupinách a segmentech, které správce front uchovává pro volání MQGET , jsou oddělena od informací o skupině a segmentu, které si uchovává pro volání MQPUT. Kromě toho správce front uchovává samostatné informace pro:

- Volání MQGET , které odebírají zprávy z fronty.
- Volání MQGET , která prochází zprávy ve frontě.

Pro daný popisovač fronty může aplikace směšovat MQGET volání, která uvádějí MQGMO_LOGICAL_ORDER s voláními MQGET , které ne. Povšimněte si však následujících bodů:

- Pokud vynecháte MQGMO_LOGICAL_ORDER, každé úspěšné volání MQGET způsobí, že správce front nastaví informace o uložené skupině a segmentu na hodnoty odpovídající vrácené zprávě; nahradí existující informace o skupině a segmentu uchované správcem front pro obsluhu fronty. Upravovány jsou pouze informace odpovídající akci volání (procházení nebo odebrání).
- Pokud vynecháte MQGMO_LOGICAL_ORDER, volání se nezdaří, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání může být úspěšné s kódem dokončení MQCC_WARNING . Tabulka 497 na stránce 393 zobrazuje různé případy, které mohou nastat. V těchto případech, pokud kód dokončení není MQCC_OK, je kód příčiny jedním z následujících (je-li to vhodné):
  - 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 ne vstup; v těchto případech je kód dokončení vždy MQCC_OK (nepředpokládá se žádné další chyby).

Tabulka 497. 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 neukončené skupinou nebo logickou zprávou	MQCC_WARNING	MQCC_OK

Aplikace, které chtějí načítat zprávy a segmenty v logickém pořadí, se doporučuje uvést MQGMO_LOGICAL_ORDER, protože se jedná o nejjednodušší volbu, která se má použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupinách a segmentech, protože tyto informace spravuje správce front. Avšak specializované aplikace mohou vyžadovat větší kontrolu nad tím, než je uvedeno ve volbě MQGMO_LOGICAL_ORDER , a toho lze dosáhnout neuvedením této volby. Aplikace potom musí zajistit, aby pole MsgId, CorrelId, GroupId, MsgSeqNumbera Offset v MQMDa volby MQMO_* v produktu MatchOptions v MQGMObyly nastaveny správně před každým voláním MQGET .

Například aplikace, která chce předávat fyzické zprávy, které přijímá, bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, nesmí uvádět MQGMO_LOGICAL_ORDER. Ve složité síti s více cestami mezi odesílajícím a přijímajícím správcem front může dojít k nedostatku fyzických zpráv v pořadí. Uvedením MQGMO_LOGICAL_ORDERani odpovídajícího MQPMO_LOGICAL_ORDER na volání MQPUT nebude moci aplikace postoupení načítat a předávat každou fyzickou zprávu ihned, jakmile dorazí, aniž by musela čekat na to, až přijde další, v logickém pořadí.

Můžete uvést MQGMO_LOGICAL_ORDER s libovolnými z dalších voleb MQGMO_* a s různými volbami MQMO_* za vhodných okolností (viz předchozí sekce).

- 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 objekt CFSTRUCT, na který je mapována mapa fronty, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Tato volba je podporována pro všechny lokální fronty pro následující platformy:
  - AIX AIX

– 

Linux

– 

IBM i

– 

Windows

a pro IBM MQ MQI clients připojené k těmto systémům.

ZPRÁVA MQGMO_COMPLETE_MSG

Volání MQGET může vrátit pouze úplnou logickou zprávu. Je-li logická zpráva segmentovaná, správce front znovu složí segmenty a vrátí k aplikaci úplnou logickou zprávu; skutečnost, že logická zpráva byla segmentována, není zřejmé, že by aplikace načítala tuto zprávu.

Poznámka: Toto je jediná volba, která způsobí, že správce front znovu sestaví segmenty zpráv. Pokud nejsou uvedeny, segmenty se vrátí jednotlivě do aplikace, jsou-li přítomné ve frontě (a vyhovují dalším kritériím výběru zadaným na volání MQGET). Aplikace, které nechtějí přijímat jednotlivé segmenty, musí vždy uvádět MQGMO_COMPLETE_MSG.

Chcete-li použít tuto volbu, aplikace musí poskytovat vyrovnávací paměť, která je dostatečně velká, aby pojmula úplnou zprávu, nebo zadejte volbu MQGMO_ACCEPT_TRUNCATED_MSG .

Pokud fronta obsahuje segmentované zprávy s některými chybějícími segmenty (například proto, že byly v síti zpožděny a dosud nedorazily), uvedení MQGMO_COMPLETE_MSG brání načtení segmentů náležejících k neúplným logickým zprávám. Tyto segmenty zpráv však stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné obnovitelné logické zprávy, i když je *CurrentQDepth* větší než nula.

Pro trvalé zprávy může správce front znovu sestavit segmenty pouze v rámci pracovní jednotky:

- Je-li volání MQGET v uživatelem definované jednotce práce, použije se tato jednotka práce. Pokud během procesu sestavení dojde k selhání volání, obnoví správce front ve frontě všechny segmenty, které byly odebrány během opětovného sestavení. Selhání však nezabrání úspěšnému potvrzení jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelsky definovaná jednotka práce, správce front vytvoří pracovní jednotku po dobu trvání hovoru. Je-li volání úspěšné, správce front automaticky potvrdí jednotku práce (aplikace ji nemusí provést). Pokud se volání nezdaří, správce front provede zálohu jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce, ale uživatelem definovaná jednotka práce existuje, správce front nemůže znovu sestavit soubor. Pokud zpráva nevyžaduje opětovnou montáž, volání může být stále úspěšné. Pokud však zpráva vyžaduje opětovnou sestavení, volání selže s kódem příčiny MQRC_UOW_NOT_AVAILABLE.

V případě přechodných zpráv správce front nevyžaduje, aby byla k dispozici jednotka práce, která má být k dispozici pro provedení opětovného sestavení.

Každá fyzická zpráva, která má segment, má svůj vlastní deskriptor zprávy. Pro segmenty tvořící jedinou logickou zprávu jsou většinu polí v deskriptoru zprávy stejné pro všechny segmenty v logické zprávě; obvykle se jedná pouze o pole MsgId, Offseta MsgFlags , která se liší mezi segmenty v logické zprávě. Je-li však segment umístěn ve frontě nedoručených zpráv ve středním správci front, načte obslužná rutina DLQ zprávu s volbou MQGMO_CONVERT , což může mít za následek změnu znakové sady nebo kódování právě měněného segmentu. Pokud obslužná rutina DLQ úspěšně odešle segment na jeho cestě, segment může mít znakovou sadu nebo kódování, které se liší od ostatních segmentů v logické zprávě, když segment dorazí do cílového správce front.

Logická zpráva skládající se ze segmentů, v nichž se pole CodedCharSetId a Encoding liší, nelze znovu sestavit správce front do jediné logické zprávy. Místo toho správce front znovu sestaví a vrátí prvních několik po sobě jdoucích segmentů na začátku logické zprávy se stejnými identifikátory kódování a kódování a volání MQGET bude dokončeno s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_INCONSISTENT_CCSIDS nebo MQRC_INCONSISTENT_ENCODINGS. To nastane bez ohledu na to, zda je zadán parametr MQGMO_CONVERT . Chcete-li načíst zbývající segmenty, aplikace musí znovu zadat volání MQGET bez volby MQGMO_COMPLETE_MSG , načtení segmentů 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 zpráv na hodnoty, které se liší mezi segmenty. Neexistuje však žádná výhoda, pokud přijímající aplikace používá produkt MQGMO_COMPLETE_MSG k načtení logické zprávy. Když správce front znovu složí logickou zprávu, vrací v deskriptoru zpráv hodnoty z deskriptoru zpráv pro první segment; jedinou výjimkou je pole MsgFlags , které nastavuje správce front, aby indikoval, že opětovně sestavená zpráva je jediným segmentem.

Je-li pro zprávu sestavy zadán parametr MQGMO_COMPLETE_MSG , provede správce front speciální zpracování. Správce front danou frontu zkontroluje a zjišťuje, zda jsou ve frontě všechny zprávy sestavy daného typu týkající se různých segmentů v logické zprávě. Pokud jsou, lze je načíst jako jedinou zprávu zadáním příkazu MQGMO_COMPLETE_MSG. Aby to bylo možné, zprávy sestavy musí být generovány správcem front nebo agentem MCA, který podporuje segmentaci, nebo musí původní aplikace vyžadovat alespoň 100 bajtů dat zprávy (to znamená, že musí být zadány příslušné volby MQRO_*_WITH_DATA nebo MQRO_*_WITH_FULL_DATA). Je-li pro segment méně než zaplněno celé množství dat aplikace, chybějící bajty se nahradí hodnotami null ve vrácené zprávě sestavy.

Pokud je zadán parametr MQGMO_COMPLETE_MSG s hodnotou MQGMO_MSG_UNDER_CURSOR nebo MQGMO_BROWSE_MSG_UNDER_CURSOR, kurzor procházení musí být umístěn na zprávě, jejíž pole *Offset* v MQMD má hodnotu 0. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_INVALID_MSG_UNDER_CURSOR.

MQGMO_COMPLETE_MSG znamená MQGMO_ALL_SEGMENTS_AVAILABLE, které proto nemusí být zadány.

MQGMO_COMPLETE_MSG lze zadat s libovolněmi dalšími volbami MQGMO_* kromě MQGMO_SYNCPOINT_IF_PERSISTENTa s libovolnou volbou 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 objekt CFSTRUCT, na který má být mapa fronty mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na těchto platformách:
  - AIX AIX
  - IBM i IBM i
  - Linux Linux
  - Windows Windows

  a pro IBM MQ MQI clients připojené k těmto systémům je tato volba podporovaná 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 z chybějících zpráv (například proto, že byly v síti zpožděny a ještě nedorazili), uvedení MQGMO_ALL_MSGS_AVAILABLE zabrání načtení zpráv náležejících do neúplných skupin. Tyto zprávy však stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné skupiny zpráv, které lze načíst, ačkoli CurrentQDepth je větší než nula. Nejsou-li k dispozici žádné další zprávy, které lze načíst, je po uplynutí zadané čekací doby (pokud existuje) vrácen kód příčiny MQRC_NO_MSG_AVAILABLE .

Zpracování příkazu MQGMO_ALL_MSGS_AVAILABLE závisí na tom, zda je zadán také parametr MQGMO_LOGICAL_ORDER :

- Jsou-li zadány obě volby, má MQGMO_ALL_MSGS_AVAILABLE efekt pouze v případě, že neexistuje žádná aktuální skupina nebo logická zpráva. Pokud se jedná o aktuální skupinu nebo logickou zprávu, MQGMO_ALL_MSGS_AVAILABLE se ignoruje. To znamená, že MQGMO_ALL_MSGS_AVAILABLE může zůstat při zpracování zpráv v logickém pořadí zpracování.

- Je-li MQGMO_ALL_MSGS_AVAILABLE zadán bez MQGMO_LOGICAL_ORDER, MQGMO_ALL_MSGS_AVAILABLE má vždy efekt. To znamená, že po odebrání první zprávy ve skupině z fronty musí být tato volba vypnuta, aby bylo možné odebrat zbývající zprávy ve skupině.

Úspěšné dokončení volání MQGET se zadáním MQGMO_ALL_MSGS_AVAILABLE znamená, že v době, kdy bylo volání MQGET vydáno, byly všechny zprávy ve skupině ve frontě. Avšak mějte na paměti, že jiné aplikace mohou stále odebírat zprávy ze skupiny (skupina není zamknuta na aplikaci, která načte první zprávu ve skupině).

Pokud vynecháte tuto volbu, zprávy náležící do skupin lze načíst i v případě, že je skupina neúplná.

MQGMO_ALL_MSGS_AVAILABLE znamená MQGMO_ALL_SEGMENTS_AVAILABLE, které proto nemusí být zadány.

MQGMO_ALL_MSGS_AVAILABLE lze zadat s libovolnou z dalších voleb 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 objekt CFSTRUCT, na který má být mapa fronty mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na těchto platformách:
  - AIX AIX
  - IBM i IBM i
  - Linux Linux
  - Windows Windows

  a pro IBM MQ MQI clients připojené k těmto systémům je tato volba podporovaná pro všechny lokální fronty.

DOSTUPNÉ MQGMO_ALL_SEGMENTS_AVAILABLE
Segmenty v logické zprávě jsou k dispozici pro načtení pouze tehdy, jsou-li k dispozici všechny segmenty v logické zprávě. Pokud fronta obsahuje segmentované zprávy s některými chybějícími segmenty (například proto, že byly v síti zpožděny a dosud nedorazili), zadání MQGMO_ALL_SEGMENTS_AVAILABLE brání načtení segmentů náležejících k neúplným logickým zprávám. Tyto segmenty však stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné obnovitelné logické zprávy, i když je CurrentQDepth větší než nula. Nejsou-li k dispozici žádné další zprávy, které lze načíst, je po uplynutí zadané čekací doby (pokud existuje) vrácen kód příčiny MQRC_NO_MSG_AVAILABLE .

Zpracování příkazu MQGMO_ALL_SEGMENTS_AVAILABLE závisí na tom, zda je zadán také parametr MQGMO_LOGICAL_ORDER :

- Jsou-li zadány obě volby, má příkaz MQGMO_ALL_SEGMENTS_AVAILABLE efekt pouze v případě, že neexistuje žádná aktuální logická zpráva. Pokud se jedná o aktuální logickou zprávu, MQGMO_ALL_SEGMENTS_AVAILABLE se ignoruje. To znamená, že MQGMO_ALL_SEGMENTS_AVAILABLE může zůstat při zpracování zpráv v logickém pořadí zpracování.
- Je-li MQGMO_ALL_SEGMENTS_AVAILABLE zadán bez MQGMO_LOGICAL_ORDER, MQGMO_ALL_SEGMENTS_AVAILABLE má vždy efekt. To znamená, že tato volba musí být vypnuta po odebrání prvního segmentu z logické zprávy z fronty, aby bylo možné odebrat zbývající segmenty v logické zprávě.

Není-li tato volba uvedena, lze segmenty zpráv načíst i v případě, že je logická zpráva neúplná.

Zatímco MQGMO_COMPLETE_MSG i MQGMO_ALL_SEGMENTS_AVAILABLE vyžadují, aby všechny segmenty byly k dispozici před tím, než může být některý z nich načten, původní zpráva vrací úplnou zprávu, zatímco druhá umožňuje, aby byly segmenty načteny jeden po druhém.

Je-li pro zprávu sestavy zadán parametr MQGMO_ALL_SEGMENTS_AVAILABLE , správce front danou frontu zkontroluje a zjišťuje, zda pro každý ze segmentů, které tvoří úplnou logickou zprávu, je uvedena alespoň jedna zpráva sestavy. Pokud ano, je splněna podmínka MQGMO_ALL_SEGMENTS_AVAILABLE . Správce front však nekontroluje *typ* zpráv sestavy a tak může ve zprávách sestav týkajících se segmentů této logické zprávy existovat směs typů sestav. V důsledku toho úspěch MQGMO_ALL_SEGMENTS_AVAILABLE neznamená, že MQGMO_COMPLETE_MSG uspěje. Je-li pro segmenty určité logické zprávy uvedena směs typů sestav, musí být tyto zprávy sestavy načteny jedním po druhém.

Můžete uvést MQGMO_ALL_SEGMENTS_AVAILABLE s libovolní z ostatních voleb MQGMO_* a s libovolnou 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 objekt CFSTRUCT, na který má být mapa fronty mapována, musí být na úrovni CFLEVEL (3) nebo vyšší.
- Na těchto platformách:
  - AIX
  - IBM i
  - Linux
  - Windows

  a pro IBM MQ MQI clients připojené k těmto systémům je tato volba podporovaná 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 (či rozšíření) zprávy, by měly být představeny atributem fronty **PropertyControl** . Je-li zadána hodnota MsgHandle , je tato volba ignorována a vlastnosti zprávy jsou k dispozici prostřednictvím 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: Je-li zpráva později přečtena aplikací, která nevytvořila popisovač zprávy, umístí správce front všechny vlastnosti zprávy do struktury MQRFH2 . Možná zjistíte, že přítomnost neočekávaného záhlaví produktu MQRFH2 narušuje chování existující aplikace.

MQGMO_NO_PROPERTIES

Žádné vlastnosti zprávy, kromě těch, které jsou obsaženy v deskriptoru (nebo rozšíření) zprávy, budou načteny. Je-li zadán příznak MsgHandle , bude ignorován.

MQGMO_PROPERTIES_FORCE_MQRFH2

Vlastnosti zprávy s výjimkou těch, které jsou obsaženy v deskriptoru zprávy (nebo přípony), by měly být reprezentovány pomocí záhlaví MQRFH2 . Toto poskytuje kompatibilitu s dřívější verzí pro aplikace, které očekávají načtení vlastností, ale nemohou být změněny tak, aby používaly obslužné rutiny zpráv. Je-li zadán MsgHandle , je 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

Není-li 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. MQGMO_NONE pomáhá programovou dokumentaci; není určena, aby byla tato volba použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze zjistit.

Počáteční hodnota pole Options je MQGMO_NO_WAIT plus MQGMO_PROPERTIES_AS_Q_DEF.

WaitInterval (MQLONG)

Toto je přibližná doba, vyjádřená v milisekundách, po kterou volání MQGET čeká na příchod vhodné zprávy (to znamená zpráva splňující kritéria výběru zadaná v parametru **MsgDesc** volání MQGET.

Důležité: Pokud je okamžitě k dispozici vhodná zpráva, není zde žádná čekací doba nebo prodleva.

Další podrobnosti viz pole *MsgId* popsané v “MQMD-Deskriptor zpráv” na stránce 419 .) Pokud po uplynutí této doby neuplynula žádná vhodná zpráva, volání skončí s funkcí MQCC_FAILED a kódem příčiny MQRC_NO_MSG_AVAILABLE.

V systému z/OSje doba, po kterou volání MQGET skutečně čeká, ovlivněno systémem načítání a pokyny pro plánování práce, a může se lišit mezi hodnotou zadanou pro *WaitInterval* a přibližně 100 milisekund větší než *WaitInterval*.

WaitInterval se používá ve spojení s volbou MQGMO_WAIT nebo MQGMO_SET_SIGNAL. Pokud ani jedna z nich není určena, je ignorována. Je-li zadán jeden z těchto hodnot, musí být hodnota *WaitInterval* větší než nula nebo rovna nule nebo následující speciální hodnota:

MQWI_UNLIMITED
Neomezený interval čekání.

Počáteční hodnota tohoto pole je 0.

Signal1 (MQLONG)

Jedná se o vstupní pole, které se používá pouze ve spojení s volbou MQGMO_SET_SIGNAL; identifikuje signál, který má být doručen, když je k dispozici zpráva.

Poznámka: Datový typ a použití tohoto pole jsou určovány prostředím; z tohoto důvodu nemusí aplikace, které chcete portovat mezi různými prostředími, používat signály.

- V systému z/OSmusí toto pole obsahovat adresu prvku Event Control Block (ECB). ECB musí tuto žádost schválit před vydáním výzvy MQGET. Uskladnění obsahující ECB nesmí být uvolněno, dokud nebude fronta uzavřena. ECB je uveřejněna správcem front s jednou z popsaných kódů dokončení signálu. Tyto kódy dokončení jsou nastaveny v bitech 2 až 31 ECB, což je oblast definovaná v makru z/OS macro IHAECB jako pro kód dokončení uživatele.
- Ve všech ostatních prostředích se jedná o vyhrazené pole; jeho hodnota není významná.

Kódy dokončení signálu jsou:

MQEC_MSG_ARRIVED
Do fronty byla doručena vhodná zpráva. Tato zpráva nebyla rezervována pro volajícího; druhý požadavek MQGET musí být zadán, ale jiná aplikace může tuto zprávu načíst před tím, než bude proveden druhý požadavek.

MQEC_WAIT_INTERVAL_EXPIRED
Platnost zadaného *WaitInterval* vypršela, aniž by byla doručena vhodná zpráva.

ČEKÁNÍ MQEC_WAIT_CANCELED
Čekání bylo zrušeno z neurčeného důvodu (například ukončení správce front nebo znepřístupněný stav fronty). Chcete-li dále diagnostikovat, zadejte žádost znovu.

UVÁDĚNÍ MQEC_Q_MGR_QUIESCING
Čekání bylo zrušeno, protože správce front přešel do klidového stavu (MQGMO_FAIL_IF_QUIESCING byl zadán na volání MQGET).

FUNKCE MQEC_CONNECTION_QUIESCING
Čekání bylo zrušeno, protože připojení vstoupilo do stavu uvedení do klidového stavu (volání MQGMO_FAIL_IF_QUIESCING bylo určeno v rámci volání MQGET).

Počáteční hodnota tohoto pole je určena prostředím:

- V systému z/OSje počáteční hodnotou ukazatel Null.
- Ve všech ostatních prostředích je počáteční hodnota `0`.

Signal2 (MQLONG)

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)

Jedná se o výstupní pole, které správce front nastaví na lokální název fronty, ze které byla zpráva načtena, jak je definováno v lokálním správci front. To se liší od názvu použitého k otevření fronty, pokud:

- Byla otevřena fronta aliasů (v takovém případě se jedná o název lokální fronty, do které je alias vrácen), 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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

MatchOptions (MQLONG)

Tyto volby umožňují aplikaci vybrat, která pole v parametru **MsgDesc** se mají použít pro výběr zprávy vrácené voláním MQGET. Aplikace nastavuje 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 tuto zprávu, jsou kandidáty na načtení pomocí parametru **MsgDesc** na volání MQGET. Pole, pro která není zadána odpovídající volba shody, jsou při výběru zprávy, která má být vrácena, ignorována. Pokud nezadáte žádná kritéria výběru na volání MQGET (hodnota *libovolná* 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. Viz atribut fronty produktu **IndexType** , kde jsou další podrobnosti.

Zadáte-li MQGMO_LOGICAL_ORDER, budou pro návrat při dalším volání MQGET způsobilé pouze určité zprávy:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, mohou být vráceny pouze zprávy, které mají *`MsgSeqNumber`* rovnu `1` a *`Offset`* rovnající se `0` . V této situaci můžete použít jednu nebo více následujících voleb porovnání, abyste vybrali, která z vhodných zpráv se vrátí:
  - MQMO_MATCH_MSG_ID
  - MQMO_MATCH_CORREL_ID
  - MQMO_MATCH_GROUP_ID
- Pokud se jedná o aktuální skupinu nebo logickou zprávu, je možné vrátit pouze další zprávu ve skupině nebo další segment v logické zprávě a nelze ji změnit zadáním voleb MQMO_ *.

V obou předchozích případech můžete uvést volby shody, které se nepoužijí, ale hodnota relevantního pole v parametru **MsgDesc** musí odpovídat hodnotě odpovídajícího pole ve zprávě, která se má vrátit; volání selže s kódem příčiny MQRC_MATCH_OPTIONS_ERROR, že tato podmínka není splněna.

MatchOptions je ignorován, pokud uvedete buď MQGMO_MSG_UNDER_CURSOR nebo MQGMOROWS_MSG_UNDER_CURSOR.

Získávání zpráv založené na vlastnosti zprávy není prováděno pomocí voleb shody; další informace viz "SelectionString (MQCHARV)" na stránce 489.

Můžete uvést jednu nebo více z 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 týkat (například identifikátor korelace).

Vynecháte-li tuto volbu, pole *MsgId* v parametru **MsgDesc** se ignoruje a každý identifikátor zprávy se bude shodovat.

Poznámka: Identifikátor zprávy MQMI_NONE je speciální hodnota, která odpovídá libovolnému identifikátoru zprávy v deskriptoru MQMD pro zprávu. Proto uvedení MQMO_MATCH_MSG_ID s hodnotou MQMI_NONE je 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 aplikovat (například identifikátor zprávy).

Vynecháte-li tuto volbu, pole *CorrelId* v parametru **MsgDesc** se ignoruje a jakýkoli korelační identifikátor se bude shodovat.

Poznámka: Identifikátor korelace MQCI_NONE je speciální hodnota, která odpovídá hodnotě *any* identifikátoru korelace v deskriptoru MQMD pro zprávu. Proto uvedení hodnoty MQMO_MATCH_CORREL_ID s parametrem MQCI_NONE je 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 týkat (například identifikátor korelace).

Vynecháte-li tuto volbu, pole *GroupId* v parametru **MsgDesc** se ignoruje a jakýkoli identifikátor skupiny se bude shodovat.

Poznámka: Identifikátor skupiny MQGI_NONE je speciální hodnota, která odpovídá identifikátoru *any* identifikátoru skupiny v deskriptoru MQMD pro zprávu. Proto uvedení MQMO_MATCH_GROUP_ID s MQGI_NONE je 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 týkat (například identifikátor skupiny).

Vynecháte-li tuto volbu, pole *MsgSeqNumber* v parametru **MsgDesc** se ignoruje a jakékoli pořadové číslo zprávy se bude shodovat.

MQMO_MATCH_OFSET
Zpráva, která má být načtena, musí mít offsetu, který odpovídá hodnotě pole *Offset* v parametru **MsgDesc** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou aplikovat (například pořadové číslo zprávy).

Pokud vynecháte tuto volbu, nebude pole *Offset* v parametru **MsgDesc** ignorováno a všechny odchylky se budou 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* v rámci struktury MQGMO zadané ve volání MQGET.

Tuto volbu můžete zadat pro všechny lokální fronty. Pokud ji zadáte pro frontu, která má *IndexType* MQIT_MSG_TOKEN (fronta spravovaná WLM), můžete zadat žádné jiné volby shody s MQMO_MATCH_MSG_TOKEN.

Nemůžete uvést MQMO_MATCH_MSG_TOKEN s MQGMO_WAIT nebo MQGMO_SET_SIGNAL. Pokud chce aplikace čekat na příchod zprávy do fronty, která má *IndexType* MQIT_MSG_TOKEN, zadejte MQMO_NONE.

Vynecháte-li tuto volbu, bude pole *MsgToken* v produktu MQGMO ignorováno a každý token zprávy se bude shodovat.

Pokud neuvedete žádnou z popsaných voleb, můžete použít následující volbu:

MQMO_NONE
Při výběru zprávy, která má být vrácena, použijte žádné shody; všechny zprávy ve frontě jsou vhodné pro načtení (ale podléhají kontrole pomocí voleb MQGMO_ALL_MSGS_AVAILABLE, MQGMO_ALL_SEGMENTS_AVAILABLE a MQGMO_COMPLETE_MSG).

Dokumentace k programu podpory MQMO_NONE. Není určeno, aby byla tato volba použita s jinou volbou MQMO_ *, ale protože její hodnota je nula, takové použití nelze zjistit.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQMO_MATCH_MSG_ID s MQMO_MATCH_CORREL_ID. Toto pole je ignorováno, pokud *Version* je menší než MQGMO_VERSION_2.

Poznámka: Počáteční hodnota pole *MatchOptions* je definovaná pro kompatibilitu se staršími správci front MQSeries . Avšak při čtení posloupnosti zpráv z fronty bez použití kritérií výběru tato počáteční hodnota vyžaduje, aby aplikace resetoval pole *MsgId* a *CorrelId* na hodnotu MQMI_NONE a MQCI_NONE před každým voláním MQGET. Vyvarovat se nutnosti resetovat *MsgId* a *CorrelId* nastavením *Version* na MQGMO_VERSION_2a *MatchOptions* na MQMO_NONE.

Související pojmy
Selektory zpráv v JMS

GroupStatus (MQCHAR)
Tento příznak označuje, zda je načtená zpráva ve skupině.

Má jednu z následujících hodnot:

MQGS_NOT_IN_GROUP
Zpráva se nenachází ve skupině.

MQGS_MSG_IN_GROUP
Zpráva se nachází ve skupině, ale není poslední ve skupině.

MQGS_LAST_MSG_IN_GROUP
Zpráva je poslední ve skupině.

Tato hodnota je také vrácena, 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 *Version* je menší než MQGMO_VERSION_2.

SegmentStatus (MQCHAR)

Jedná se o příznak, který označuje, zda načtená zpráva je segmentem logické zprávy. Má jednu z následujících hodnot:

SEGMENT MQSS_NOT_SEGMENT
Zpráva není segment.

SEGMENT MQSS_SEGMENT
Zpráva je segment, ale nejedná se o poslední segment logické zprávy.

MQSS_LAST_SEGMENT
Zpráva je posledním segmentem logické zprávy.

Tato hodnota je také vrácena, pokud se logická zpráva skládá pouze z jednoho segmentu.

V systému z/OSsprávce front vždy nastaví toto pole na hodnotu MQSS_NOT_A_SEGMENT.

Toto je výstupní pole. Počáteční hodnota tohoto pole je MQSS_NOT_A_SEGMENT. Toto pole je ignorováno, pokud *Version* je menší než MQGMO_VERSION_2.

Segmentace (MQCHAR)

Jedná se o příznak, který označuje, zda je pro načtenou zprávu povolena další segmentace. Má jednu z následujících hodnot:

MQSEG_BLOKOVÁNO
Segmentace není povolena.

MQSEG_ALLOWED
Segmentace je povolena.

V systému z/OSsprávce front vždy nastaví toto pole na hodnotu MQSEG_INHIBITED.

Toto je výstupní pole. Počáteční hodnota tohoto pole je MQSEG_INHIBITED. Toto pole je ignorováno, pokud *Version* je menší než MQGMO_VERSION_2.

Reserved1 (MQCHAR)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud *Version* je menší než MQGMO_VERSION_2.

MsgToken (MQBYTE16)

Pole MsgToken -struktura MQGMO. Toto pole je používáno správcem front k jedinečné identifikaci zprávy.

Jedná se o bajtový řetězec, který je generovaný správcem front pro jedinečnou identifikaci zprávy ve frontě. Token zpráv je generován při prvním umístění zprávy do správce front a zůstává se zprávou, dokud není zpráva trvale odebrána ze správce front, pokud nebude restartován správce front.

Když je zpráva odebrána z fronty, *MsgToken* , která zjistila, že instance zprávy již není platná, a nikdy se znovu nepoužije. Je-li správce front restartován, nemusí být po restartu serveru *MsgToken* , který označil zprávu ve frontě před restartováním, platný. Avšak *MsgToken* se nikdy znovu nepoužije k identifikaci jiné instance zprávy. Produkt *MsgToken* je generován správcem front a není viditelný pro žádnou externí aplikaci.

Je-li zpráva vrácena voláním MQGET, kde je dodána MQGMO verze 3 nebo vyšší, *MsgToken* správce front vrací zprávu označující zprávu ve frontě ve frontě MQGMO. Existuje jedna výjimka: když je zpráva odebrána z fronty mimo synchronizační bod, správce front nemůže vrátit *MsgToken* , protože není užitečný k identifikaci vrácené zprávy při následném volání MQGET. Aplikace by měly používat produkt *MsgToken* pouze k odkazování na zprávu při následných voláních MQGET.

Je-li zadán příkaz *MsgToken* a je zadán parametr *MatchOption* MQMO_MATCH_MSG_TOKEN a není zadán ani MQGMO_MSG_UNDER_CURSOR, ani MQGMO_BROWS_MSG_UNDER_CURSOR, lze vrátit pouze zprávu identifikovanou argumentem *MsgToken* . Volba je platná ve všech lokálních frontách bez ohledu na INDXTYPE a na z/OS musíte použít parametr INDXTYPE (MSGTOKEN) pouze ve frontách správy zátěže (WLM).

Je zkontrolováno jakékoli jiné zadané *MatchOptions* a pokud se neshodují, je vrácen příkaz MQRC_NO_MSG_AVAILABLE. Je-li kód MQGMO_BE NEXT kódován pomocí MQMO_MATCH_MSG_TOKEN, je zpráva označená *MsgToken* vrácena pouze v případě, že je za volajícím hantem za kurzorem procházení.

Je-li zadán parametr MQGMO_MSG_UNDER_CURSOR nebo MQGMO_BIT_MSG_UNDER_CURSOR, bude MQMO_MATCH_MSG_TOKEN ignorován.

MQMO_MATCH_MSG_TOKEN není platný s následujícími volbami získání zprávy:

- MQGMO_WAIT
- SIGNÁL MQGMO_SET_DATA

Pro volání MQGET s uvedením MQMO_MATCH_MSG_TOKEN musí být k volání předán objekt MQGMO verze 3 nebo novější, jinak se vrátí MQRC_WRONG_GMO_VERSION.

Pokud *MsgToken* není momentálně platný, vrací se MQCC_FAILED s MQRC_NO_MSG_AVAILABLE, pokud neexistuje jiná chyba.

ReturnedLength (MQLONG)

Jedná se o výstupní pole, které správce front nastaví na délku v bajtech dat zprávy vrácených voláním MQGET v rámci parametru **Buffer** . Pokud správce front tuto schopnost nepodporuje, je hodnota *ReturnedLength* nastavena na hodnotu MQRL_UNDEFINED.

Jsou-li zprávy převáděny mezi kódováními nebo znakovými sadami, mohou data zprávy někdy měnit velikost. Při návratu z volání MQGET:

- Pokud *ReturnedLength* není MQRL_UNDEFINED, počet bajtů vrácených dat zprávy je dán systémem *ReturnedLength*.
- Má-li parametr *ReturnedLength* hodnotu MQRL_UNDEFINED, je počet bajtů vrácených dat zprávy 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, jsou nevýznamné bajty v parametru **Buffer** 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 *Version* je menší než MQGMO_VERSION_3.

Reserved2 (MQLONG)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud *Version* je menší než **MQGMO_VERSION_4**.

MsgHandle (MQHMSG)

Je-li zadána volba MQGMO_PROPERTIES_AS_Q_DEF a atribut fronty `PropertyControl` není nastaven na hodnotu MQPROP_FORCE_MQRFH2 , jedná se o manipulátor zprávy, který bude naplněn vlastnostmi zprávy načítané z fronty. Popisovač je vytvořen voláním MQCRTMH. Všechny vlastnosti, které jsou již přidruženy k popisovači, budou před načtením zprávy vymazány.

Je možné zadat také následující hodnotu:
MQM_NONE
Nebyl zadán popisovač zprávy.

Pokud je zadán platný popisovač zprávy a ve výstupu obsahuje vlastnosti zprávy, není na volání MQGET vyžadován žádný deskriptor zprávy. Pro vstupní pole se použije deskriptor zprávy přidružený k popisovači zpráv.

Je-li v rámci volání MQGET zadán deskriptor zprávy, má vždy přednost před deskriptorem zpráv přidruženým k manipulátoru zprávy.

Je-li zadán parametr MQGMO_PROPERTIES_FORCE_MQRFH2 nebo je zadán parametr MQGMO_PROPERTIES_AS_Q_DEF a atribut fronty `PropertyControl` má hodnotu 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 zpráv aktualizovány tak, aby odrážely stav načtené zprávy (stejně jako deskriptor zprávy, pokud byl dodán na volání MQGET). Vlastnosti této zprávy lze poté provést zjišťování pomocí volání MQINQMP.

S výjimkou rozšíření deskriptoru zpráv, je-li přítomna vlastnost, která může být inquired s voláním MQINQMP, není obsažena v datech zprávy; pokud zpráva ve frontě obsahuje vlastnosti v datech zprávy, tyto jsou odebrány z dat zprávy před tím, než se data vrátí do aplikace.

Pokud není poskytnut žádný popisovač zprávy nebo je verze nižší než MQGMO_VERSION_4 , je třeba zadat platný deskriptor zprávy pro volání MQGET. Všechny vlastnosti zprávy (s výjimkou těch, které jsou obsaženy v deskriptoru zpráv) jsou vráceny v datech zprávy pod hodnotou volby vlastností ve struktuře MQGMO a atributu fronty produktu `PropertyControl` .

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQHM_NONE. Toto pole je ignorováno, pokud `Version` je 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 498. 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 404)	Není	0
CodedCharSetId (vyhrazeno-viz “Znaková sada a kódování” na stránce 404)	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)

Jedná se o identifikátor struktury. Hodnota musí být:

MQIIH_STRUCTION_ID
Identifikátor pro strukturu záhlaví informací produktu IMS .

Pro programovací jazyk C je také definována konstanta MQIIH_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQIIH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQIIH_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQIIH_VERSION_1
Číslo verze pro strukturu záhlaví informací produktu IMS .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQIIH_VERSION
Aktuální verze struktury záhlaví informací produktu IMS .

Počáteční hodnota tohoto pole je MQIIH_VERSION_1.

StrucLength (MQLONG)

Toto je délka struktury MQIIH. Hodnota musí být:

MQIIH_LENGTH_1
Délka struktury záhlaví informací produktu IMS .

Počáteční hodnota tohoto pole je MQIIH_LENGTH_1.

Kódování (MQLONG)

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

Kódování pro podporované struktury, které následují strukturu MQIIH, je stejné jako struktura MQIIH samotné struktury MQIIH a převzata z libovolného předchozího záhlaví MQ .

CodedCharSetId (MQLONG)

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

ID znakové sady pro podporované struktury, které postupují podle struktury MQIIH, je stejné jako u struktury MQIIH a převzato z jakéhokoli předchozího záhlaví MQ .

Formát (MQCHAR8)

Určuje název formátu produktu MQ pro data, která následují za strukturou MQIIH.

Na základě 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)

Hodnota příznaků musí být:

MQIIH_NONE
Žádné vlajky.

MQIIH_PASS_EXPIRATION
Zpráva odpovědi obsahuje:

- Stejné volby sestavy vypršení platnosti jako zpráva požadavku
- Zbývající doba vypršení platnosti ze zprávy požadavku bez úpravy provedené pro dobu zpracování mostu

Není-li tato hodnota nastavena, je doba vypršení platnosti nastavena na hodnotu *unlimited*(neomezeno).

MQIIH_REPLY_FORMAT_NONE
Nastavuje hodnotu MQIIH.Format pole odpovědi na MQFMT_NONE.

MQIIH_IGNORE_PURG
Nastaví indikátor TMAMIPRG v rámci předpony OTMA, který požaduje, aby OTMA ignorovala volání PURG na TP PCB pro transakce CM0 .

MQIIH_CM0_REQUEST_RESPONSE
Pro režim vázaného zpracování 0 (CM0) tento parametr nastavuje indikátor TMAMHRSP v předponě OTMA. Nastavení tohoto indikátoru vyžaduje, aby OTMA/IMS vygenerovala zprávu DFS2082 RESPONSE MODE TRANSACTION TERMINATED WITHOUT REPLY, když původní aplikační program produktu IMS neodpověděl na IOPCB ani přepínač zpráv na jinou transakci.

Počáteční hodnota tohoto pole je MQIIH_NONE.

LTermOverride (MQCHAR8)

Přepis logického terminálu, umístěný v poli IO PCB. Je volitelný; není-li uveden, použije se název TPIPE. Je ignorován, 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)

Název mapy služeb formátu zprávy, umístěný v poli IO PCB. Tato položka není povinná. Na vstupu se jedná o MID, na výstupu, který představuje MOD. Je ignorován, 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.

Formát ReplyTo(MQCHAR8)

Jedná se o název formátu MQ zprávy odpovědi, která je odeslána jako odezva 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 buď hodnotu MQIIH.replyToFormat= MQFMT_STRING, nebo MQIIH.replyToFormat= MQFMT_IMS_VAR_STRING. Vysvětlení použití těchto polí najdete v tématu “Formát (MQCHAR8)” na stránce 444.

Je-li výchozí hodnota (MQIIH.replyToFormat= MQFMT_NONE) použita ve zprávě požadavku a zpráva odpovědi je načtena pomocí MQGMO_CONVERT, nebude provedena žádná konverze dat.

Ověřovatel (MQCHAR8)

Jedná se o heslo produktu RACF nebo PassTicket. Je volitelný; je-li zadán, použije se s ID uživatele v kontextu zabezpečení MQMD k sestavení UTOKEN, které je odesláno do produktu IMS za účelem poskytnutí kontextu zabezpečení. Není-li zadán, použije se ID uživatele bez ověření. Závisí to na nastavení přepínačů RACF , což může vyžadovat přítomnost ověřovatele.

Tato hodnota je ignorována, pokud je první bajt prázdný nebo má hodnotu null. Je možné použít následující speciální hodnotu:

MQIAUT_NONE
Žádné ověření.

Pro programovací jazyk C je také definována konstanta MQIAUT_NONEE_ARRAY; má stejnou hodnotu jako MQIAUT_NONE, ale je to pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota tohoto pole je MQIAUT_NONE.

ID TranInstance(MQBYTE16)

Jedná se o identifikátor instance transakce. Toto pole je používáno výstupními zprávami z IMS, takže je při prvním vstupu ignorován. Pokud jste nastavili *TranState* na hodnotu MQITS_IN_CONVERSATION, musí být tato hodnota poskytnuta na dalším vstupu a všechny následné vstupy, aby bylo možné IMS korelovat se zprávami na správnou konverzaci. Můžete použít následující speciální hodnotu:

MQITII_NONE
Žádný identifikátor instance transakce.

Pro programovací jazyk C je také definována konstanta MQITII_NONE_ARRAY; má stejnou hodnotu jako MQITII_NONE, ale je to pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_TRAN_INSTANCE_IDLENGTH. Počáteční hodnota tohoto pole je MQITII_NONE.

TranState (MQCHAR)

Tento stav označuje stav konverzace produktu IMS . Tato hodnota je na prvním vstupu ignorována, protože žádná konverzace neexistuje. Na následných vstupech označuje, zda je konverzace aktivní nebo ne. Na výstupu je nastaven pomocí IMS. Hodnota musí být jedna z následujících:

MQITS_IN_CONVERSATION
-V rozhovoru.

MQITS_NOT_IN_CONVERSATION
Ne v rozhovoru.

MQITS_ARCHITECTED
Vrátit data stavu transakce ve formě architektury.

Tato hodnota se používá pouze s příkazem IMS /DISPLAY TRAN . Místo znakového formuláře vrací data stavu transakce ve formě architektury IMS . z/OS Další informace najdete v tématu Zápis transakčních programů IMS prostřednictvím produktu IBM MQ.

Počáteční hodnota tohoto pole je MQITS_NOT_IN_CONVERSATION.

CommitMode (MQCHAR)

Jedná se o režim vázaného zpracování IMS . Další informace o režimech potvrzování produktu IMS naleznete v příručce *OTMA Reference* . Hodnota musí být jedna z následujících:

MQICM_COMMIT_THEN_SEND
Potvrdit poté odeslání.

Tento režim implikuje dvojité řazení výstupu, ale kratší doba obsazenosti oblasti. Rychlá cesta a konverzační transakce nemohou být spuštěny s tímto režimem.

MQICM_SEND_THEN_COMMIT
Odeslat a potvrdit.

Každá transakce IMS zahájená jako výsledek režimu vázaného zpracování MQICM_SEND_COMMIT se spustí v režimu RESPONSE, bez ohledu na to, jak je transakce definována v definici systému IMS (parametr MSGTYPE v makru TRANSACT). Toto platí také pro transakce zahájené přepínačem transakce.

Počáteční hodnota tohoto pole je MQICM_COMMIT_THEN_SEND.

SecurityScope (MQCHAR)

To označuje, že je požadováno zpracování zabezpečení produktu IMS . Jsou definovány tyto hodnoty:

KONTROLA MQISS_CHECK
Zkontrolujte rozsah zabezpečení: ACEE je postaven v řídicí oblasti, ale ne v závislé oblasti.

MQISS_FULL
Plný rozsah zabezpečení: ACEE uložený v mezipaměti je sestavován v řídicí oblasti a ACEE bez mezipaměti je sestaven v závislé oblasti. Používáte-li produkt MQISS_FULL, zkontrolujte, zda má ID uživatele, pro který je objekt ACEE zabudován, přístup k prostředkům používaným v závislé oblasti.

Pokud není zadán parametr MQISS_CHECK ani MQISS_FULL pro toto pole, předpokládá se hodnota MQISS_CHECK.

Počáteční hodnota tohoto pole je MQISS_CHECK.

Vyhrazeno (MQCHAR)

Jedná se o 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 499. 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)

Dotaz na strukturu voleb vlastností zprávy-pole StrucId

Jedná se o identifikátor struktury. Hodnota musí být:

MQIMPO_STRUCT
Identifikátor pro strukturu voleb vlastností zprávy dotazu.

Pro programovací jazyk C je také definována konstanta MQIMPO_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQIMPO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQIMPO_STRUC_ID.

Verze (MQLONG)

Dotaz na strukturu vlastností vlastností zprávy-pole Verze

Jedná se o číslo verze struktury. Hodnota musí být:

MQIMPO_VERSION_1
Číslo verze pro strukturu voleb vlastností zprávy dotazu.

Následující konstanta uvádí číslo verze aktuální verze:

VERZE AKTUÁLNÍ_VERZE MQIMPO_CURRENT_VERSION
Aktuální verze struktury voleb vlastností dotazových zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQIMPO_VERSION_1.

Volby (MQLONG)

Dotaz na strukturu voleb vlastností zprávy-pole Volby

Následující volby řídí akci MQINQMP. Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace voleb, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné.

Volby hodnot dat: Následující volby se vztahují ke zpracování dat hodnoty, když je vlastnost načtena ze zprávy.

HODNOTA MQIMPO_CONVERT_VALUE
Tato volba vyžaduje, aby hodnota vlastnosti byla převedena tak, aby odpovídala hodnotám *RequestedCCSID* a *RequestedEncoding* určeným před voláním MQINQMP vrací hodnotu vlastnosti v oblasti *Value* .

- Je-li konverze úspěšná, jsou pole *ReturnedCCSID* a *ReturnedEncoding* nastavena na stejné hodnoty jako *RequestedCCSID* a *RequestedEncoding* při návratu z volání MQINQMP.
- Pokud převod selže, ale volání MQINQMP se jinak dokončí bez chyby, hodnota vlastnosti se vrátí nekonvertované.

  Je-li vlastnost řetězec, jsou pole *ReturnedCCSID* a *ReturnedEncoding* nastavena na znakovou sadu a kódování nepřeváděné řetězce.

  Kód dokončení je MQCC_WARNING v tomto případě, s kódem příčiny MQRC_PROP_VALUE_NOT_CONVERTED. Kurzor vlastností se zálohuje na vrácenou vlastnost.

Pokud se hodnota vlastnosti rozbalí během převodu a překročí velikost parametru **Value** , hodnota se vrátí nekonvertovaný s kódem dokončení MQCC_FAILED; kód příčiny je nastaven na hodnotu MQRC_PROPERTY_VALUE_TOO_BIG.

Parametr **DataLength** volání MQINQMP vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

Tato volba také vyžaduje, aby:

- Pokud název vlastnosti obsahuje zástupný znak, a
- Pole *ReturnedName* je inicializováno s adresou nebo offsetem pro vrácený název,

pak je vrácený název převeden tak, aby odpovídal hodnotám *RequestedCCSID* a *RequestedEncoding* .

- Je-li konverze úspěšná, jsou pole *VSCCSID* souboru *ReturnedName* a kódování vráceného názvu nastaveny na vstupní hodnotu *RequestedCCSID* a *RequestedEncoding*.
- Pokud převod selže, ale volání MQINQMP se jinak dokončí bez chyby nebo varování, vrácené jméno se nekonvertuje. Kód dokončení je MQCC_WARNING v tomto případě, s kódem příčiny MQRC_PROP_NAME_NOT_CONVERTED.

  Kurzor vlastností se zálohuje na vrácenou vlastnost. Hodnota MQRC_PROP_VALUE_NOT_CONVERTED je vrácena v případě, že hodnota i název nejsou převedeny.

Pokud se vrácený název rozbalí během převodu a překročí velikost pole *VSBufsize* v poli *RequestedName*, vrácený řetězec zůstane nekonvertovaný, kód dokončení MQCC_FAILED a kód příčiny je nastaven na hodnotu MQRC_PROPERTY_TOOPO_BIG.

Pole *VSLength* struktury MQCHARV vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

TYP MQIMPO_CONVERT_TYPE

Tato volba vyžaduje převedení hodnoty vlastnosti z aktuálního datového typu do datového typu zadaného v parametru **Type** volání MQINQMP.

- Je-li konverze úspěšná, parametr **Type** se nezmění při návratu volání MQINQMP.
- Pokud konverze selže, ale volání MQINQMP se jinak dokončí bez chyby, volání selže s příčinou MQRC_PROP_CONV_NOT_SUPPORTED. Kurzor vlastnosti se nemění.

Pokud konverze datového typu způsobí, že se hodnota během konverze rozšíří a převedená hodnota překročí velikost parametru **Value** , hodnota se vrátí nekonvertovaný, kód dokončení MQCC_FAILED a kód příčiny je nastaven na hodnotu MQRC_PROPERTY_VALUE_TOO_BIG.

Parametr **DataLength** volání MQINQMP vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

Není-li hodnota parametru **Type** volání MQINQMP platná, volání selže s příčinou MQRC_PROPERTE_ERROR.

Není-li požadovaná konverze typu dat podporována, volání selže s příčinou MQRC_PROP_CONV_NOT_SUPPORTED. Jsou podporovány následující převody datových typů:

Tabulka 500. Podporované konverze datových typů

Datový typ vlastnosti	Podporované cílové datové typy
LOGICKÁ HODNOTA MQTYPE_BOOLEAN	MQTYPE_STRING, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64
ŘETĚZEC MQTYPE_BYTE_STRING	ŘETĚZEC MQTYPE_STRING

Tabulka 500. Podporované konverze 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	ŘETĚZEC MQTYPE_STRING
MQTYPE_FLOAT32	MQTYPE_STRING, MQTYPE_FLOAT64
MQTYPE_FLOAT64	ŘETĚZEC MQTYPE_STRING
ŘETĚZEC MQTYPE_STRING	MQTYPE_BOOLEAN, MQTYPE_INT8, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64, MQTYPE_FLOAT32, MQTYPE_FLOAT64
MQTYPE_NULL	Není

Obecná pravidla týkající se podporovaných převodů jsou následující:

- Hodnoty číselných vlastností lze převádět z jednoho datového typu do jiného, za předpokladu, že během převodu nebudou ztracena žádná data.

  Např. hodnota vlastnosti s datovým typem MQTYPE_INT32 může být převedena na hodnotu s datovým typem MQTYPE_INT64, ale nelze ji převést na hodnotu s typem dat MQTYPE_INT16.
- Hodnotu vlastnosti libovolného datového typu lze převést na řetězec.
- Hodnotu vlastnosti řetězce lze převést na jakýkoli jiný typ dat za předpokladu, že je řetězec správně formátován pro převod. Pokusí-li se aplikace převést hodnotu vlastnosti řetězce, která není správně naformátována, 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 do jiného 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" se převede na FALSE.

  Všimněte si, že výrazy "TRUE" a "FALSE" nejsou citlivé na velikost písmen.

  Jakýkoli jiný řetězec nelze převést; produkt 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 tento řetězec následující formát:

```
[blanks][sign]digits
```

Význam komponent řetězce je následující:

blanks
Volitelné úvodní prázdné znaky

sign
Volitelné znaménko plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být přítomen alespoň jeden číselný znak.

Po pořadí znaků číslic může řetězec obsahovat i jiné znaky, které nejsou číslice, ale konverze se zastaví, jakmile je dosaženo začátku těchto znaků. Předpokládá se, že řetězec představuje desítkové 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 tento řetězec následující formát:

```
[blanks][sign]digits[.digits][e_char[e_sign]e_digits]
```

Význam komponent řetězce je následující:

blanks
Volitelné úvodní prázdné znaky

sign
Volitelné znaménko plus (+) nebo znak minus (-).

digits
Souvislá posloupnost číselných znaků (0-9). Musí být přítomen alespoň jeden číselný znak.

e_char
Exponent znak, který je buď "E" nebo "e".

e_sign
Volitelný znak plus (+) nebo znaménko minus (-) pro exponent.

e_digits
Souvislá posloupnost znaků číslic (0-9) pro exponent. Pokud řetězec obsahuje exponent exponent, musí být přítomen alespoň jeden znak číslice.

Po pořadí znaků číslic nebo volitelných znaků představujících exponent může řetězec obsahovat jiné znaky, které nejsou číslice, ale konverze se zastaví, jakmile se dosáhne první z těchto znaků. Předpokládá se, že řetězec představuje desetinné číslo s plovoucí 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 číselné hodnoty vlastnosti na řetězec se hodnota převede na řetězcovou reprezentaci hodnoty jako dekadické číslo, nikoli řetězec obsahující znak ASCII pro tuto hodnotu. Například, celé číslo 65 je převedeno na řetězec "65", nikoli řetězec "A".
- Při převádění hodnoty vlastnosti řetězce bajtu na řetězec se každý bajt převede na dva hexadecimální znaky, které představují bajt. Příklad: 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 se nevrátí.

Je-li zadána vyrovnávací paměť **ReturnedName** , pole *VSLength* struktury MQCHARV se vyplní s délkou názvu vlastnosti. Název vlastnosti není vrácen.

Volby iterace: Následující volby se vztahují k iteraci přes vlastnosti pomocí názvu se zástupným znakem

MQIMPO_INQ_FIRST
Zjišťuje se první vlastnost, která odpovídá uvedenému názvu. Po tomto volání je kurzor založen na vlastnosti, která je vrácena.

Toto je výchozí hodnota.

Volba MQIMPO_INQ_PROP_UNDER_CURSOR může být následně použita s voláním MQINQMP, je-li to nutné, aby se mohla znovu dotázat na stejnou vlastnost.

Všimněte si, že existuje pouze jeden kurzor vlastnosti; proto, je-li název vlastnosti uvedený ve volání MQINQMP, změny kurzoru se resetují.

Tato volba není platná při jedné z následujících voleb:

MQIMPO_INQ_NEXT
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_NEXT
Zvodí na další vlastnosti, která odpovídá uvedenému názvu, pokračuje hledání od kurzoru vlastnosti. Kurzor se přesune na vrácenou vlastnost.

Jedná-li se o první volání MQINQMP pro zadaný název, bude vrácena první vlastnost, která odpovídá zadanému názvu.

Volba MQIMPO_INQ_PROP_UNDER_CURSOR lze následně použít s voláním MQINQMP, je-li to nutné, a dotázat se na stejnou vlastnost znovu.

Pokud byla vlastnost pod kurzorem odstraněna, funkce MQINQMP vrátí následující odpovídající vlastnost za hodnotou, která byla odstraněna.

Je-li přidána vlastnost, která odpovídá zástupnému znaku, zatímco iterace probíhá, vlastnost může nebo nemusí být vrácena během dokončení iterace. Vlastnost je vrácena, jakmile se iterace restartuje pomocí struktury MQIMPO_INQ_FIRST.

Vlastnost odpovídající zástupnému znaku, který byl odstraněn, zatímco iterace probíhal, není po jejím odstranění vrácena.

Tato volba není platná při jedné z následujících voleb:

MQIMPO_INQ_FIRST
MQIMPO_INQ_PROP_UNDER_CURSOR

MQIMPO_INQ_PROP_UNDER_CURSOR
Načtení hodnoty vlastnosti, na kterou ukazuje kurzor, který je uveden ve vlastnosti. Vlastnost, na kterou ukazuje vlastnost kurzor, je ta, která byla naposledy dotazovaná, pomocí volby MQIMPO_INQ_FIRST nebo MQIMPO_INQ_NEXT.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy, když je zadán popisovač zprávy v poli *MsgHandle* MQGMO na volání MQGET nebo pokud je popisovač zprávy zadán v polích *OriginalMsgHandle* nebo *NewMsgHandle* ve struktuře MQPMO na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud vytvořen nebo byla-li vlastnost, na kterou ukazuje kurzor, byla odstraněna, volání se nezdaří s kódem dokončení MQCC_FAILED a příčinou je MQRC_PROPERTY_NOT_AVAILABLE.

Tato volba není platná při jedné 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.

Program MQIMPO_NONE opomáhá dokumentaci programu; není určeno, že tato volba bude použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití však nelze zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQIMPO_INQ_FIRST.

RequestedEncoding (MQLONG)
Dotaz na strukturu voleb vlastností zprávy-pole RequestedEncoding

Jedná se o kódování, do kterého se má dotazovaná hodnota vlastnosti převádět, když je zadán parametr MQIMPO_CONVERT_VALUE nebo MQIMPO_CONVERT_TYPE.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

RequestedCCSID (MQLONG)
Dotaz na strukturu voleb vlastností zprávy-pole RequestedCCSID

Znaková sada, do které se má dotazovaná hodnota vlastnosti převést, je-li hodnota znakový řetězec. Jedná se také o znakovou sadu, do níž má být program *ReturnedName* převeden, je-li zadán parametr MQIMPO_CONVERT_VALUE nebo MQIMPO_CONVERT_TYPE.

Počáteční hodnota tohoto pole je MQCCSI_APPL.

ReturnedEncoding (MQLONG)
Dotaz na strukturu voleb vlastností zprávy-pole ReturnedEncoding

Na výstupu se jedná o kódování vrácené hodnoty.

Je-li zadána volba MQIMPO_CONVERT_VALUE a převod byl úspěšný, pole *ReturnedEncoding* při návratu má stejnou hodnotu jako hodnota předaná v poli.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

ReturnedCCSID (MQLONG)
Dotaz na strukturu voleb vlastností zprávy-pole ReturnedCCSID

Na výstupu se jedná o znakovou sadu hodnoty vrácené v případě, že parametr **Type** volání MQINQMP je MQTYPE_STRING.

Je-li zadána volba MQIMPO_CONVERT_VALUE a převod byl úspěšný, pole *ReturnedCCSID* při návratu má stejnou hodnotu jako hodnota předaná v poli.

Počáteční hodnota tohoto pole je nula.

Reserved1 (MQCHAR)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak (4 bajtové pole).

ReturnedName (MQCHARV)
Dotazovat strukturu voleb vlastností zprávy-pole ReturnedName

Aktuální název dotazované vlastnosti.

Na vstupu lze vyrovnávací paměť typu string předat pomocí pole *VSPtr* nebo *VSOffset* struktury MQCHARV . Délka vstupní vyrovnávací paměti řetězce je určena pomocí pole *VSBufsize* struktury MQCHARV.

Při návratu z volání MQINQMP je vyrovnávací paměť řetězce dokončena s názvem neurčené vlastnosti, za předpokladu, že vyrovnávací paměť řetězce byla dostatečně dlouhá, aby plně obsahovala název. Pole *VSLength* struktury MQCHARV se vyplní s délkou názvu vlastnosti. Pole *VSCCSID* struktury MQCHARV je vyplněno, aby byla uvedena znaková sada vráceného názvu bez ohledu na to, zda došlo k selhání převodu názvu či nikoli.

Jedná se o vstupní/výstupní pole. Počáteční hodnota tohoto pole je MQCHARV_DEFAULT.

TypeString (MQCHAR8)
Dotaz na strukturu voleb vlastností zprávy-pole TypeString

Řetězcová reprezentace datového typu vlastnosti.

Pokud byla vlastnost zadána v záhlaví MQRFH2 a atribut MQRFH2 `dt` není rozpoznán, lze toto pole použít k určení datového typu vlastnosti. *`TypeString`* je vrácen v kódované znakové sadě 1208 (UTF-8) a je prvních osm bajtů hodnoty atributu `dt` vlastnosti, které se nezdařilo rozpoznat

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je řetězec s hodnotou null 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 504 na stránce 470.

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 613 .

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 501. 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

Tabulka 501. Pole v MQMD pro MQMD (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
CodedCharSetId (identifikátor znakové sady dat zprávy)	MQCCSI_Q_MGR	0
Formát (název formátu dat zprávy)	MQFMT_NONE	Mezery
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 501. 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)

Jedná se o identifikátor struktury a musí být:

ID_STRUKTURY MQM_STRUCT
Identifikátor pro strukturu deskriptoru zpráv.

Pro programovací jazyk C je také definována konstanta MQMD_STRUC_ID_ARRAY; má stejnou hodnotu jako MQMD_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMD_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury a musí být jedna z následujících:

MQMD_VERSION_1
Struktura deskriptoru zpráv Version-1 .

Tato verze je podporována ve všech prostředích.

MQMD_VERSION_2
Struktura deskriptoru zpráv Version-2 .

Tato verze je podporována ve všech prostředích IBM MQ V6.0 a novějších, plus IBM MQ MQI clients připojených k těmto systémům.

Poznámka: Při použití version-2 MQMD provádí správce front 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 poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

VERZE MQM_AKTUÁLNÍ_VERZE
Aktuální verze struktury deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMD_VERSION_1.

Sestava (MQLONG)

Zpráva sestavy je zpráva o jiné zprávě, která se používá k informování aplikace o očekávaných nebo neočekávaných událostech, které se vztahují k původní zprávě. Pole *Report* umožňuje aplikaci odesláním původní zprávy určit, které zprávy sestavy jsou povinné, zda mají být data zprávy aplikace zahrnuta do nich, a také (pro sestavy i odpovědi), jak mají být nastaveny zprávy a identifikátory korelace v sestavě nebo zprávě odpovědi. Je možné požadovat libovolný nebo žádný (nebo žádný) z následujících typů zpráv sestavy:

- Výjimka

- Konec platnosti
- Potvrdit při příchodu (COA)
- Potvrdit při doručení (COD)
- Pozitivní upozornění na akci (PAN)
- Negativní upozornění na akci (NAN)

Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Aplikace, která přijímá zprávu sestavy, může určit příčinu, proč byla sestava generována, tak, že prozkoumáte pole *Feedback* v MQMD; další podrobnosti viz pole *Feedback* .

Použití voleb sestavy při vkládání zprávy do tématu může způsobit vygenerování a odeslání zpráv sestavy s nulovým počtem zpráv do aplikace. Důvodem je skutečnost, že zpráva o publikování může být odeslána na nulu, jednu nebo více odebírajících aplikací.

Volby výjimky: Určete jednu z uvedených voleb pro vyžádání zprávy hlášení výjimek.

VÝJIMKA MQRO_EXCEPTION
Agent kanálu zpráv generuje tento typ sestavy při odeslání zprávy do jiného správce front a tuto zprávu nelze doručit do zadané cílové fronty. Například cílová fronta nebo intermediační přenosová fronta může být plná, nebo může být zpráva příliš velká pro frontu.

Generování zprávy o výjimce závisí na perzistenci původní zprávy a na rychlosti kanálu zpráv (normální nebo rychlé), přes kterou se původní zpráva pohybuje:

- Pro všechny trvalé zprávy a pro přechodné zprávy, které cestují prostřednictvím běžných kanálů zpráv, se sestava výjimek generuje pouze v případě, že akce určená odesílající aplikací pro chybový stav může být úspěšně dokončena. Odesílající aplikace může určit jednu z následujících akcí k řízení dispozice původní zprávy, když dojde k chybovému stavu:
  - MQRO_DEAD_LETTER_Q (tato místa umístí původní zprávu do fronty nedoručených zpráv).
  - MQRO_DISCARD_MSG (toto zahodí původní zprávu).

  Pokud nemůže být akce určená odesílající aplikací úspěšně dokončena, bude původní zpráva ponechána na přenosové frontě a nebude vygenerována žádná zpráva o výjimce.
- V případě přechodných zpráv, které cestují prostřednictvím rychlých kanálů zpráv, je původní zpráva odebrána z přenosové fronty a vygenerovaná zpráva o výjimce *i v případě* , že zadaná akce pro chybový stav nemůže být úspěšně dokončena. Je-li například zadán parametr MQRO_DEAD_LETTER_Q, ale původní zprávu nelze umístit do fronty nedoručených zpráv, protože tato fronta je plná, vygeneruje se zpráva o výjimce a bude zahozena původní zpráva.

  Další informace o normálních a rychlých kanálech zpráv naleznete v tématu Rychlost přechodných zpráv (NPMSPEED).

Sestava výjimek se negeneruje, pokud aplikace, která vložila původní zprávu, může být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 .

Aplikace mohou také odesílat zprávy o výjimkách, aby označovaly, že zprávu nelze zpracovat (například proto, že se jedná o debetní transakci, která by způsobila překročení úvěrového limitu účtu).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z příkazů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WIT_DATA
To je stejné jako MQRO_EXCEPTION, s výjimkou toho, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jeden z příkazů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_EXCEPTION_WITH_FULL_DATA
Sestavy výjimek s úplnými požadovanými daty.

To je stejné jako MQRO_EXCEPTION, až na to, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta ve zprávě sestavy.

Neuvádějte více než jeden z příkazů MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA a MQRO_EXCEPTION_WITH_FULL_DATA.

Volby ukončení platnosti: Určete jednu z vypsaných voleb pro vyžádání zprávy o vypršení platnosti sestavy.

MQRO_EXPIRATION
Tento typ sestavy je generován správcem front, pokud je zpráva vyřazena před doručením do aplikace, protože uplynul její čas ukončení platnosti (viz pole *Expiry*). Není-li tato volba nastavena, nebude vygenerována žádná zpráva sestavy, pokud je z tohoto důvodu odstraněna zpráva (i když uvedete jednu z voleb MQRO_EXCEPTION_ *).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z příkazů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_DATA
To je stejné jako MQRO_EXPIRATION, s výjimkou toho, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jeden z příkazů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

MQRO_EXPIRATION_WITH_FULL_DATA
To je stejné jako MQRO_EXPIRATION s tím rozdílem, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z příkazů MQRO_EXPIRATION, MQRO_EXPIRATION_WITH_DATA a MQRO_EXPIRATION_WITH_FULL_DATA.

Volby potvrzení při příjmu: Určete jednu z uvedených voleb pro vyžádání zprávy o potvrzení při příjmu.

MQRO_COA

Tento typ sestavy je generován správcem front, který je vlastníkem fronty místa určení, je-li 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 frontou, může být zpráva COA vygenerovaná správcem front načtena pouze tehdy, je-li potvrzena transakce.

Sestava COA se negeneruje, pokud je pole *Format* v deskriptoru zprávy MQFMT_XMIT_Q_HEADER nebo MQFMT_DEAD_LETTER_HEADER. Zabráníte tak vygenerování sestavy COA, pokud je zpráva vložena do přenosové fronty nebo je nedoručitelná a vložena do fronty nedoručených zpráv.

V případě fronty mostu IMS je vygenerována sestava COA, když se zpráva dostane do fronty produktu IMS (potvrzení přijaté od produktu IMS) a nikoli, je-li zpráva vložena do fronty mostu MQ . To znamená, že pokud IMS není aktivní, žádná sestava COA se negeneruje, dokud se nespustí IMS a zpráva se zařadí do fronty IMS .

Uživatel, který spouští program, který vkládá zprávu do MQMD.Report= MQRO_COA musí mít na frontě odpovědí oprávnění + passid. Pokud uživatel nemá oprávnění + passide, zpráva COA se nedostae do fronty odpovědí. Došlo k pokusu o vložení zprávy do fronty nedoručených zpráv.

Neuvádějte více než jeden z příkazů MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_DATA
To je stejné jako MQRO_COA, až na to, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto ve zprávě sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jeden z příkazů MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

MQRO_COA_WITH_FULL_DATA
To je stejné jako MQRO_COA, až na to, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z příkazů MQRO_COA, MQRO_COA_WITH_DATA a MQRO_COA_WITH_FULL_DATA.

Volby potvrzení při doručení: Určete jednu z uvedených voleb pro vyžádání zprávy sestavy potvrzení o 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.

Je-li zpráva načtena jako součást pracovní jednotky, vygeneruje se zpráva sestavy v rámci stejné pracovní jednotky, takže sestava nebude k dispozici, dokud nebude potvrzena jednotka práce. Je-li jednotka práce zálohována, sestava se neodešle.

Sestava COD není vždy generována v případě, že je načtena zpráva s volbou MQGMO_MARK_PKIP_BACOUT. Je-li primární jednotka práce zálohována, ale sekundární jednotka práce je potvrzena, zpráva se odstraní z fronty, ale hlášení COD se nevygeneruje.

Sestava COD se negeneruje, pokud pole *Format* v deskriptoru zprávy je MQFMT_DEAD_LETTER_HEADER. Zabráníte tak vygenerování sestavy COD, pokud je zpráva nedoručitelná a vložena do fronty nedoručených zpráv.

Funkce MQRO_COD není platná, je-li cílová fronta frontou XCF.

Nezadávejte více než jednu z hodnot MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

MQRO_CED_WITH_DATA
To je stejné jako MQRO_COD, kromě toho, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Je-li MQGMO_ACCEPT_TRUNCATED_MSG zadán v rámci volání MQGET pro původní zprávu a načtená zpráva je oříznuta, závisí množství dat zprávy aplikace umístěné ve zprávě sestavy na daném 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.

Funkce MQRO_COD_WITH_DATA není platná, je-li cílová fronta frontou XCF.

Nezadávejte více než jednu z hodnot MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

MQRO_COD_WITH_FULL_DATA
To je stejné jako MQRO_COD, až na to, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

MQRO_COD_WITH__FULL_DATA není platný, je-li cílová fronta frontou XCF.

Nezadávejte více než jednu z hodnot MQRO_COD, MQRO_COD_WITH_DATA a MQRO_COD_WITH_FULL_DATA.

Volby oznámení akce: Určete jednu nebo obě volby uvedené pro požadavek, aby přijímající aplikace odeslala zprávu s kladnou akcí nebo s negativním výsledkem.

MQRO_PAN
Tento typ sestavy je generován aplikací, která danou zprávu načte a jedná s ním. Zpráva označuje, že akce požadovaná ve zprávě byla úspěšně provedena. Aplikace, která generuje sestavu, určuje, zda má být nějaká data zahrnuta do sestavy.

Kromě odeslání tohoto požadavku do aplikace při načítání zprávy nepodnikává správce front žádnou akci založenou na této volbě. Načtení aplikace musí v případě potřeby vygenerovat sestavu.

MQRO_NAN
Tento typ sestavy je generován aplikací, která danou zprávu načte a jedná s ním. Znamená to, že akce požadovaná ve zprávě nebyla úspěšně provedena. Aplikace, která generuje sestavu, určuje, zda má být nějaká data zahrnuta do sestavy. Můžete například chtít zahrnout některá data označující, proč nebylo možné požadavek provést.

Kromě odeslání tohoto požadavku do aplikace při načítání zprávy nepodnikává správce front žádnou akci založenou na této volbě. Načtení aplikace musí v případě potřeby vygenerovat sestavu.

Aplikace musí určit, které podmínky odpovídají pozitivní akci a které odpovídají negativní akci. Avšak, pokud byl požadavek proveden pouze částečně, vygenerujte sestavu NAN spíše než sestavu PAN, je-li požadována. Každá možná podmínka musí odpovídat buď kladné akci, nebo záporné akci, ale ne oběma.

Volby identifikátoru zprávy: Určete jednu z uvedených voleb pro řízení způsobu nastavení *MsgId* zprávy sestavy (nebo odpovědi na zprávu odpovědi).

MQRO_NEW_MSG_ID
Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď generována jako výsledek této zprávy, vygeneruje se nová *MsgId* pro zprávu nebo zprávu odpovědi.

MQRO_PASS_MSG_ID
Je-li zpráva nebo odpověď vygenerována jako výsledek této zprávy, je zpráva *MsgId* této zprávy zkopírována do *MsgId* sestavy nebo zprávy odpovědi.

MsgId publikační zprávy bude pro každého odběratele, který obdrží kopii publikace, jinak, a proto se *MsgId* zkopírovaný do sestavy nebo zprávy odpovědi bude pro každou z nich lišit.

Není-li tato volba zadána, předpokládá se hodnota MQRO_NEW_MSG_ID.

Volby identifikátoru korelace: Určete jednu z uvedených voleb pro řízení, jak má být nastavena hodnota *CorrelId* zprávy sestavy (nebo zprávy odpovědi).

MQRO_COPY_MSG_ID_TO_CORREL_ID
Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď vygenerována jako výsledek této zprávy, je zpráva *MsgId* této zprávy zkopírována do *CorrelId* sestavy nebo zprávy odpovědi.

Pro každého odběratele, který obdrží kopii publikace, se bude pro každého odběratele lišit *MsgId* , a proto se *MsgId* kopie souboru sestavy nebo zprávy odpovědi do sestavy *CorrelId* bude lišit pro každou z nich.

ID_KOLEKCE_MQRO_PASS_RELACE_
Je-li zpráva nebo odpověď vygenerována jako výsledek této zprávy, je zpráva *CorrelId* této zprávy zkopírována do *CorrelId* sestavy nebo zprávy odpovědi.

CorrelId publikační zprávy bude specifické pro odběratele, pokud nepoužije volbu MQSO_SET_CORREL_ID a nastaví pole ID SubCorrelv MQSD na MQCI_NONE. Proto je možné, že se

CorrelId zkopírovaný do sestavy *CorrelId* sestavy nebo zprávy odpovědi bude pro každou z nich lišit.

Není-li tato volba zadána, předpokládá se hodnota MQRO_COPY_MSG_ID_TO_CORREL_ID.

Servery odpovídání na požadavky nebo generování zpráv sestav musí zkontrolovat, zda byly volby MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID nastaveny v původní zprávě. Pokud byly, servery musí provést akci popsanou pro tyto volby. Není-li nastaven ani jeden z nich, servery musí přijmout odpovídající výchozí akci.

Volby odebrání: Určete jednu z voleb vypsaných k řízení dispozice původní zprávy, pokud ji nelze doručit do cílové fronty. Aplikace může nastavit volby odebrání nezávisle na požadování sestav výjimek.

MQRO_DEAD_LETTER_Q
Jedná se o výchozí akci a umístí zprávu do fronty nedoručených zpráv, pokud tuto zprávu nelze doručit do cílové fronty. K tomu dojde v následujících situacích:

- Když aplikace, která zadala původní zprávu, nemůže být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Vygeneruje se zpráva hlášení o výjimce, pokud ji někdo požadoval odesilatel.
- Když byla aplikace, která vložila původní zprávu, do tématu vložena

MQRO_DISCARD_MSG
Vyřadí zprávu, pokud ji nelze doručit do cílové fronty. K tomu dojde v následujících situacích:

- Když aplikace, která zadala původní zprávu, nemůže být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Vygeneruje se zpráva hlášení o výjimce, pokud ji někdo požadoval odesilatel.
- Když byla aplikace, která vložila původní zprávu, do tématu vložena

Pokud chcete vrátit původní zprávu odesílateli, aniž by byla původní zpráva umístěna do fronty nedoručených zpráv, musí odesilatel určit MQRO_DISCARD_MSG s MQRO_EXCEPTION_WITH_FULL_DATA.

MQRO_PASS_DISCARD_AND_EXPIRY
Je-li tato volba nastavena na zprávu a je generována zpráva nebo odpověď kvůli ní, deskriptor zprávy této sestavy zdědí:

- MQRO_DISCARD_MSG, pokud byl nastaven.
- Zbývající doba vypršení platnosti zprávy (pokud se nejedná o sestavu o vypršení platnosti). Je-li toto hlášení o vypršení platnosti, je doba vypršení platnosti nastavena na 60 sekund.

Volba aktivity

AKTIVITA MQRO_ACTIVITY
Použití této hodnoty umožňuje trasování cesty **jakékoliv** zprávy v rámci sítě správce front. Volba sestavy může být uvedena na libovolné aktuální zprávě uživatele a okamžitě vám umožňuje začít vypočítávat trasu zprávy přes síť.

Pokud aplikace, která generuje zprávu, nemůže povolit generování sestavy o aktivitě, může být povoleno vytváření sestav pomocí uživatelské procedury pro přechod rozhraní API dodané administrátory správců front.

Poznámka:

1. Čím nižší je počet správců front v síti, kteří mohou generovat sestavy o aktivitách, tím méně je trasa k dané trase.
2. Sestavy aktivit mohou být obtížné umístit ve správném pořadí, aby bylo možné určit trasu, která byla přijata.
3. Sestavy aktivit nemusí být schopny najít trasu k požadovanému místu určení.
4. Zprávy s touto sadou voleb sestavy musí být přijaty kterýchkoli správcem front, a to i v případě, že nerozumí této volbě. To umožňuje nastavit volbu sestavy na libovolné uživatelské zprávě, i když jsou zpracovány správcem front, který neníIBM WebSphere MQ 6.0 nebo pozdější.

5. Pokud proces, buď správce front nebo uživatelský proces, provede aktivitu na zprávě s touto sadou voleb, může se rozhodnout vygenerovat a vložit sestavu aktivity.

Výchozí volba: Zadejte následující, pokud nejsou požadovány žádné volby sestavy:

MQRO_NONE
Použijte tuto hodnotu, chcete-li označit, že nebyly zadány žádné další volby. Funkce MQRO_NONE je definována pro dokumentaci programu podpory. Není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze detekovat.

Obecné informace:

1. Všechny požadované typy sestav musí být výslovně vyžádány aplikací, která odesílá původní zprávu. Je-li například požadována zpráva COA, ale sestava výjimek není, vygeneruje se zpráva COA, když je zpráva umístěna do cílové fronty, ale pokud je fronta cíle zaplněna, jakmile zpráva dorazí, nebude vygenerována žádná sestava výjimek. Nejsou-li nastaveny žádné volby obslužného programu *Report* , správce front nebo agent kanálu zpráv (MCA) negeneruje žádné zprávy sestavy.

   Některé volby sestavy lze zadat i v případě, že lokální správce front je nerozpoznal; to je užitečné, pokud má být volba zpracována správcem front *destination* . Další informace viz část "Volby sestav a příznaky zpráv" na stránce 893.

   Je-li požadována zpráva sestavy, musí být název fronty, do které má být sestava odeslána, uvedena v poli *ReplyToQ* . Když je přijata zpráva sestavy, charakter sestavy lze určit prozkoumáním pole *Feedback* v deskriptoru zprávy.
2. Pokud správce front nebo MCA, který generuje zprávu sestavy, nemůže vložit zprávu sestavy do fronty odpovědí (například, protože fronta odpovědí nebo přenosová fronta je plná), zpráva sestavy bude umístěna místo fronty nedoručených zpráv. Pokud se *také* nezdaří, nebo pokud neexistuje žádná fronta nedoručených zpráv, závisí akce na typu zprávy hlášení:
   - Je-li zpráva hlášení výjimkou, zpráva, která generovala zprávu o výjimce, je ponechána ve své přenosové frontě, což zajišťuje, že zpráva nebude ztracena.
   - Pro všechny ostatní typy sestav je zpráva sestavy vyřazena a zpracování bude normálně pokračovat. Důvodem je to, že původní zpráva již byla doručena bezpečně (zprávy sestav COA nebo COD) nebo již není o žádný zájem (pro zprávu o vypršení platnosti zprávy).

   Jakmile byla zpráva sestavy úspěšně umístěna do fronty (cílová fronta nebo mezilehlá přenosová fronta), zpráva již není předmětem speciálního zpracování; zachází se stejně jako s jakoukoli jinou zprávou.
3. Když je sestava generována, je otevřena fronta *ReplyToQ* a zpráva sestavy nabyla pomocí oprávnění *UserIdentifier* v MQMD zprávy způsobující tuto sestavu, s výjimkou následujících případů:
   - Zprávy výjimek generované přijímajícím agentem MCA jsou při pokusu o vložení zprávy způsobující vložení zprávy použity bez ohledu na to, jakou má agent MCA práci.
   - Sestavy COA generované správcem front byly použity bez ohledu na to, zda byla zpráva při generování sestavy vložena do správce front, který byl použit. Například, pokud byla zpráva vložena přijímajícím agentem MCA pomocí identifikátoru uživatele MCA, umístí správce front zprávu COA pomocí identifikátoru uživatele MCA.

   Aplikace generující sestavy musí používat stejné oprávnění, které používají při generování odpovědi; obvykle se jedná o oprávnění identifikátoru uživatele v původní zprávě.

   Má-li sestava cestovat do vzdáleného cíle, odesilatelé a příjemci se mohou rozhodnout, zda ji přijmou, stejně jako pro jiné zprávy.
4. Je-li požadována zpráva hlášení s daty, postupujte takto:
   - Zpráva sestavy se vždy vygeneruje s množstvím dat požadovaných odesílatelem původní zprávy. Je-li zpráva zprávy příliš velká pro frontu odpovědí, dojde k výše popsanému zpracování; zpráva sestavy se nikdy neosekne tak, aby se vešla do fronty odpovědí.
   - Je-li *Format* původní zprávy MQFMT_XMIT_Q_HEADER, data obsažená v sestavě nezahrnují MQXQH. Data sestavy začínají prvním bajtem dat nad rámec MQXQH v původní zprávě. Dochází k tomu, zda je fronta přenosovou frontou či nikoli.

5. Je-li ve frontě odpovědí přijata zpráva COA, COD nebo Zpráva o vypršení platnosti, je zaručeno, že byla doručena původní zpráva, byla doručena nebo vypršela její platnost, podle situace. Je-li však jedna nebo více z těchto zpráv sestavy vyžádáno a není přijata, nelze předpokládat, že by se mohlo jednat o jednu z následujících možností:
   a. Zpráva sestavy je zadržena, protože odkaz je mimo provoz.
   b. Zpráva sestavy je zadržena, protože blokující podmínka existuje ve střední přenosové frontě nebo ve frontě odpovědí (například plná nebo zablokovaná fronta pro vložení)).
   c. Zpráva sestavy se nachází ve frontě nedoručených zpráv.
   d. Při pokusu správce front o vygenerování zprávy sestavy ji nebylo možné vložit do příslušné fronty ani do fronty nedoručených zpráv, takže zprávu sestavy nebylo možné vygenerovat.
   e. Došlo k selhání správce front mezi hlášenou akcí (přijetí, doručení nebo vypršení platnosti) a generováním odpovídající zprávy sestavy. (To se nestane pro zprávy COD, pokud aplikace načte původní zprávu v rámci pracovní jednotky, protože zpráva hlášení COD je generována v rámci stejné pracovní jednotky.)

   Výjimečná zpráva hlášení může být zadržena stejným způsobem z důvodů 1, 2 a 3 výše. Pokud však program MCA nemůže generovat zprávu s hlášením o výjimce (zprávu sestavy nelze vložit do fronty odpovědí nebo do fronty nedoručených zpráv), zůstane původní zpráva v přenosové frontě na odesílateli a kanál je uzavřen. K tomu dojde bez ohledu na to, zda byla zpráva sestavy generována při odesílání nebo na přijímajícím konci kanálu.
6. Je-li původní zpráva dočasně zablokována (výsledkem je generování zprávy o výjimce a původní zpráva byla vložena do fronty nedoručených zpráv), ale blokace je vymazána a aplikace pak přečte původní zprávu z fronty nedoručených zpráv a znovu ji umístí do místa určení, může dojít k následujícím:
   - I když byla vygenerována zpráva o výjimce, bude původní zpráva nakonec úspěšně doručena do místa určení.
   - Pro jednu původní zprávu je vygenerována více než jedna zpráva zprávy o výjimce, 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 na téma, které může být nula, jedno nebo mnoho. To je třeba vzít v úvahu při výběru možnosti použití voleb sestavy, protože mnoho zpráv sestav může být generováno jako výsledek.
2. Při vkládání zprávy do tématu může být k dispozici mnoho cílových front, které mají být předány kopie zprávy. Mají-li některé z těchto cílových front problém, jako je například zaplnění fronty, závisí úspěšné dokončení příkazu MQPUT na nastavení NPMSGDLV nebo PMSGDLV (v závislosti na trvání zprávy). Pokud je nastavení takové, že doručení zprávy do cílové fronty musí být úspěšné (například, že se jedná o trvalou zprávu na trvalém odběrateli a PMSGDLV je nastaveno na ALL nebo ALLDUR), pak je úspěch definován jako jedno z následujících kritérií:
   - Úspěšné vložení do fronty odběratele
   - Použití MQRO_DEAD_LETTER_Q a úspěšné vložení do fronty nedoručených zpráv, pokud fronta odběratele nemůže převzít zprávu.
   - Použijte MQRO_DISCARD_MSG, pokud fronta odběratele nemůže převzít zprávu.

Hlásit zprávy pro segmenty zpráv:

1. Zprávy sestavy mohou být požadovány pro zprávy, které mají povolenou segmentaci (viz popis příznaku MQMF_SEGMENTATION_ALLOWED). Pokud správce front zjistí, že je nutné zprávu segmentovat, může být vygenerována zpráva sestavy pro každý z segmentů, který následně zjistí příslušnou podmínku. Aplikace musí být připraveny pro příjem více zpráv sestav pro každý typ požadované zprávy. Pole *GroupId* ve zprávě se sestavou použijte ke korelaci více sestav s identifikátorem skupiny původní zprávy a pole *Feedback* identifikuje typ každé zprávy sestavy.
2. Je-li hodnota MQGMO_LOGICAL_ORDER použita k načtení zpráv sestav pro segmenty, uvědomte si, že sestavy *různých typů* mohou být vráceny po sobě jdoucími voláními MQGET. Je-li například požadována zpráva COA i CHSK pro zprávu segmentovanou správcem front, mohou zprávy COA

a COD vracet zprávy sestav COA a COD prokládané nepředvídatelným způsobem. Vyvarovat se pomocí volby MQGMO_COMPLETE_MSG (volitelně s MQGMO_ACCEPT_TRUNCATED_MSG). MQGMO_COMPLETE_MSG způsobí, že správce front znovu sestaví zprávy sestavy se stejným typem sestavy. Například první volání MQGET může znovu sestavit všechny zprávy COA vztahující se k původní zprávě a druhý volání MQGET může znovu sestavit všechny zprávy COD. Která je znovu sestavená jako první závisí na tom, který typ zprávy hlášení se vyskytne první ve frontě.

3. Aplikace, které samy umístí segmenty, mohou uvádět různé volby sestavy pro každý segment. Povšimněte si však následujících bodů:
   - Pokud jsou segmenty načteny pomocí volby MQGMO_COMPLETE_MSG, budou správcem front uznány pouze volby sestavy v *prvním* segmentu.
   - Pokud jsou segmenty načteny jeden po druhém a většina z nich má jednu z voleb MQRO_ *, ale alespoň jeden segment ne, nemůžete použít volbu MQGMO_COMPLETE_MSG k načtení zpráv sestavy s jediným voláním MQGET, nebo použít volbu MQGMO_ALL_SEGMENTS_AVAILABLE pro zjištění, zda byly obdrženy všechny zprávy sestavy.
4. V síti produktu MQ mohou správci front mít různé schopnosti. Je-li zpráva sestavy pro segment generována správcem front nebo agentem MCA, který nepodporuje segmentaci, správce front nebo MCA standardně neobsahuje nezbytné informace o segmentech ve zprávě sestavy a může to ztížit identifikaci původní zprávy, která způsobila vygenerování sestavy. Zamezte těmto potížím tím, že požadujete data se zprávou sestavy, tj. určením příslušných voleb MQRO_ * _WITH_DATA nebo MQRO_ * _WITH_FULL_DATA. Uvědomte si však, že je-li zadána hodnota MQRO_ * _WITH_DATA, může být do aplikace, která načte zprávu sestavy, vrácena hodnota *menší než* 100 bajtů dat aplikace, pokud je zpráva sestavy generována správcem front nebo agentem MCA, který nepodporuje segmentaci.

Obsah deskriptoru zpráv pro zprávu sestavy: Pokud správce front nebo agent kanálu zpráv (MCA) vygeneruje zprávu s hlášením, nastaví pole v deskriptoru zpráv na následující hodnoty a poté vloží zprávu normálním způsobem.

Tabulka 502. Hodnoty použité pro pole MQMD, je-li generována zpráva sestavy

Pole v MQMD	Použitá hodnota
StrucId	ID_STRUKTURY MQM_STRUCT
Version	MQMD_VERSION_2
Report	MQRO_NONE
MsgType	SESTAVA MQMT_REPORT
Expiry	MQEI_UNLIMITED
Feedback	Podle potřeby pro charakter sestavy (MQFB_COA, MQFB_COD, MQFB_EXPIRATION nebo 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 uvedeno ve volbách sestavy v původním deskriptoru zpráv
CorrelId	Jak je uvedeno ve volbách sestavy v původním deskriptoru zpráv
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Název správce front
UserIdentifier	Podle nastavení volby MQPMO_PASS_IDENTITY_CONTEXT

Tabulka 502. Hodnoty použité pro pole MQMD, je-li generována zpráva sestavy (pokračování)

Pole v MQMD	Použitá hodnota
AccountingToken	Podle nastavení volby MQPMO_PASS_IDENTITY_CONTEXT
ApplIdentityData	Podle nastavení volby MQPMO_PASS_IDENTITY_CONTEXT
PutApplType	MQAT_QMGR nebo případně pro agenta MCA (Message Channel Agent)
PutApplName	Prvních 28 bajtů názvu správce front nebo názvu agenta kanálu zpráv. Pro zprávy sestav generované mostem IMS toto pole obsahuje název skupiny XCF a název člena XCF systému IMS , kterého se zpráva týká.
PutDate	Datum, kdy se odešle zpráva hlášení
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 zpráv, pokud není MQOL_UNDEFINED a jinak nastaveno na délku původních dat zprávy

Aplikace generující sestavu je doporučována pro nastavení podobných hodnot, s výjimkou následujících:

- Pole *ReplyToQMgr* může být nastaveno na prázdné místo (správce front to změní na název lokálního správce front, když je zpráva vložena).
- Nastavte pole kontextu pomocí volby, která má být použita pro odpověď, obvykle MQPMO_PASS_IDENTITY_CONTEXT.

Analýza pole sestavy: Pole *Report* obsahuje podpole; z tohoto důvodu aplikace, které potřebují zkontrolovat, zda odesilatel zprávy vyžádal určitou sestavu, musí používat jednu z technik popsaných v “Analýza pole sestavy” na stránce 895.

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)

Označuje typ zprávy. Typy zpráv jsou seskupeny následujícím způsobem:

MQM_SYSTEM_FIRST
Nejnižší hodnota pro systémem definované typy zpráv.

MQM_SYSTEM_LAST
Nejvyšší hodnota pro typy zpráv definované systémem.

V rozsahu systému jsou momentálně definovány následující hodnoty:

MQM_DATAGRAM
Zpráva je taková, která nevyžaduje odpověď.

POŽADAVEK MQMT_REQUEST
Zpráva je taková, která vyžaduje odpověď.

Do pole *ReplyToQ* zadejte název fronty, do níž má být odeslána odpověď. Pole *Report* udává, jak nastavit *MsgId* a *CorrelId* odpovědi.

MQMT_REPLY
Zpráva je odpovědí na předchozí zprávu požadavku (MQMT_REQUEST). Zpráva musí být odeslána do fronty uvedené v poli *ReplyToQ* zprávy požadavku. Pole *Report* v požadavku řídí, jak nastavit *MsgId* a *CorrelId* na odpověď.

Poznámka: Správce front nevynucuje vztah požadavek-odezva. Jedná se o zodpovědnost aplikace.

SESTAVA MQMT_REPORT
Zpráva se hlásí k očekávanému nebo neočekávanému výskytu, obvykle související s nějakou jinou zprávou (například byla přijata zpráva požadavku, která obsahovala neplatná data). Odešlete zprávu do fronty označené v poli *ReplyToQ* deskriptoru zprávy původní zprávy. Nastavte pole *Feedback* tak, aby určovalo povahu sestavy. Použijte pole *Report* původní zprávy, abyste mohli řídit, jak nastavit *MsgId* a *CorrelId* zprávy sestavy.

Zprávy sestav generované správcem front nebo agentem oznamovacího kanálu jsou vždy odesílány do fronty *ReplyToQ* s použitím polí *Feedback* a *CorrelId* , jak je popsáno výše.

Lze také použít hodnoty definované aplikací. Musí být v následujícím rozsahu:

MQM_APPL_FIRST
Nejnižší hodnota pro typy zpráv definované aplikací.

MQM_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 tomu tak 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)

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 po dobu vypršení platnosti, je třeba nastavit **MQMD**.**Expiry** na 600.

Hodnota se sníží tak, aby odrážela dobu, kterou zpráva stráví na cílové frontě, a také na všech intermediačních přenosových frontách, pokud je vložení do vzdálené fronty. Lze ji také snížit pomocí agentů kanálů zpráv tak, aby odrážely časy přenosu, jsou-li tyto údaje významné. Podobně může i aplikace přeposílání této zprávy do jiné fronty snížit hodnotu, je-li to nutné, pokud si ji zprávu uchovala po významnou dobu. Avšak čas vypršení platnosti je považován za přibližný a hodnota nemusí být snížena, aby odrážela malé časové intervaly.

Když je zpráva načtena aplikací pomocí volání MQGET, pole *Expiry* představuje dobu vypršení platnosti, která stále zůstává.

Po uplynutí doby vypršení platnosti zprávy bude možné, že správce front bude vyřazen z ukončení. Zpráva je zahozena v případě, že dojde k volání příkazu MQGET při procházení nebo při procházení, které by vrátilo zprávu, protože již platnost zprávy dosud nevypršela. Například volání MQGET bez procházení s polem *MatchOptions* v produktu MQGMO nastaveným na čtení MQMO_NONE z fronty s řazením FIFO zahodí všechny zprávy s vypršelou platností do první zprávy bez vypršení platnosti. Při použití fronty s prioritou bude stejné volání vyřazeno vypršelé zprávy s vyšší prioritou a zprávami stejné priority, které dorazily do fronty před první zprávou bez vypršení platnosti.

Platnost zprávy, jejíž platnost vypršela, se nikdy nevrací do aplikace (buď při procházení nebo při volání MQGET bez procházení), takže hodnota v poli *Expiry* deskriptoru zpráv po úspěšném volání MQGET je buď větší než nula, nebo speciální hodnota MQEI_UNLIMITED.

Je-li zpráva vložena do vzdálené fronty, zpráva může vypršet (a být vyřazena), zatímco se nachází ve střední přenosové frontě, než se zpráva dostane do cílové fronty.

Sestava je generována, pokud je zahozena zpráva s vypršenou platností, pokud byla zpráva uvedena jako jedna z voleb sestavy MQRO_EXPIRATION_ *. Není-li zadána žádná z těchto voleb, nebude vygenerována žádná taková sestava. Předpokládá se, že zpráva již není relevantní po uplynutí této doby (možná proto, že ji později nahradila novější zpráva).

U zprávy v rámci synchronizačního bodu začíná interval vypršení platnosti v době, kdy je zpráva vložena, nikoli doba, po kterou je synchronizační bod potvrzen. Je možné, že interval vypršení platnosti může projít před potvrzením synchronizačního bodu. V tomto případě bude zpráva po operaci potvrzení vyřazena a zpráva se nevrátí do aplikace jako odezva na operaci MQGET.

Jakýkoliv jiný program, který vyřadí zprávy na základě doby platnosti, musí také odeslat odpovídající zprávu, pokud byla požadována.

Notes:

1. Je-li zpráva vložena s hodnotou *Expiry* nula nebo s číslem větším než 999 999 999, volání MQPUT nebo MQPUT1 selže s kódem příčiny MQRC_EXPIRY_ERROR; v tomto případě se nevygeneruje žádná zpráva.

   Chcete-li povolit kód příčiny 2013, MQRC_EXPIRY_ERROR, je třeba povolit proměnnou prostředí AMQ_ENFORCE_MAX_EXPIRY_ERROR.

   Příklad použití příkazu Linuxje následující:

   ```
   $ export AMQ_ENFORCE_MAX_EXPIRY_ERROR=True
   ```

   Všimněte si, že:

   - Důležitá věc je exportovat proměnnou
   - Skutečná hodnota je však ignorována, při kontrole nastavení by však mohla být užitečná použití produktu `True` .
2. Vzhledem k tomu, že zpráva s uplynulou dobou platnosti může být zahozena až později, mohou existovat zprávy ve frontě, které prošly jejich vypršením platnosti, a které proto nejsou způsobilé pro načtení. Tyto zprávy se však započítávají do počtu zpráv ve frontě pro všechny účely, včetně spuštění hloubky.

   Pokud se odběratel nebo odběratel (klient) pokusí získat zprávu a že vypršela platnost této zprávy, klient neobdrží nic, protože zpráva byla zahozena, protože byla příliš stará. Klient kromě toho neobdrží žádnou chybovou zprávu.
3. Je-li zpráva požadována pro vyřazení, vygeneruje se zpráva o vypršení platnosti, je-li tato zpráva vyřazena z konce.
4. Vyřazení zprávy s vypršenou platností a generování sestavy vypršení platnosti, je-li požadováno, nejsou nikdy součástí pracovní jednotky aplikace, i když byla zpráva naplánována k vyřazení v důsledku volání MQGET v rámci pracovní jednotky.
5. Je-li zpráva s téměř skončenou platností načtena voláním MQGET v rámci pracovní jednotky a jednotka práce je následně vrácena, může se stát, že zpráva bude způsobilá k vyřazení, než ji bude možné znovu načíst.
6. Je-li zpráva s téměř ukončenou platností zamknuta voláním MQGET s MQGMO_LOCK, může být zpráva považována za vhodnou k vyřazení, než ji bude možné načíst voláním MQGET s MQGMO_MSG_UNDER_CURSOR.Kód příčiny MQRC_NO_MSG_UNDER_CURSOR je vrácen při této následné operaci MQGET, pokud k tomu dojde.
7. Když je načtena zpráva požadavku s dobou vypršení platnosti větší než nula, může aplikace provést jednu z následujících akcí, když odešle zprávu odpovědi:

   - Zkopírujte zbývající dobu vypršení platnosti ze zprávy požadavku do zprávy odpovědi.
   - Nastavte čas vypršení platnosti ve zprávě odpovědi na explicitní hodnotu větší než nula.
   - Nastavte dobu vypršení platnosti ve zprávě odpovědi na MQEI_UNLIMITED.

Akce, která se má provést, závisí na návrhu aplikace. Avšak výchozí akce pro vložení zpráv do fronty nedoručených zpráv (undelivered-message) musí být zachováním zbývajícího času vypršení platnosti zprávy a k dalšímu snížení její hodnoty.

8. Zprávy spouštěče jsou vždy generovány spolu s MQEI_UNLIMITED.
9. Zpráva (obvykle v přenosové frontě), která má název produktu *Format* MQFMT_XMIT_Q_HEADER, má druhý deskriptor zprávy v rámci MQXQH. Má proto k sobě přidružená dvě pole *Expiry* . V tomto případě by měly být zaznamenány následující dodatečné body:
   - Když aplikace vloží zprávu do vzdálené fronty, umístí správce front zprávu na počátku do lokální přenosové fronty a předpony dat aplikační zprávy se strukturou MQXQH. Správce front nastaví hodnoty dvou polí *Expiry* tak, aby byly shodné s hodnotami zadanými v aplikaci.

     Pokud aplikace vloží zprávu přímo do lokální přenosové fronty, musí data zprávy již začínat strukturou MQXQH a název formátu musí být MQFMT_XMIT_Q_HEADER. V takovém případě aplikace nemusí nastavit hodnoty těchto dvou polí *Expiry* tak, aby byla stejná. (Správce front zkontroluje, že pole *Expiry* v rámci MQXQH obsahuje platnou hodnotu a že data zprávy jsou dostatečně dlouhá na to, aby mohla být zahrnuta). Pro aplikaci, která může zapisovat přímo do přenosové fronty, musí aplikace vytvořit záhlaví přenosové fronty s vloženým deskriptorem zprávy. Je-li však hodnota vypršení platnosti v deskriptoru zpráv zapsána do přenosové fronty nekonzistentní s hodnotou v deskriptoru vložené zprávy, dojde k odmítnutí vypršení platnosti.
   - Je-li zpráva s názvem *Format* MQFMT_XMIT_Q_HEADER načtena z fronty (zda se jedná o normální nebo přenosovou frontu), správce front sníží *obě* tato pole *Expiry* s časem stráveným čekáním na frontu. Pokud data zprávy nejsou dostatečně dlouhá, aby zahrnula pole *Expiry* do pole MQXQH, žádná chyba se neobjevuje.
   - Správce front používá pole *Expiry* v odděleném deskriptoru zprávy (to znamená, že ne test v deskriptoru zprávy vloženého do struktury MQXQH), aby otestuje, zda je zpráva vhodná pro vyřazení.
   - Pokud se počáteční hodnoty těchto dvou polí *Expiry* liší, doba *Expiry* v odděleném deskriptoru zpráv, když je zpráva načtena, může být větší než nula (takže zpráva není způsobilá pro zrušení), 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 hodnota MQIIH_PASS_EXPIRATION není nastavena v poli Příznaky objektu MQIIH. Další informace viz Příznaky .

Je rozpoznána následující speciální hodnota:

MQEI_UNLIMITED
Zpráva má neomezenou dobu 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 v z/OS
Na serveru IBM MQ for z/OSjsou zprávy, jejichž platnost vypršela, zahozeny při příštím použití volání MQGET.

Pokud však takové volání neexistuje, zpráva s vypršenou platností se nevyřadí a u některých front se může hromadit velké množství zpráv s vypršenou platností. Chcete-li tento postup napravit, nastavte správce front tak, aby skenoval fronty pravidelně, a zařaďte zprávy s ukončenou platností do jedné nebo více front jedním z následujících způsobů:

Pravidelné skenování
Můžete zadat období pomocí atributu správce front EXPRYINT (interval vypršení platnosti). Při každém dosažení intervalu vypršení platnosti správce front hledá kandidátské fronty, které stojí za to, aby zařadily vypršenou zprávu.

Správce front udržuje informace o zprávách s vypršenou platností na každé frontě a ví, zda má skenování vypršelých zpráv za to, že se vyplatí. Takže v každém okamžiku 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 se má restartovat, nebo první, kdo má sadu EXPRYINT. Pokud je tento správce front ukončen, převezme řízení fronty jiný správce front v dané skupině sdílení front. Nastavte hodnotu intervalu vypršení platnosti pro všechny správce front v rámci skupiny sdílení front na stejnou hodnotu.

Nezapomeňte, že zpracování vypršení platnosti probíhá pro každou frontu v případě, že se správce front restartuje, bez ohledu na nastavení EXPRYINT.

Explicitní požadavek
Zadejte příkaz REFRESH QMGR TYPE (EXPIRY) a určete frontu nebo fronty, které chcete skenovat.

Enforcing lower expiration times
Administrátoři mohou omezit dobu vypršení platnosti jakékoli zprávy zařazené do fronty nebo tématu pomocí atributu **CAPEXPRY** uvedeného v atributu **CUSTOM** ve frontě nebo tématu.

Doba vypršení platnosti zadaná v poli **Expiry** deskriptoru MQMD, která je větší než hodnota **CAPEXPRY** zadaná v atributu **CUSTOM** ve frontě nebo tématu, bude nahrazena hodnotou **CAPEXPRY** . Bude použit čas vypršení platnosti určený aplikací, která je nižší než hodnota **CAPEXPRY** .

Všimněte si, že hodnota **CAPEXPRY** je vyjádřena v desetinách sekund, takže jedna minuta má hodnotu 600.

Je-li na cestě vyřešení použit více než jeden objekt, například když je zpráva vložena do aliasu nebo vzdálené fronty, pak se nejnižší ze všech hodnot **CAPEXPRY** použije jako horní limit pro vypršení platnosti zprávy.

Změny hodnot **CAPEXPRY** se projeví okamžitě. Hodnota vypršení platnosti je vyhodnocována pro každé vložení do fronty nebo tématu a je tak citlivá na rozlišení objektu, které se může lišit mezi jednotlivými operacemi vložení.

Všimněte si však, že stávající zprávy ve frontě, před změnou v produktu **CAPEXPRY**, nejsou změnou ovlivněny (tj. jejich doba vypršení platnosti zůstane neporušená). Nová doba vypršení platnosti má pouze nové zprávy, které jsou vloženy do fronty po provedení změny v produktu **CAPEXPRY** .

Například v klastru, kde se provádí vložení do fronty otevřené pomocí MQOO_BIND_NOT_FIXED, lze zprávy přiřadit různým hodnotám vypršení platnosti při každém vložení, v závislosti na hodnotě **CAPEXPRY** nastavené pro přenosovou frontu, kterou používá kanál, který odesílá zprávu do vybraného cílového správce front.

Všimněte si, že vložení do fronty nebo tématu aplikací JMS s určením zpoždění doručení selže s MQRC_EXPIRY_ERROR, pokud je prodleva doručení nad vyřešenou dobu vypršení platnosti pro cílovou frontu nebo téma. Atribut **CAPEXPRY** nastavený ve frontě vyřešený pro cíl JMS může způsobit tuto chybu.

Poznámka: Produkt **CAPEXPRY** nesmí být použit ve všech frontách, které budou obsahovat interně generované zprávy produktu IBM MQ , jako např. SYSTEM.CLUSTER front a SYSTEM.PROTECTION.POLICY.QUEUE.

Související odkazy
Fronty DEFINE
téma DEFINE

Zpětná vazba (MQLONG)
Pole Zpětná vazba se používá se zprávou typu MQMT_REPORT k označení povahy sestavy a je smysluplná pouze s daným typem zprávy.

Pole může obsahovat jednu z hodnot MQFB_ *, nebo jednu z hodnot MQRC_ *. Kódy zpětné vazby jsou seskupeny následujícím způsobem:

MQFB_NONE
Nebyla poskytnuta žádná zpětná vazba.

MQFB_SYSTEM_FIRST
Nejnižší hodnota pro zpětnou vazbu generovanou systémem.

MQFB_SYSTEM_LAST
Nejvyšší hodnota zpětné vazby generované systémem.

Rozsah kódů zpětné vazby generovaných systémem MQFB_SYSTEM_FIRST prostřednictvím struktury MQFB_SYSTEM_LAST zahrnuje obecné kódy zpětné vazby uvedené v tomto tématu (MQFB_ *) a také kódy příčiny (MQRC_ *), které se mohou vyskytnout, když nelze zprávu vložit do cílové fronty.

MQFB_APPL_FIRST
Nejnižší hodnota pro zpětnou vazbu generovaná aplikací.

MQFB_APPL_LAST
Nejvyšší hodnota zpětné vazby generované aplikací.

Aplikace, které generují zprávy sestav, nesmějí používat kódy zpětné vazby v systémovém rozsahu (jiném než MQFB_QUIT), pokud chtějí simulovat zprávy sestavy generované správcem front nebo agentem oznamovacího kanálu.

V rámci volání MQPUT nebo MQPUT1 musí být zadaná hodnota buď MQFB_NONE, nebo musí být v rámci rozsahu systému nebo rozsahu aplikace. Tato hodnota je zkontrolována bez ohledu na hodnotu parametru *MsgType*.

Obecné kódy zpětné vazby:

MQFB_COA
Potvrzení přijetí do cílové fronty (viz MQRO_COA).

MQFB_COD
Potvrzení o doručení do přijímací aplikace (viz MQRO_COD).

MQFB_EXPIRATION
Zpráva byla zahozena, protože nebyla odebrána z cílové fronty před uplynutím jeho doby vypršení platnosti.

MQFB_PAN
Pozitivní upozornění na akci (viz MQRO_PAN).

MQFB_NAN
Negativní upozornění na akci (viz MQRO_NAN).

MQFB_QUIT
Ukončit aplikaci.

To může použít program plánování pracovní zátěže k řízení počtu instancí aplikačního programu, které jsou spuštěny. Odeslání zprávy MQMT_REPORT s tímto kódem zpětné vazby na instanci aplikačního programu indikuje instanci, že by měla zastavit zpracování. Dodržování této konvence je však záležitostí pro aplikaci; správce front jej nevynucuje.

Kódy zpětné vazby kanálu:

MQFB_CHANNEL_COMPLETED
Kanál byl ukončen normálně.

MQFB_CHANNEL_FAIL
Kanál byl ukončen nestandardním způsobem a přešel do stavu ZASTAVENO.

MQFB_CHANNEL_FAIL_RETRY
Kanál byl nestandardně ukončen a přejde do stavu RETRY.

IMS-feedback kódy zpětné vazby

Tyto kódy se používají při přijetí neočekávaného chybného kódu IMS-OTMA. Chybový kód nebo, je-li kód příčiny 0x1A kód příčiny přidružený k tomuto chybovým kódu, je indikován v *Feedback*.

1. Pro kódy *Feedback* v rozsahu MQFB_IMS_FIRST (300) přes MQFB_IMS_LAST (399) byl přijat chybový kód jiný než 0x1A . Výraz *sense code* je dán výrazem (*Feedback* - MQFB_IMS_FIRST+1)
2. Pro kódy *Feedback* v rozsahu MQFB_IMS_NACK_1A_REASON_FIRST (600) až MQFB_IMS_NACK_1A_REASON_LAST (855) byl obdržen chybový kód 0x1A . Výraz *kód příčiny* přidružený k chybnému kódu je dán výrazem (*Zpětná vazba* - MQFB_IMS_NACK_1A_REASON_FIRST)

Význam chybových kódů IMS-OTMA a odpovídajících kódů příčiny jsou popsány v příručce *Open Transaction Manager Access Guide and Reference*.

Pomocí mostu IMS mohou být generovány následující kódy zpětné vazby:

MQFB_DATA_LENGTH_ZERO
Délka segmentu byla nula v datech aplikace zprávy.

MQFB_DATA_LENGTH_NEGATIVE
Délka segmentu byla záporná v datech aplikace zprávy.

MQFB_DATA_LENGTH_TOO_BIG
Délka segmentu byla příliš velká v datech aplikace zprávy.

PŘETEČENÍ MQFFB_BUFFER_OVERFLOW
Hodnota jednoho z polí s délkou by způsobila přetečení vyrovnávací paměti zpráv.

MQFB_LENGTH_OFF_BY_ONE
Hodnota jednoho z polí s délkou byla 1 bajt příliš krátká.

CHYBA MQFB_IIH_ERROR
Pole *Format* v MQMD určuje MQFMT_IMS, ale zpráva nezačíná platnou strukturou MQIIH.

MQFB_NOT_AUTHORIZED_FOR_IMS
ID uživatele obsažené v deskriptoru zpráv MQMD nebo heslo obsažené v poli *Authenticator* ve struktuře MQIIH selhalo při ověřování, které provedl most IMS . V důsledku toho nebyla zpráva předána produktu IMS.

CHYBA MQFB_IMS_ERROR
IMSvrátila neočekávanou chybu. Další informace o chybě naleznete v protokolu chyb produktu IBM MQ v systému, na kterém je umístěn most systému IMS .

MQFB_IMS_FIRST
Pokud má chybový kód IMS-OTMA 0x1A, jsou IMS-generované kódy zpětné vazby v rozsahu MQFB_IMS_FIRST (300) až MQFB_IMS_LAST (399). Samotný chybový kód IMS-OTMA je *Feedback* mínus MQFB_IMS_ERROR.

MQFB_IMS_LAST
Nejvyšší hodnota zpětné vazby generované produktem IMS, pokud chybový kód není 0x1A.

MQFB_IMS_NACK_1A_REASON_FIRST
Má-li chybový kód hodnotu 0x1A, IMS-generované kódy zpětné vazby 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 zpětné vazby generované produktem IMS, je-li kód chybového bajtu 0x1A

CICS-bridge feedback codes: Rozhraní CICS bridgemůže generovat následující kódy zpětné vazby:

MQFB_CICS_APPL_ABENDED
Aplikační program uvedený ve zprávě byl abnormálně ukončen. Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

MQFB_CICS_APPL_NOT_STARTED
EXEC CICS LINK pro aplikační program uvedený ve zprávě selhal. Tento kód zpětné vazby se vyskytuje pouze v poli *Reason* struktury MQDLH.

SELHÁNÍ MQFB_CICS_BRIDGE_FAILURE
CICS bridge byl nestandardně ukončen bez dokončení normálního zpracování chyb.

CHYBA MQFB_CICS_CCSID_ERROR
Identifikátor znakové sady není platný.

MQFB_CICS_CIH_ERROR
Struktura záhlaví informačního obsahu produktu CICS chybí nebo není platná.

CHYBA MQFB_CICS_COMMAGA_ERROR
Délka CICS COMMAREA není platná.

CHYBA MQFB_CICS_CORREL_ID_ERROR
Identifikátor korelace není platný.

CHYBA MQFB_CICS_DLQ_ERROR
Úloha CICS bridge nebyla schopna zkopírovat odpověď na tento požadavek do fronty nedoručených zpráv. Požadavek byl zálohován.

CHYBA MQFB_CICS_ENCODING_ERROR
Kódování není platné.

MQFB_CICS_INTERNAL_ERROR
V produktu CICS bridge došlo k neočekávané chybě.

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
Pracovní jednotka byla zálohována, z jednoho z následujících důvodů:

- Bylo zjištěno selhání během zpracování jiného požadavku v rámci stejné jednotky práce.
- Došlo k nestandardkonci CICS , zatímco jednotka práce právě probíhá.

CHYBA MQFB_CICS_UOW_ERROR
Pole řízení počtu pracovních jednotek *UOWControl* není platné.

Trasovací kódy zpětné vazby pro zprávy:

AKTIVITA MQFB_ACTIVITY
Používá se ve formátu MQFMT_EMBEDDED_PCF, aby byla povolena volba uživatelských dat následující sestavy aktivity.

MQFB_MAX_AKTIVIT
Tato zpráva je vrácena, je-li zpráva trasování cesty vyřazena, protože počet aktivit, které zpráva obsahuje, překračuje maximální povolený limit aktivit.

MQFB_NOT_FORWARDED
Tato hodnota je vrácena, je-li zpráva trasování cesty zahozena, protože má být odeslána do vzdáleného správce front, který nepodporuje zprávy trasování cesty.

MQFB_NOT_DELIVERED
Tato hodnota je vrácena, je-li zpráva trasování cesty zahozena, protože má být vložena do lokální fronty.

MQFB_UNSUPPORTED_FORWARDING
Tato hodnota je vrácena, je-li zpráva trasování cesty vyřazena, protože hodnota v parametru postoupení nebyla rozpoznána a nachází se v zamítnuté bitové masce.

MQFB_UNSUPPORTED_DELIVERY
Tato hodnota je vrácena, je-li zpráva trasování cesty vyřazena, protože hodnota v parametru doručení nebyla rozpoznána, a je v zamítnuté bitové masce.

IBM MQ kódy příčin: V případě zpráv o výjimce obsahuje *Feedback* kód příčiny IBM MQ . Mezi možné kódy příčiny patří:

MQRC_PUT_BLOKOVÁNO
(2051, X'803 ') Volání s blokováno pro frontu.

MQRC_Q_FULL
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

MQRC_MSG_TOO_BIG_FOR_Q_MGR
(2031, X'7EF') Délka zprávy je větší než maximum pro správce front.

MQRC_MSG_TOO_BIG_FOR_Q
(2030, X'7EE') Délka zprávy je větší než maximum pro frontu.

Úplný seznam kódů příčiny viz:

- Informace o produktu IBM MQ for z/OSnajdete v tématu Kódy dokončení a příčin rozhraní API.
- Informace o všech ostatních platformách najdete v tématu Kódy dokončení a příčin 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)

Určuje číselné kódování číselných dat ve zprávě. Nevztahuje se na číselná data ve struktuře MQMD jako takové. Numerické kódování definuje znázornění použité pro binární celá čísla, packed-decimální 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 znázornění znakové sady je závislé na kódování binárních celých čísel. Toto se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Na základě 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 záhlaví, makra, COPY nebo INCLUDE souborů odpovídajících prostředí, ve kterém bude aplikace spuštěna.

Aplikace, které vložila zprávy, obvykle uvádějí MQENC_NATIVE. Aplikace, které načítají zprávy, musí porovnat toto pole s hodnotou MQENC_NATIVE;, pokud se hodnoty liší, aplikace může vyžadovat převod číselných dat ve zprávě. Pomocí volby MQGMO_CONVERT požádejte správce front o převedení zprávy v rámci zpracování volání MQGET. Podrobné informace o tom, jak je pole `Encoding` konstruováno, naleznete v příručce “Kódování počítače” na stránce 890 .

Určíte-li volbu MQGMO_CONVERT na volání MQGET, bude toto pole obsahovat vstupní/výstupní pole. Hodnota zadaná aplikací je kódování, do kterého mají být v případě potřeby převedena data zprávy. Je-li konverze úspěšná nebo zbytečná, hodnota se nezmění. Pokud je konverze neúspěšná, 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)

Toto pole uvádí identifikátor znakové sady znakových dat v těle zprávy.

Poznámka: Znaková data v MQMD a dalších datových strukturách MQ , které jsou parametry na voláních, musí být ve znakové sadě správce front. Tento atribut je definován atributem **CodedCharSetId** správce front; podrobnosti o tomto atributu viz “Atributy správce front” na stránce 791 .

Je-li toto pole nastaveno na hodnotu MQCCSI_Q_MGR při volání MQGET s MQGMO_CONVERT v rámci voleb, chování se liší mezi aplikacemi klienta a serveru. Pro serverové aplikace je kódová stránka použitá pro převod znaků `CodedCharSetId` správce front; pro klientské aplikace je kódová stránka použitá pro převod znaků aktuální kódovou stránkou národního prostředí.

U klientských aplikací je MQCCSI_Q_MMGR vyplněn na základě národního prostředí klienta a nikoli podle správce front. Výjimkou z tohoto pravidla je vložení zprávy do fronty mostu IMS . Vrácená hodnota v poli *CodedCharSetId* MQMD je hodnotou CCSID správce front.

Nesmíte používat následující speciální hodnotu:

MQCCSI_APPL
Výsledkem je nesprávná hodnota v poli CodedCharSetId v deskriptoru MQMD a způsobila návratový kód MQRC_SOURCE_CCSID_ERROR (nebo MQRC_FORMAT_ERROR pro produkt z/OS). je-li zpráva přijata pomocí volání MQGET s volbou MQGMO_CONVERT, použijte tento příkaz.

Můžete použít následující speciální hodnoty:

MQCCSI_Q_MGR
Znaková data ve zprávě jsou uvedena ve znakové sadě správce front.

Na základě volání MQPUT a MQPUT1 změní správce front tuto hodnotu v deskriptoru MQMD, který je odeslán spolu se zprávou na identifikátor skutečné znakové sady správce front. Výsledkem je, že hodnota MQCCSI_Q_MGR není nikdy vrácena voláním MQGET.

MQCCSI_DEFAULT
Hodnota CodedCharSetId dat v poli *String* je definována polem CodedCharSetId ve struktuře záhlaví, která předchází struktuře MQCFH, nebo pole CodedCharSetId v MQMD, pokud je MQCFH na začátku zprávy.

MQCSI_INHERIT
Znaková data ve zprávě se nacházejí ve stejné znakové sadě jako v této struktuře. Jedná se o znakovou sadu správce front. (Pouze pro MQMD má hodnota 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 MQMD. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Nepoužívejte MQCCSI_INHERIT, je-li hodnota pole PutApplType v deskriptoru MQMD MQAT_BROKER.

MQCCSI_EMBEDDED
Znaková data ve zprávě se nacházejí ve znakové sadě s identifikátorem, který je obsažen v samotných datech zprávy. V datech zprávy může být libovolný počet identifikátorů znakových sad, který se vztahuje na různé části dat. Tato hodnota musí být použita pro zprávy PCF (s formátem MQFMT_ADMIN, MQFMT_EVENT nebo MQFMT_PCF), které obsahují data ve směsi znakových sad. Ke každé struktuře MQCFST, MQCFSL a MQCFSF obsažené ve zprávě PCF musí být zadán explicitní identifikátor znakové sady a nikoli MQCCSI_DEFAULT.

Má-li zpráva ve formátu MQFMT_EMBEDDED_PCF obsahovat data ve směsi znakových sad, nepoužívejte MQCCSI_EMBEDDED. Místo toho nastavte hodnotu MQEPH_CCSID_EMBEDDED v poli Příznaky ve struktuře MQEPH. To je ekvivalentní nastavení MQCCSI_EMBEDDED v předchozí struktuře. Každá struktura MQCFST, MQCFSL a MQCFSF obsažená v rámci zprávy PCF musí mít zadán explicitní identifikátor znakové sady a nikoli MQCCSI_DEFAULT. Další informace o struktuře MQEPH naleznete v tématu “MQEPH-Vložené záhlaví PCF” na stránce 361.

Tuto hodnotu zadejte pouze v rámci volání MQPUT a MQPUT1 . Je-li zadán na volání MQGET, brání převodu zprávy.

Na základě volání MQPUT a MQPUT1 změní správce front hodnoty MQCCSI_Q_MGR a MQCCSI_INHERIT v deskriptoru MQMD, které se odešle se zprávou, jak je popsáno výše, ale nezmění MQMD určený v rámci volání MQPUT nebo MQPUT1 . Na zadané hodnotě není provedena žádná další kontrola.

Aplikace, které načítají zprávy, musí porovnat toto pole s hodnotou, kterou aplikace očekává; pokud se hodnoty liší, aplikace může vyžadovat převod znakových dat ve zprávě.

V systému z/OSse pole Encoding deskriptoru MQMD používá k určení celočíselného kódování znakových dat v těle zprávy, pokud pole CodedCharSetId deskriptoru MQMD označuje, že znázornění znakové sady je závislé na kódování použité pro binární celá čísla. V systému Multiplatformsse předpokládá, že pořadí

bajtů znakových dat je stejné jako kódování nativního celého čísla pro platformu, na které je správce front spuštěn. To se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Určíte-li volbu MQGMO_CONVERT na volání MQGET, bude toto pole obsahovat vstupní/výstupní pole. Hodnota uvedená aplikací je identifikátor kódované znakové sady, do kterého se mají v případě potřeby konvertovat data zprávy. Pokud je konverze úspěšná nebo zbytečná, hodnota se nezmění (kromě toho, že hodnota MQCCSI_Q_MGR nebo MQCCSI_INHERIT je převedena na skutečnou hodnotu). Pokud je konverze neúspěšná, 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)

Jedná se o název, který odesilatel zprávy používá k označení povahy dat ve zprávě příjemci zprávy. Jakékoli znaky, které jsou ve znakové sadě správce front, lze zadat pro daný název, ale musíte omezit jméno na následující:

- Velká písmena A až Z
- Číselné číslice 0 až 9

Jsou-li použity jiné znaky, nemusí být možné přeložit název mezi znakové sady odesílajícího a přijímajícího správce front.

Zadejte název s mezerami do délky pole nebo použijte znak null pro ukončení názvu před koncem pole; hodnoty null a všechny následné znaky jsou považovány za mezery. Neuvádějte jméno s úvodními nebo vloženými mezerami. Pro volání MQGET vrátí správce front název doplněný mezerami do délky pole.

Správce front nekontroluje, zda je daný název v souladu s výše popsanými doporučeními.

Názvy začínající řetězcem MQ v horním, dolním a smíšeném případě mají významy, které jsou definovány správcem front; nepoužívejte názvy začínající těmito písmeny pro vlastní formáty. Vestavěné formáty správce front jsou:

MQFMT_NONE
Povaha dat není definována: data nelze převést, je-li zpráva načtena z fronty pomocí volby MQGMO_CONVERT.

Pokud uvedete MQGMO_CONVERT na volání MQGET a znaková sada nebo kódování dat ve zprávě se liší od hodnoty zadané argumentem **MsgDesc** , zpráva se vrátí s následujícím kódem dokončení a s kódem příčiny (za předpokladu, že nejsou žádné jiné chyby):

- Kód dokončení MQCC_WARNING a kód příčiny MQRC_FORMAT_ERROR, je-li data MQFMT_NONE na začátku zprávy.
- Kód dokončení MQCC_OK a kód příčiny MQRC_NONE, pokud data MQFMT_NONE jsou na konci zprávy (tj. před jedním nebo více strukturami záhlaví MQ). Struktury záhlaví MQ se převedou na požadovanou znakovou sadu a kódování v tomto případě.

Pro programovací jazyk C je také definována konstanta MQFMT_NONE_ARRAY; má stejnou hodnotu jako MQFMT_NONE, ale je to pole znaků namísto řetězce.

MQFMT_ADMIN
Jedná se o požadavek na příkaz-server nebo zprávu odpovědi ve formátu PCF (Programmable command Format). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT. Další informace o používání uživatelem programovatelných zpráv ve formátu příkazu najdete v tématu Použití programů Programmable Command Formats .

Pro programovací jazyk C je také definována konstanta MQFMT_ADMIN_ARRAY; má stejnou hodnotu jako MQFMT_ADMIN, ale je to pole znaků namísto řetězce.

MQFMT_CICS
Data zprávy začínají záhlavím informací produktu CICS MQCIH a za ním následují data aplikace. Název formátu dat aplikace je dán polem `Format` ve struktuře MQCIH.

z/OS V systému z/OSzadejte volbu MQGMO_CONVERT v rámci volání MQGET k převodu zpráv s formátem MQFMT_CICS.

Pro programovací jazyk C je také definována konstanta MQFMT_CICS_ARRAY; má stejnou hodnotu jako MQFMT_CICS, ale je to pole znaků místo řetězce.

MQFMT_COMMAND_1
Zpráva je zprávou příkazu MQSC příkazu-server, obsahující počet objektů, kód dokončení a kód příčiny. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_COMMAND_1_ARRAY ; má stejnou hodnotu jako MQFMT_COMMAND_1, ale je to pole znaků místo řetězce.

MQFMT_COMMAND_2
Jedná se o zprávu MQSC příkazu-server s odpovědí obsahující informace o požadovaných objektech. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_COMMAND_2_ARRAY ; má stejnou hodnotu jako MQFMT_COMMAND_2, ale je to pole znaků místo řetězce.

HLAVIČKA MQFMT_DEAD_LETTER_HEADER
Data zprávy začínají záhlaví nedoručených zpráv MQDLH. Data z původní zprávy bezprostředně následují za strukturou MQDLH. Název formátu původních dat zprávy je dán polem `Format` ve struktuře MQDLH. Podrobnosti o této struktuře viz “MQDLH-Záhlaví nedoručených zpráv” na stránce 349 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Sestavy COA a COD se nevygenerují pro zprávy, které mají `Format` MQFMT_DEAD_LETTER_HEADER.

Pro programovací jazyk C je také definována konstanta MQFMT_DEAD_LETTER_HEADER_ARRAY; to má stejnou hodnotu jako MQFMT_DEAD_LETTER_HEADER, ale je to pole znaků místo řetězce.

ZÁHLAVÍ MQFMT_DICT_HEADER
Data zprávy začínají záhlavím MQDH záhlaví distribučního seznamu, což zahrnuje pole záznamů MQOR a MQPMR. Za záhlavím rozdělovníku může následovat další data. Formát dalších dat (pokud existuje) je dán polem *`Format`* ve struktuře MQDH. Podrobnosti o této struktuře viz “Záhlaví MQDH-Distribution” na stránce 343 . Zprávy s formátem MQFMT_DIST_HEADER lze převést, pokud je v rámci volání MQGET zadána volba MQGMO_CONVERT.

Tento formát je podporován v následujících prostředích:

- AIX AIX
- IBM i IBM i
- Linux Linux
- Windows Windows

a pro IBM MQ MQI clients připojené k těmto systémům.

V případě programovacího jazyka C je také definována 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 cesty za předpokladu, že je hodnota příkazu PCF nastavena na hodnotu MQCMD_TRACE_ROUTE. Použití tohoto formátu umožňuje odeslání uživatelských dat spolu s trasováním přenosové cesty za předpokladu, že se jejich aplikace mohou vypořádat s předchozími parametry PCF.

Hlavička PCF musí být první hlavička, nebo se zpráva nebude považovat za zprávu přenosové cesty. To znamená, že zpráva nemůže být ve skupině a že zprávy trasování přenosové cesty nemohou být segmentovány. Je-li zpráva trasování přenosové cesty odeslána ve skupině, zpráva byla odmítnuta s kódem příčiny MQRC_MSG_NOT_ALLOWED_IN_GROUP.

Všimněte si, že MQFMT_ADMIN lze také použít pro formát zprávy přenosové cesty trasování, ale v tomto případě nelze odeslat žádná uživatelská data spolu se zprávou trasování cesty.

UDÁLOST MQFMT_EVENT
Zpráva je zpráva události MQ , která hlásí událost, která se vyskytla. Zprávy událostí mají stejnou strukturu jako programovatelné příkazy; viz Zprávy příkazu PCF , kde získáte další informace o této struktuře, a Monitorování událostí pro informace o událostech.

Zprávy událostí Version-1 mohou být převedeny ve všech prostředích, je-li volba MQGMO_CONVERT zadána při volání MQGET. Zprávy událostí Version-2 lze konvertovat pouze na z/OS.

Pro programovací jazyk C je také definována konstanta MQFMT_EVENT_ARRAY; má stejnou hodnotu jako MQFMT_EVENT, ale je to pole znaků místo řetězce.

MQFMT_IMS
Data zprávy začínají záhlavím informací produktu IMS MQIIH, za nímž následují data aplikace. Název formátu dat aplikace je uveden v poli `Format` ve struktuře MQIIH.

Podrobnosti o způsobu zpracování struktury MQIIH při použití příkazu MQGET s funkcí MQGMO_CONVERT naleznete v části “Formát (MQCHAR8)” na stránce 408 a “Formát ReplyTo(MQCHAR8)” na stránce 409.

Pro programovací jazyk C je také definována konstanta MQFMT_IMS_ARRAY; má stejnou hodnotu jako MQFMT_IMS, ale je to pole znaků místo řetězce.

MQFMT_IMS_VAR_STRING
Zpráva je řetězec proměnné IMS , který je řetězcem ve tvaru `ll`*`zz`*`ccc`, kde:

`ll`
je 2bajtová délka pole uvádějící celkovou délku položky řetězce proměnné IMS . Tato délka se rovná délce `ll` (2 bajty) a délce *`zz`* (2 bajtů) a délky samotného znakového řetězce. `ll` je 2bajtové binární celé číslo v kódování zadaném v poli `Encoding` .

`zz`
je 2bajtové pole obsahující příznaky, které jsou významné pro IMS. *`zz`* je bytový řetězec skládající se ze dvou polí MQBYTE a je přenášen beze změny od odesílatele k příjemci (to znamená, že *`zz`* není předmětem žádné konverze).

`ccc`
je řetězec znaků s proměnnou délkou obsahující `ll-4` znaků. *`ccc`* je ve znakové sadě zadané v poli `CodedCharSetId` .

V systému z/OSse data zprávy mohou skládat z posloupnosti řetězcových řetězců IMS , které jsou společně s každým řetězcem ve tvaru `llzzccc`. Mezi následnými řetězci proměnných IMS nesmějí být přeskočeny žádné bajty. To znamená, že pokud má první řetězec lichou délku, druhý řetězec bude špatně zarovnaný, to znamená, že nebude začínat na hranici, která je násobkem dvou. Buďte opatrní při vytváření takových řetězců na počítačích, které vyžadují sladění elementárních datových typů.

Použijte volbu MQGMO_CONVERT na volání MQGET k převedení zpráv, které mají formát MQFMT_IMS_VAR_STRING.

Pro programovací jazyk C je také definována konstanta MQFMT_IMS_VAR_STRING_ARRAY; má stejnou hodnotu jako MQFMT_IMS_VAR_STRING, ale je to pole znaků místo řetězce.

ROZŠÍŘENÍ MQFMT_MD_EXTENSION
Data zprávy začínají na rozšíření deskriptoru zpráv MQMDE a volitelně jsou následována jinými daty (obvykle data zprávy aplikace). Název formátu, znaková sada a kódování dat, které následují za MQMDE, jsou poskytnuty poli `Format`, `CodedCharSetId`a `Encoding` v MQMDE. Podrobnosti o této struktuře viz “MQMDE-Rozšíření deskriptoru zpráv” na stránce 468 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_MD_EXTENSION_ARRAY; má stejnou hodnotu jako MQFMT_MD_EXTENSION, ale je to pole znaků namísto řetězce.

MQFMT_PCF
Zpráva je uživatelem definovaná zpráva, která odpovídá struktuře zprávy PCF (Programmable command format). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT. Další informace o používání uživatelem programovatelných zpráv ve formátu příkazu najdete v tématu Použití programů Programmable Command Formats .

Pro programovací jazyk C je také definován konstantní MQFMT_PCF_ARRAY; má stejnou hodnotu jako MQFMT_PCF, ale je to pole znaků místo řetězce.

MQFMT_REF_MSG_HEADER
Data zprávy začínají odkazem na záhlaví MQRMH a volitelně jsou následována jinými 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 544 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Tento formát je podporován v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro IBM MQ MQI clients připojené k těmto systémům.

Pro programovací jazyk C je také definována konstanta MQFMT_REF_MSG_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_REF_MSG_HEADER, ale je to pole znaků místo řetězce.

ZÁHLAVÍ MQFMT_RF_HEADER
Data zprávy začínají na pravidla a formátovací záhlaví MQRFH a volitelně jsou následována jinými daty. Název formátu, znaková sada a kódování dat (pokud existuje) je dána poli `Format`, `CodedCharSetId`a `Encoding` v MQRFH. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_RF_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_RF_HEADER, ale je to pole znaků namísto řetězce.

MQFMT_RF_HEADER_2
Data zprávy začínají s pravidly version-2 a formátováním záhlaví MQRFH2a volitelně jsou následována jinými daty. Název formátu, znaková sada a kódování nepovinných dat (pokud existuje) je dána poli `Format`, `CodedCharSetId`a `Encoding` v MQRFH2. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_RF_HEADER_2_ARRAY ; má stejnou hodnotu jako MQFMT_RF_HEADER_2, ale je to pole znaků místo řetězce.

ŘETĚZEC MQFMT_STRING
Data zprávy aplikace mohou být buď řetězec SBCS (jednobajtová znaková sada), nebo řetězec DBCS (dvojbajtová znaková sada). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definována konstanta MQFMT_STRING_ARRAY; má stejnou hodnotu jako MQFMT_STRING, ale je to pole znaků místo řetězce.

SPOUŠTĚČ MQFMT_TRIGGER
Zpráva je zpráva spouštěče, která je popsána strukturou MQTM. Podrobnosti o této struktuře viz “MQTM-zpráva spouštěče” na stránce 594 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Pro programovací jazyk C je také definován konstantní 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 s informacemi o práci, za nímž následují data aplikace. Název formátu dat aplikace je dán polem `Format` ve struktuře MQWIH.

z/OS V systému z/OSurčete volbu MQGMO_CONVERT u volání MQGET pro převod uživatelských dat ve zprávách, které mají formát MQFMT_WORK_INFO_HEADER. Struktura MQWIH je však vždy vrácena ve znakové sadě a kódování správce front (to znamená, že struktura MQWIH je převedena bez ohledu na to, zda je zadána volba MQGMO_CONVERT).

Pro programovací jazyk C je také definována konstanta MQFMT_WORK_INFO_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_WORK_INFO_HEADER, ale je to pole znaků namísto řetězce.

ZÁHLAVÍ MQFMT_XMIT_Q_HEADER
Data zprávy začínají s hlavičkou přenosové fronty MQXQH. Data z původní zprávy bezprostředně následují za strukturou MQXQH. Název formátu původních dat zprávy je dán polem `Format` ve struktuře MQMD, která je součástí záhlaví MQXQH přenosové fronty. Podrobnosti o této struktuře viz "MQXQH-záhlaví přenosové fronty" na stránce 613 .

Sestavy COA a COD se nevygenerují pro zprávy, které mají `Format` MQFMT_XMIT_Q_HEADER.

Pro programovací jazyk C je také definována konstanta MQFMT_XMIT_Q_HEADER_ARRAY; má stejnou hodnotu jako MQFMT_XMIT_Q_HEADER, ale je to pole znaků namísto 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 volání MQPUT a MQPUT1 musí být hodnota větší než nula nebo rovna nule; hodnota nula je nejnižší priorita. Je možné použít také následující speciální hodnotu:

MQPRI_PRIORITY_AS_Q_DEF

- Je-li fronta fronta klastru, je priorita zprávy převzata z atributu **DefPriority** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, na 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?

  Při umístění zprávy do cílové fronty je hodnota parametru *DefPriority* zkopírována do pole *Priority* . Je-li produkt *DefPriority* později změněn, nebudou ovlivněny zprávy, které již byly umístěny do fronty.
- Není-li fronta fronta klastru, je priorita zprávy převzata z atributu **DefPriority** definovaného ve správci front *local* , a to i v případě, že je správce cílové fronty vzdálený.

  Pokud je v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude použita výchozí priorita z hodnoty tohoto atributu v definici *první* v cestě. To může být:

  – Fronta aliasů
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta *DefXmitQName*)

  Hodnota parametru *DefPriority* se při vložení zprávy zkopíruje do pole *Priority* . Pokud je produkt *DefPriority* později změněn, zprávy, které již byly vloženy, nejsou ovlivněny.

Hodnota vrácená voláním MQGET je vždy větší než nebo rovna nule; hodnota MQPRI_PRIORITY_AS_Q_DEF není nikdy vrácena.

Je-li zpráva vložena s prioritou vyšší, než je maximum podporované lokálním správcem front (toto maximum je dáno atributem správce front **MaxPriority**), zpráva je přijata správcem front, ale zařazena do fronty v maximální prioritě správce front; volání MQPUT nebo MQPUT1 je dokončeno s operací MQCC_WARNING a kódem příčiny MQRC_PRIORITY_EXCEEDS_MAXIMUM. Pole *Priority* si však zachovává hodnotu určenou aplikací, která vložila zprávu.

Pokud je v produktu z/OSzpráva s číslem MsgSeqČíslo 1 vložena do fronty, která má posloupnost doručení zprávy MQMDS_PRIORITY a typ indexu MQIT_GROUP_ID, může tato fronta zpracovat zprávu s jinou prioritou. Pokud byla zpráva umístěna do fronty s prioritou 0 nebo 1, je zpracována, jako by měla prioritu 2. Důvodem je to, že pořadí zpráv umístěných na tomto typu fronty je optimalizováno tak, aby umožňovaly účinné testy úplnosti skupiny. Další informace o posloupnosti doručení zpráv MQMS_PRIORITY a typu indexu MQIT_GROUP_ID naleznete v části Atribut posloupnostiMsgDelivery.

Při odpovídání na zprávu musí aplikace používat prioritu zprávy požadavku pro zprávu odpovědi. V jiných situacích umožňuje zadání funkce MQPRI_PRIORITY_AS_Q_DEF, aby bylo možné provést vyladění priority beze 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)

Označuje, zda zpráva přežije selhání systému a restartuje správce front. Pro volání MQPUT a MQPUT1 musí být hodnota jedna z následujících:

MQPER_PERSISTENT
Zpráva přežije selhání systému a restartuje správce front. Jakmile byla zpráva vložena a jednotka práce, ve které byla vložena, byla potvrzena (je-li zpráva vložena jako součást pracovní jednotky), zpráva je uchována v pomocné paměti. Zůstane tam, dokud nebude zpráva odebrána z fronty, a jednotka práce, ve které byl získán, byl potvrzen (pokud je zpráva načtena jako část pracovní jednotky).

Když se do vzdálené fronty odešle trvalá zpráva, mechanizmus uložit-a-předat uchovává zprávu v každém správci front podél cesty k místu určení, dokud není známo, že tato zpráva dorazila do dalšího správce front.

Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, které jsou mapovány na objekt CFSTRUCT na úrovni CFLEVEL (2) nebo nižší, nebo kde je objekt CFSTRUCT definován jako RECOVER (NO).

Trvalé zprávy lze umístit do trvalých dynamických front a předdefinovaných front.

MQPER_NOT_PERSISTENT
Zpráva obvykle nepřežije selhání systému nebo správce front se restartuje. To platí i v případě, že se při restartování správce front nachází neporušená kopie zprávy v pomocné paměti.

V případě NPMCLASS (HIGH) fronty přechodných zpráv přežije normální ukončení činnosti správce front a restart.

V případě sdílených front se přechodné zprávy mohou v rámci skupiny sdílení front znovu spustit ve skupině sdílení front, ale nepřežijí selhání prostředku CF použitého k ukládání zpráv ve sdílených frontách.

MQPER_PERSISTENCE_AS_Q_DEF

- Je-li fronta fronta klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, na 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?

Při umístění zprávy do cílové fronty je hodnota parametru *DefPersistence* zkopírována do pole *Persistence* . Je-li produkt *DefPersistence* později změněn, nebudou ovlivněny zprávy, které již byly umístěny do fronty.

- Pokud fronta není fronta klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného ve správci front *local* , a to i v případě, že je cílový správce front vzdálený.

  Pokud je v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude použita výchozí perzistence z hodnoty tohoto atributu v definici *first* v cestě. To může být:

  – Fronta aliasů
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta *DefXmitQName*)

  Hodnota parametru *DefPersistence* se při vložení zprávy zkopíruje do pole *Persistence* . Pokud je produkt *DefPersistence* později změněn, zprávy, které již byly vloženy, nejsou ovlivněny.

Trvalé i přechodné zprávy mohou existovat ve stejné frontě.

Při odpovídání na zprávu musí aplikace používat trvání 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)

Jedná se o bajtový řetězec, který se používá k rozlišení jedné zprávy od druhé. Obecně platí, že žádné dvě zprávy by neměly mít stejný identifikátor zprávy, ačkoli správce front tento stav nezakázal. Identifikátor zprávy je trvalou vlastností zprávy a uchovává se přes restarty správce front. Protože identifikátor zprávy je bajtový řetězec a ne znakový řetězec, identifikátor zprávy se nekonvertuje mezi znakovými sadami, když se tok zpráv z jednoho správce front do jiného správce front.

Pro volání MQPUT a MQPUT1 platí, že pokud aplikace zadá MQMI_NONE nebo MQPMO_NEW_MSG_ID, správce front vygeneruje jedinečný identifikátor zprávy.[3]je-li zpráva vložena a umístí ji do deskriptoru zprávy odeslaného se zprávou. Správce front také vrátí tento identifikátor zprávy v deskriptoru zpráv, který patří do odesílající aplikace. Aplikace může tuto hodnotu použít k zaznamenání informací o konkrétních zprávách a k odpovědi na dotazy z jiných částí aplikace.

Je-li zpráva vložena do tématu, správce front generuje jedinečné identifikátory zpráv, které jsou nezbytné pro každou publikovanou zprávu. Pokud aplikace zadá MQPMO_NEW_MSG_ID, správce front vygeneruje jedinečný identifikátor zprávy, který se vrátí na výstup. Je-li hodnota MQMI_NONE určena aplikací, hodnota pole *MsgId* v deskriptoru MQMD se při návratu z volání nezmění.

Další podrobnosti o zachovaných příručkách naleznete v popisu vlastnosti MQPMO_RETAIN v příručce “Volby MQPMO (MQLONG)” na stránce 501 .

Pokud je zpráva vložena do distribučního seznamu, správce front generuje podle potřeby jedinečné identifikátory zpráv, ale hodnota pole *MsgId* v produktu MQMD se při návratu z volání nezmění, i když bylo

[3] *MsgId* generovaný správcem front se skládá z 4bajtového identifikátoru produktu (AMQ ¬ nebo CSQ ¬ v systému ASCII nebo EBCDIC, kde společnost IBM představuje prázdný znak) následována implementací jedinečného řetězce specifickou pro produkt product-specific. V IBM MQ toto obsahuje prvních 12 znaků názvu správce front a hodnoty odvozené ze systémových hodin. Všichni správci front, kteří mohou komunikovat, musí mít proto 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 se neměnily zpět. Aby se vyloučila možnost identifikátoru zprávy generovaného správcem front, který duplikuje jeden generovaný aplikací, aplikace se musí vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikace však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

zadáno MQMI_NONE nebo MQPMO_NEW_MSG_ID. Pokud aplikace potřebuje znát identifikátory zpráv generované správcem front, musí aplikace poskytnout záznamy MQPMR obsahující pole *MsgId* .

Odesílající aplikace může také určit hodnotu pro jiný identifikátor zprávy než MQMI_NONE;, což zastaví správce front, který generuje jedinečný identifikátor zprávy. Aplikace, která je přesměrováním zprávy, může použít tuto volbu k šíření identifikátoru zprávy původní zprávy.

Správce front toto pole nepoužívá, s výjimkou následujících položek:

- Generovat jedinečnou hodnotu, je-li požadována, jak je popsáno výše
- Doručí 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 s hlášením, nastaví pole *MsgId* tak, jak je určeno polem *Report* původní zprávy, buď MQRO_NEW_MSG_ID, nebo MQRO_PASS_MSG_ID. Aplikace, které generují zprávy hlášení, musí také provést toto.

Pro volání MQGET je *MsgId* jedním z pěti polí, které lze použít k načtení konkrétní zprávy z fronty. Volání MQGET obvykle vrátí další zprávu ve frontě, ale konkrétní zprávu lze získat zadáním jednoho nebo více pěti výběrových kritérií v libovolné kombinaci; tato pole jsou:

- *MsgId*
- *CorrelId*
- *GroupId*
- *MsgSeqNumber*
- *Offset*

Aplikace nastaví jednu nebo více těchto polí na požadované hodnoty a poté nastaví odpovídající volby MQMO_ * v poli *MatchOptions* v produktu MQGMO k použití těchto polí jako kritérií výběru. Pouze zprávy, které mají uvedené hodnoty v těchto polích, jsou kandidáty na načtení. Předvolba pro pole *MatchOptions* (pokud není změněna aplikací) má odpovídat jak identifikátoru zprávy, tak i identifikátoru korelace.

V systému z/OSjsou kritéria výběru, která můžete použít, omezena typem indexu použitého pro frontu. Viz atribut fronty produktu **IndexType** , kde jsou další podrobnosti.

Za normálních okolností je vrácena zpráva *první* ve frontě, která splňuje kritéria výběru. Je-li však zadán parametr MQGMO_BRONEXT NEXT, bude vrácena zpráva *další* , která splní kritéria výběru; skenování této zprávy začíná zprávou *následující* aktuální pozicí kurzoru.

Poznámka: Fronta je skenována sekvenčně pro zprávu, která odpovídá kritériím výběru, takže časy načítání jsou pomalejší než v případě, že nejsou uvedena žádná kritéria výběru, zvláště pokud se má před nalezenou vhodnou zprávou vyhledat mnoho zpráv. Výjimky z této skutečnosti jsou:

- Multi volání MQGET s parametrem *CorrelId* on 64-bit Multiplatforms, kde index *CorrelId* odstraňuje potřebu provedení skutečného sekvenčního skenování.
- z/OS volání MQGET s parametrem *IndexType* v systému z/OS.

V obou těchto případech se zlepší výkon načítání.

Další informace o tom, jak jsou kritéria výběru použita v různých situacích, viz Tabulka 496 na stránce 391 .

Když uvedete MQMI_NONE, protože identifikátor zprávy má stejný účinek jako neuvedení MQMO_MATCH_MSG_ID, to znamená, že *any* identifikátor zprávy odpovídá.

This field is ignored if the MQGMO_MSG_UNDER_CURSOR option is specified in the **GetMsgOpts** parameter on the MQGET call.

Při návratu z volání MQGET je pole *MsgId* nastaveno na identifikátor zprávy vrácené zprávy (je-li k dispozici).

Je možné použít následující speciální hodnotu:

MQMI_NONE
Není uveden žádný identifikátor zprávy.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQMI_NONE_ARRAY; má stejnou hodnotu jako MQMI_NONE, ale je to pole znaků místo řetězce.

Jedná se o 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)

Pole CorrelId je vlastnost v záhlaví zprávy, které lze použít k identifikaci určité zprávy nebo skupiny zpráv.

Jedná se o bajtový řetězec, který může aplikace použít ke vztažení jedné zprávy k jiné, nebo ke vztažení zprávy k jiné práci, kterou aplikace provádí. Identifikátor korelace je trvalou vlastností zprávy a uchovává se po restartu správce front. Vzhledem k tomu, že identifikátor korelace je bajtový řetězec a nikoli znakový řetězec, není korelační identifikátor převeden mezi znakovými sadami, když se tok zpráv z jednoho správce front do jiného správce front.

Pro volání MQPUT a MQPUT1 může aplikace určit libovolnou hodnotu. Správce front tuto hodnotu přenáší se zprávou a doručuje ji aplikaci, která vydá požadavek na získání pro zprávu.

Pokud aplikace určuje MQPMO_NEW_CORREL_ID, vygeneruje správce front jedinečný korelační identifikátor, který je odeslán se zprávou, a také se vrátí do odesílající aplikace na výstupu z volání MQPUT nebo MQPUT1 .

Identifikátor korelace generovaný správcem front se skládá z tříbajtového identifikátoru produktu (AMQ nebo CSQ v systému ASCII nebo EBCDIC), za nímž následuje jeden rezervovaný bajt a specifická implementace specifické pro daný řetězec. V produktu IBM MQ tento řetězec implementace specifický pro daný produkt obsahuje prvních 12 znaků názvu správce front a hodnotu odvozenou ze systémových hodin. Všichni správci front, kteří mohou interkomunikovat, musí mít proto názvy, které se liší od prvních 12 znaků, aby se zajistilo, že identifikátory zpráv jsou jedinečné. Schopnost generovat jedinečný řetězec také závisí na tom, že systémové hodiny se neměnily zpět. Aby se vyloučila možnost identifikátoru zprávy generovaného správcem front, který duplikuje jeden generovaný aplikací, aplikace se musí vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikace však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

Tento generovaný korelační identifikátor je uložen se zprávou, je-li zachován, a je použit jako identifikátor korelace, když je zpráva odeslána jako publikace odběratelům, kteří specifikují MQCI_NONE v poli ID SubCorrelv MQSD, předaný v volání MQSUB. Další informace o zachovaných příručkách naleznete v tématu Volby MQPMO .

Když správce front nebo agent kanálu zpráv vygeneruje zprávu s hlášením, nastaví pole *CorrelId* způsobem, který je zadán polem *Report* původní zprávy, buď MQRO_COPY_MSG_ID_TO_CORREL_ID, nebo MQRO_PASS_CORREL_ID. Aplikace, které generují zprávy hlášení, musí také provést toto.

Pro volání MQGET je *CorrelId* jedním z pěti polí, které lze použít k výběru konkrétní zprávy, která má být načtena z fronty. Podrobné informace o tom, jak určit hodnoty pro toto pole, najdete v popisu pole *MsgId* .

Zadání hodnoty MQCI_NONE jako korelačního identifikátoru má stejný účinek jako neuvedení hodnoty MQMO_MATCH_CORREL_ID, tj. *libovolný* korelační identifikátor se bude shodovat.

Je-li v parametru **GetMsgOpts** ve volání MQGET zadána volba MQGMO_MSG_UNDER_CURSOR, je toto pole ignorováno.

Při návratu z volání MQGET je pole *CorrelId* nastaveno na identifikátor korelace vrácené zprávy (je-li k dispozici).

Mohou být použity následující speciální hodnoty:

MQCI_NONE
Není uveden žádný korelační identifikátor.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQCI_NONE_ARRAY; hodnota má stejnou hodnotu jako MQCI_NONE, ale je to pole znaků namísto řetězce.

MQCI_NEW_SESSION
Zpráva je začátkem nové relace.

Tato hodnota je rozpoznána produktem CICS bridge jako označení začátku nové relace, tj. začátek nové posloupnosti zpráv.

Pro programovací jazyk C je také definována konstanta MQCI_NEW_SESSION_ARRAY; má stejnou hodnotu jako MQCI_NEW_SESSION, ale je to pole znaků místo řetězce.

U volání MQGET se jedná o vstupní/výstupní pole. Pro volání MQPUT a MQPUT1 je toto vstupní pole, pokud MQPMO_NEW_CORREL_ID není zadán, a výstupní pole, pokud je zadáno 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 je používáno správcem front.

BackoutCount (MQLONG)

Jedná se o počet případů, kdy byla zpráva již dříve vrácena voláním MQGET jako součást pracovní jednotky, a následně byla vrácena. Pomáhá aplikaci při zjišťování chyb zpracování, které jsou založeny na obsahu zprávy. Počet vylučuje volání MQGET, která uvádí jakoukoli z voleb MQGMO_BROWSE_ *.

Přesnost tohoto počtu je ovlivněna atributem fronty **HardenGetBackout** ; viz “Atributy pro fronty” na stránce 827.

V systému z/OShodnota 255 znamená, že zpráva byla vrácena 255 nebo vícekrát. Vrácená hodnota není nikdy větší než 255.

Toto je výstupní pole pro volání MQGET. Pro volání MQPUT a MQPUT1 je ignorována. Počáteční hodnota tohoto pole je 0.

ReplyToQ (MQCHAR48)

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, který je definován ve správci front identifikovaném příkazem *ReplyToQMgr*. Tato fronta nesmí být modelová fronta, ačkoli odesílající správce front toto neověří, když je zpráva vložena.

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. Zadaná hodnota (nebo náhrada) se však předává aplikaci, která vydala požadavek na získání pro zprávu, bez ohledu na typ zprávy.

Je-li pole *ReplyToQMgr* prázdné, správce lokální fronty vyhledá ve svých vlastních definicích fronty název *ReplyToQ* . Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota *ReplyToQ* v předané zprávě je nahrazena hodnotou atributu **RemoteQName** z definice vzdálené fronty a tato hodnota je vrácena v deskriptoru zpráv, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, *ReplyToQ* se nemění.

Je-li jméno uvedeno, může obsahovat koncové mezery; první znak null a znaky následující za ním jsou považovány za mezery. Jinak se nekontroluje, zda název odpovídá pravidlům pojmenování pro fronty; to je také pravda pro přenesený název, pokud je *ReplyToQ* nahrazen v přenesené zprávě. Jediná kontrola je, že jméno bylo uvedeno, pokud to okolnosti vyžadují.

Není-li fronta pro odpověď vyžadována, nastavte pole *ReplyToQ* na prázdné znaky nebo (v programovacím jazyce C) na řetězec s hodnotou null nebo na jeden nebo více mezer následovaný znakem null; nenechávejte pole neinicializované.

U volání MQGET správce front vždy vrátí název doplněný mezerami na délku pole.

Pokud nelze doručit zprávu, která vyžaduje zprávu sestavy, a zpráva sestavy také nemůže být doručena do zadané fronty, původní zpráva i zpráva sestavy jdou do fronty nedoručených zpráv (viz atribut **DeadLetterQName** , který je popsán v části “Atributy správce front” na stránce 791).

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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Správce front ReplyToQMgr (MQCHAR48)

Jedná se o název správce front, do kterého má být odeslána zpráva s odpovědí nebo zpráva se sestavou. *ReplyToQ* je lokální název fronty, která je definovaná na tomto správci front.

Je-li pole *ReplyToQMgr* prázdné, správce lokální fronty vyhledá ve svých definicích front název *ReplyToQ* . Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota *ReplyToQMgr* v předané zprávě je nahrazena hodnotou atributu **RemoteQMgrName** z definice vzdálené fronty a tato hodnota je vrácena v deskriptoru zpráv, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, *ReplyToQMgr* přenášený se zprávou je název lokálního správce front.

Je-li jméno uvedeno, může obsahovat koncové mezery; první znak null a znaky následující za ním jsou považovány za mezery. Jinak se nekontroluje, zda název odpovídá pravidlům pojmenování pro správce front, nebo že tento název je známý odesílajícímu správci front; to je také pravda pro přenesený název, pokud je *ReplyToQMgr* nahrazen v přenesené zprávě.

Není-li fronta pro odpověď vyžadována, nastavte pole *ReplyToQMgr* na prázdné znaky nebo (v programovacím jazyce C) na řetězec s hodnotou null nebo na jeden nebo více mezer následovaný znakem null; nenechávejte pole neinicializované.

U volání MQGET správce front vždy vrátí název doplněný 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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

UserIdentifier (MQCHAR12)

Toto je část **kontextu identity** zprávy. Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 419 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)

Jedná se o token evidence, který je součástí *kontextu identity* zprávy. Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 419 ; také viz 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 MQPMO (MQLONG)" na stránce 501), 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.

Data ApplIdentity(MQCHAR32)

Toto je část **kontextu identity** zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 419 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.

Typ PutAppl(MQLONG)

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 419 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)

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 419 a Kontext zprávy.

V 9.2.0 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)

Toto je datum, kdy byla zpráva vložena, a je součástí **původního kontextu** zprávy. Další informace o kontextu zprávy viz “MQMD-Deskriptor zpráv” na stránce 419 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)

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 419 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.

Data ApplOrigin(MQCHAR4)

Toto je část *původního kontextu* zprávy. Další informace o kontextu zprávy viz "MQMD-Deskriptor zpráv" na stránce 419 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)

Jedná se o bajtový řetězec, který se používá k identifikaci konkrétní skupiny zpráv nebo logické zprávy, do níž náleží fyzická zpráva. *GroupId* se také používá, pokud je pro zprávu povoleno segmentace. Ve všech těchto případech má *GroupId* hodnotu jinou než null a v poli *MsgFlags* je nastaven jeden nebo více z následujících parametrů:

- MQMF_MSG_IN_GROUP
- MQM_LAST_MSG_IN_GROUP
- SEGMENT MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- MQMF_SEGMENTATION_ALLOWED

Není-li nastaven žádný z těchto parametrů, má *GroupId* speciální hodnotu null MQGI_NONE.

Aplikace nevyžaduje nastavení tohoto pole v rámci volání MQPUT nebo MQGET, pokud:

- V případě volání MQPUT je zadán parametr MQPMO_LOGICAL_ORDER.
- V rámci volání MQGET není zadán parametr 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 produkt *GroupId* nastaven na příslušnou hodnotu.

Skupiny zpráv a segmenty mohou být zpracovány správně pouze tehdy, je-li identifikátor skupiny jedinečný. Z tohoto důvodu *aplikace nesmí generovat své vlastní identifikátory skupin* ; místo toho musí aplikace provést jednu z následujících možností:

- Je-li zadán parametr 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 nevyžaduje provedení žádné speciální akce. Toto je doporučený postup.
- Není-li parametr MQPMO_LOGICAL_ORDER zadán, musí aplikace požádat správce front o vygenerování identifikátoru skupiny nastavením parametru *GroupId* na hodnotu MQGI_NONE v prvním volání MQPUT nebo MQPUT1 pro zprávu ve skupině nebo segmentu logické zprávy. Identifikátor skupiny vrácený správcem front na výstupu z tohoto volání musí být potom použit pro zbývající zprávy ve skupině nebo segmentech logické zprávy. Pokud skupina zpráv obsahuje segmentované zprávy, musí být použit stejný identifikátor skupiny pro všechny segmenty a zprávy ve skupině.

Není-li zadáno MQPMO_LOGICAL_ORDER, zprávy ve skupinách a segmentech logických zpráv lze vložit do libovolného pořadí (například v opačném pořadí), ale identifikátor skupiny musí být alokován voláním MQPUT *first* MQPUT nebo MQPUT1 , které bylo vydáno pro některou z těchto zpráv.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou ve Fyzickém pořadí na frontě. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou, pokud je otevřený objekt jedinou frontou a nikoli distribučními seznamy, ale ponechá ji nezměněnou, pokud je objekt otevřený distribučnímu seznamu. V případě, že aplikace potřebuje znát generované identifikátory skupin, musí v případě potřeby poskytnout záznamy MQPMR obsahující pole *GroupId* .

Na vstupu do volání MQGET používá správce front hodnotu popsanou v části Tabulka 496 na stránce 391. Na výstupu z 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. Toto je hodnota, která se používá pro zprávy, které nejsou ve skupinách, ne segmenty logických zpráv a pro které segmentaci není povoleno.

Pro programovací jazyk C je také definována 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 *Version* je menší než MQMD_VERSION_2.

Počet MsgSeqNumber (MQLONG)

Jedná se o pořadové číslo logické zprávy v rámci skupiny.

Pořadová čísla začínají hodnotou 1 a u každé nové logické zprávy ve skupině se zvyšují o 1 až do maximální hodnoty 999 999 999. Fyzická zpráva, která se nenachází ve skupině, má pořadové číslo 1.

Aplikace nemusí toto pole nastavit v rámci volání MQPUT nebo MQGET, pokud:

- V případě volání MQPUT je zadán parametr MQPMO_LOGICAL_ORDER.
- Na volání MQGET není zadáno 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 produkt *MsgSeqNumber* nastaven na příslušnou hodnotu.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou ve Fyzickém pořadí na frontě. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou.

Ve vstupu do volání MQGET používá správce front hodnotu zobrazenou v části Tabulka 496 na stránce 391. Na výstupu z volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je jedna. Toto pole je ignorováno, pokud *Version* je menší než MQMD_VERSION_2.

Offset (MQLONG)

Toto je posun v bajtech dat ve fyzické zprávě od začátku logické zprávy, z níž jsou data součástí. Tato data se nazývají *segment*. Posunutí je v rozsahu od 0 do 999 999 999. Fyzická zpráva, která není segmentem logické zprávy, má offsetovou hodnotu nula.

Aplikace nevyžaduje nastavení tohoto pole v rámci volání MQPUT nebo MQGET, pokud:

- V případě volání MQPUT je zadán parametr MQPMO_LOGICAL_ORDER.
- V rámci volání MQGET není zadán 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 produkt *Offset* nastaven na příslušnou hodnotu.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou ve Fyzickém pořadí na frontě. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou.

Pro hlášení zpráv sestavy v segmentu logické zprávy je pole *OriginalLength* (za předpokladu, že není MQOL_UNDEFINED) použito k aktualizaci offsetu v informacích o segmentu uchovaných správcem front.

Ve vstupu do volání MQGET používá správce front hodnotu zobrazenou v části Tabulka 496 na stránce 391. Na výstupu z volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je nula. Toto pole je ignorováno, pokud *Version* je menší než MQMD_VERSION_2.

MsgFlags (MQLONG)

MsgFlags jsou příznaky, které určují atributy zprávy, nebo řídí jejich zpracování.

MsgFlags jsou rozděleny do následujících kategorií:

- Příznaky segmentace
- Příznaky stavu

Příznaky segmentace: Je-li zpráva příliš velká pro frontu, pokus o vložení zprávy do fronty se obvykle nezdaří. Segmentace je technika, pomocí níž správce front nebo aplikace rozdělí zprávu na menší části, které se nazývají segmenty, a umístí každý segment do fronty jako samostatnou fyzickou zprávu. Aplikace, která načte zprávu, může buď načíst segmenty jednu po druhé, nebo požádat správce front, aby znovu složl segmenty do jediné zprávy vrácené voláním MQGET. Toho je dosaženo určením volby MQGMO_COMPLEE_MSG na volání MQGET a poskytnutím vyrovnávací paměti, která je dostatečně velká, aby pojmula úplnou zprávu. (Podrobnosti o volbě MQGMO_COMPLETE_MSG viz “MQGMO-Volby získání zprávy” na stránce 366 .) Zpráva může být segmentována v odesílajícím správci front v intermediačních správci front nebo v cílovém správci front.

Chcete-li řídit segmentaci zprávy, můžete určit jednu z následujících možností:

MQMF_SEGMENTATION_BLOKOVÁNO
Tato volba zabraňuje tomu, aby byla zpráva rozdělena do segmentů správcem front. Je-li pro zprávu, která je již segmentem, zadána, zabrání tomu, aby segment byl rozdělen do menších segmentů.

Hodnota tohoto parametru je binární nula. Toto nastavení je výchozí.

MQMF_SEGMENTATION_ALLOWED
Tato volba umožňuje rozdělení zprávy do segmentů prostřednictvím správce front. Je-li pro zprávu, která je již segmentem, zadána, tato volba umožňuje rozdělení segmentu do menších segmentů. MQMF_SEGMENTATION_ALLOWED lze nastavit bez nastavení hodnoty MQMF_SEGMENT nebo MQMF_LAST_SEGMENT.

- V systému z/OSsprávce front nepodporuje segmentaci zpráv. Je-li zpráva příliš velká pro frontu, 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 segmentovat zprávy ve vzdáleném správci front.

Když správce front segmentuje zprávu, aktivuje správce front v kopii MQMD, který je odeslán s každým segmentem, příznak MQMF_SEGMENT, ale nezmění nastavení těchto parametrů v deskriptoru MQMD, který je poskytován aplikací v rámci volání MQPUT nebo MQPUT1 . Pro poslední segment v logické zprávě správce front zapíná také příznak MQMF_LAST_SEGMENT v deskriptoru MQMD, který se odesílá s tímto segmentem.

Poznámka: Dávejte pozor při vkládání zpráv s MQMF_SEGMENTATION_ALLOWED, ale bez MQPMO_LOGICAL_ORDER. Je-li zpráva:

- ne segment a
- Není ve skupině a
- Nepředává se,

musí aplikace po volání *each* MQPUT nebo MQPUT1 resetovat pole *GroupId* na hodnotu MQGI_NONE, aby správce front mohl generovat jedinečný identifikátor skupiny pro každou zprávu. Pokud to není provedeno, nespřízněné zprávy mohou mít stejný identifikátor skupiny, což může vést k následnému chybnému zpracování. Další informace o tom, kdy obnovit pole *GroupId* , najdete v popisech pole *GroupId* a volby MQPMO_LOGICAL_ORDER.

Správce front rozdělí zprávy do segmentů podle potřeby tak, aby segmenty (a všechny požadované údaje záhlaví) vešly do fronty. Pro velikost segmentu generovaného správcem front však existuje nižší mezní hodnota a pouze poslední segment vytvořený ze zprávy může být menší než tento limit (dolní mez velikosti segmentu generovaného aplikací je jeden bajt). Segmenty generované správcem front mohou mít nestejnou délku. Správce front zpracovává zprávu následujícím způsobem:

- Uživatelsky definované formáty jsou rozděleny na hranicích, 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 přítomná data. Správce front však nikdy nerozděluje zprávu ve struktuře záhlaví IBM MQ . To znamená, že segment obsahující jednu strukturu záhlaví MQ nemůže být dále rozdělen správcem front, a výsledkem je minimální možná velikost segmentu pro tuto zprávu větší než 16 bajtů.

  Druhý nebo pozdější segment generovaný správcem front začíná jedním z následujících způsobů:

  – Struktura záhlaví MQ
  – Začátek dat zprávy aplikace
  – Část cesty prostřednictvím dat zprávy aplikace
- MQFMT_STRING je rozdělen bez ohledu na charakter přítomného data (SBCS, DBCS, nebo smíšených SBCS/DBCS). Je-li řetězec DBCS nebo smíšený SBCS/DBCS, může tento řetězec vyústit v segmenty, které nelze převést z jedné znakové sady na jinou. Správce front nikdy nerozděluje zprávy MQFMT_STRING do segmentů, které jsou menší než 16 bajtů (kromě posledního segmentu).
- Správce front nastaví pole *Format*, *CodedCharSetId*a *Encoding* v deskriptoru MQMD každého segmentu, aby správně popisovala data přítomná na *začátku* segmentu, název formátu je buď název vestavěného formátu, nebo název uživatelsky definovaného formátu.
- Pole *Report* v deskriptoru MQMD se segmenty s hodnotou *Offset* větší než nula je upraveno. Pro každý typ sestavy platí, že pokud je volba sestavy MQRO_ * _WITH_DATA, ale segment nemůže obsahovat žádný z prvních 100 bajtů uživatelských dat (tj. data následující za každou strukturou záhlaví IBM MQ , která může být přítomna), bude volba sestavy změněna na MQRO_ *.

Správce front postupuje podle výše uvedených pravidel, ale jinak rozděluje zprávy nepředvídatelně; neprovádět hypotézy o tom, kde je zpráva rozdělena.

V případě *trvalých* zpráv může správce front provádět segmentaci pouze v rámci pracovní jednotky:

- Je-li volání MQPUT nebo MQPUT1 funkční v rámci uživatelské jednotky práce, použije se jednotka práce. Pokud během procesu segmentace selže volání, odebere správce front všechny segmenty, které byly umístěny do fronty, jako výsledek selhání volání. Selhání však nezabrání úspěšnému potvrzení jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelsky definovaná jednotka práce, správce front vytvoří pracovní jednotku pouze po dobu trvání hovoru. Je-li volání úspěšné, správce front potvrdí jednotku práce automaticky. Pokud se volání nezdaří, správce front provede zálohu jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce, ale uživatelem definovaná jednotka práce existuje, správce front nemůže provést segmentaci. Pokud zpráva nevyžaduje segmentaci, může být volání přesto 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, aby bylo možné provést segmentaci.

Při převodu dat ve zprávách, které mohou být segmentovány, je zapotřebí zvláštní opatrnosti:

- Pokud přijímající aplikace převádí data na volání MQGET a určuje volbu MQGMO_COMPLETE_MSG, předání řízení dat bude předáno úplnou zprávu pro příslušnou uživatelskou proceduru a skutečnost, že byla zpráva segmentována, je zřejmá pro ukončení.
- Pokud přijímající aplikace načte v daném okamžiku jeden segment, vyvolá se uživatelská procedura konverze dat k převodu jednoho segmentu v daném okamžiku. Výjezd musí tedy převést data v segmentu nezávisle na datech v některém z ostatních segmentů.

  Je-li povaha dat ve zprávě taková, že libovolná segmentace dat na šestnáctibajtových okrajích může vést k segmentům, které nemohou být převedeny uživatelskou procedurou, nebo že formát je MQFMT_STRING a znaková sada je DBCS nebo smíšená SBCS/DBCS, odesílající aplikace musí vytvořit a vložit segmenty, přičemž hodnota MQMF_SEGMENTATION_INHIBITED potlačuje další segmentaci. Tímto způsobem odesílající aplikace může zajistit, aby každý segment obsahoval dostatečné informace pro umožnění úspěšného převodu segmentu na výstupu konverze dat.
- Je-li pro odesílajícího agenta MCA (Message Channel Agent) určena konverze odesílatele, program MCA převádí pouze zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí o převod zpráv, které jsou segmenty.

Tento příznak je vstupní příznak volání MQPUT a MQPUT1 a výstupní příznak pro volání MQGET. Při druhém volání správce front také zobrazí hodnotu příznaku pro pole *Segmentation* v produktu MQGMO.

Počáteční hodnota tohoto příznaku je MQMF_SEGMENTATION_INHIBITED.

Příznaky stavu: Jedná se o příznaky, které označují, zda fyzická zpráva patří do skupiny zpráv, je segment logické zprávy, obojí, nebo ani jedno. Na volání MQPUT nebo MQPUT1 může být určena jedna nebo více z následujících možností, nebo je vrácena pomocí volání MQGET:

MQMF_MSG_IN_GROUP
Zpráva je členem skupiny.

MQM_LAST_MSG_IN_GROUP
Zpráva je poslední logickou zprávou ve skupině.

Je-li tento příznak nastaven, správce front zapíná MQMF_MSG_IN_GROUP v kopii MQMD, který je odeslán se zprávou, ale nemění nastavení těchto parametrů v deskriptoru MQMD, které je poskytováno aplikací v rámci volání MQPUT nebo MQPUT1 .

Je platný pro skupinu, která se má skládat pouze z jedné logické zprávy. Pokud se jedná o tento případ, je nastavena hodnota MQMF_LAST_MSG_IN_GROUP, ale pole *MsgSeqNumber* má hodnotu jedna.

SEGMENT MQMF_SEGMENT
Zpráva je segmentem logické zprávy.

Když je MQMF_SEGMENT zadán bez MQMF_LAST_SEGMENT, musí být délka dat zprávy aplikace v segmentu (*kromě* délek všech 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, je-li zpráva vložena do fronty s typem indexu MQIT_GROUP_ID.

MQMF_LAST_SEGMENT
Zpráva je posledním segmentem logické zprávy.

Je-li tento příznak nastaven, správce front zapíná MQMF_SEGMENT v kopii MQMD, který je odeslán se zprávou, ale nemění nastavení těchto parametrů v deskriptoru MQMD, který je poskytován aplikací v rámci volání MQPUT nebo MQPUT1 .

Logická zpráva se může skládat pouze z jednoho segmentu. Je-li tomu tak, je nastavena hodnota MQMF_LAST_SEGMENT, ale pole *Offset* má hodnotu nula.

Je-li zadáno MQMF_LAST_SEGMENT, může být délka dat zprávy aplikace v segmentu (*kromě* délek všech struktur záhlaví, které mohou být přítomny), nula.

V systému z/OSnení tato volba podporována, je-li zpráva vložena do fronty s typem indexu MQIT_GROUP_ID.

Aplikace musí zajistit správné nastavení těchto parametrů při vkládání zpráv. Je-li zadán parametr MQPMO_LOGICAL_ORDER nebo byl zadán v předchozím volání MQPUT pro manipulátor fronty, musí být nastavení příznaků konzistentní s informacemi o skupině a segmentu zachované správcem front pro manipulátor fronty. Následující podmínky se vztahují na *po sobě jdoucí* volání MQPUT pro manipulátor fronty, je-li zadáno MQPMO_LOGICAL_ORDER:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, všechny tyto příznaky (a jejich kombinace) jsou platné.
- Jakmile je zadán parametr MQMF_MSG_IN_GROUP, musí zůstat zapnutý, dokud není zadána hodnota MQMF_LAST_MSG_IN_GROUP. Volání se nezdaří s kódem příčiny MQRC_INCOMPLETE_GROUP, pokud tato podmínka není splněna.
- Jakmile je zadán parametr MQMF_SEGMENT, musí zůstat zapnutý, dokud není zadán parametr MQMF_LAST_SEGMENT. Volání se nezdaří s kódem příčiny MQRC_INCOMPLETE_MSG, pokud tato podmínka není splněna.
- Po zadání hodnoty MQMF_SEGMENT bez MQMF_MSG_IN_GROUP musí hodnota MQMF_MSG_IN_GROUP zůstat *off* , dokud není zadán parametr MQMF_LAST_SEGMENT. Volání se nezdaří s kódem příčiny MQRC_INCOMPLETE_MSG, pokud tato podmínka není splněna.

Fyzické pořadí ve frontě uvádí platné kombinace příznaků a hodnoty použité pro různá pole.

Tyto příznaky jsou vstupní příznaky na volání MQPUT a MQPUT1 a výstupní příznaky na volání MQGET. Při druhém volání správce front také odráží hodnoty parametrů pro pole *GroupStatus* a *SegmentStatus* v MQGMO.

Seskupené nebo segmentované zprávy nemůžete používat s publikováním/odběrem.

Výchozí příznaky: Uvedou se následující informace, které označují, že zpráva má výchozí atributy:

MQMF_NONE
Žádné příznaky zpráv (výchozí atributy zpráv).

To inhibuje segmentaci a označuje, že zpráva není ve skupině a není segmentem logické zprávy. Funkce MQMF_NONE je definována pro dokumentaci programu podpory. Není určeno, aby tento parametr byl použit spolu s jiným, ale jako jeho hodnota je nula, takové použití nelze detekovat.

Pole *MsgFlags* je rozděleno na dílčí pole, kde jsou podrobnosti viz “Volby sestav a příznaky zpráv” na stránce 893.

Počáteční hodnota tohoto pole je MQMF_NONE. Toto pole je ignorováno, pokud *Version* je menší než MQMD_VERSION_2.

OriginalLength (MQLONG)

Toto pole je relevantní pouze pro zprávy hlášení, které jsou segmenty. Určuje délku segmentu zprávy, k němuž se zpráva sestavy vztahuje; neudává délku logické zprávy, jejíž část tvoří část formuláře, nebo délku dat ve zprávě sestavy.

Poznámka: Při generování zprávy sestavy pro zprávu, která je segmentem, se kopie správce front a agent kanálu zpráv do MQMD pro zprávu hlásí do polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags*, v polích z původní zprávy. V důsledku toho je zpráva zprávy také segmentem. Aplikace, které generují zprávy sestav, musí provádět stejné nastavení 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:

- Je-li odesílaná zpráva segmentem a je také zprávou sestavy, přijme správce front zadanou hodnotu. Hodnota musí být:
  - Větší než nula, pokud segment není posledním segmentem
  - Ne méně než nula, je-li 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.
- Je-li odesílaná zpráva segment, ale ne zpráva sestavy, správce front ignoruje pole a použije místo toho délku dat zprávy aplikace.
- Ve všech ostatních případech správce front toto pole ignoruje a místo toho použije hodnotu MQOL_UNDEFINED.

Jedná se o výstupní pole ve volání MQGET.

Počáteční hodnota tohoto pole je MQOL_UNDEFINED. Toto pole je ignorováno, pokud *Version* je 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 504 na stránce 470.

Pokud aplikace poskytuje předpony version-2 MQMD *a* před data zprávy aplikace s MQMDE, struktury se zpracují, jak ukazuje následující tabulka.

Tabulka 503. 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 503 na stránce 469).
- 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 504. 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
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

Tabulka 504. Pole v MQMDE pro MQMDE (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.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)

Hodnota musí být:

MQM_STRUCTION_ID
Identifikátor pro strukturu rozšíření deskriptoru zpráv.

Pro programovací jazyk C je také definována konstanta MQMDE_STRUC_ID_ARRAY; má stejnou hodnotu jako MQMDE_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQMDE_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQMDE_VERSION_2
Struktura rozšíření deskriptoru zpráv Version-2 .

Následující konstanta uvádí číslo verze aktuální verze:

MQM_AKTUÁLNÍ_VERZE
Aktuální verze struktury rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MQMDE_VERSION_2.

StrucLength (MQLONG)

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)

Uvádí číselné kódování dat, která se řídí strukturou MQMDE; nevztahuje se na číselná data ve struktuře MQMDE.

Na základě 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 najdete v poli *Encoding* , které popisuje téma “MQMD-Deskriptor zpráv” na stránce 419 .

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG)

Uvádí identifikátor znakové sady dat, která se řídí strukturou MQMDE; nevztahuje se na znaková data v samotné struktuře MQMDE.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Je možné použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru 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)

Uvádí název formátu dat, která se řídí strukturou MQMDE.

Na základě 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 419.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

Lze zadat následující příznak:

MQMDEF_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQMDEF_NONE.

GroupId (MQBYTE24)

Viz pole *GroupId* popsané v části"MQMD-Deskriptor zpráv" na stránce 419. Počáteční hodnota tohoto pole je MQGI_NONE.

Počet MsgSeqNumber (MQLONG)

Viz pole *MsgSeqNumber* popsané v části"MQMD-Deskriptor zpráv" na stránce 419. Počáteční hodnota tohoto pole je 1.

Offset (MQLONG)

Viz pole *Offset* popsané v části"MQMD-Deskriptor zpráv" na stránce 419. Počáteční hodnota tohoto pole je 0.

MsgFlags (MQLONG)

Viz pole *MsgFlags* popsané v části "MQMD-Deskriptor zpráv" na stránce 419. Počáteční hodnota tohoto pole je MQMF_NONE.

OriginalLength (MQLONG)

Viz pole *OriginalLength* popsané v části"MQMD-Deskriptor zpráv" na stránce 419. 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 505. 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)

Struktura popisovače zpráv pro strukturu voleb vyrovnávací paměti-pole StrucId

Jedná se o identifikátor struktury. Hodnota musí být:

MQMHBO_STRUCTION_ID
Identifikátor pro popisovač zprávy pro strukturu voleb vyrovnávací paměti.

Pro programovací jazyk C je také definována konstanta MQMHBO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQMHBO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMHBO_STRUC_ID.

Verze (MQLONG)

Struktura popisovače zpráv pro strukturu voleb vyrovnávací paměti-pole Verze

Jedná se o číslo verze struktury. Hodnota musí být:

MQMHBO_VERSION_1
Číslo verze pro popisovač zprávy pro strukturu voleb vyrovnávací paměti.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQMHBO_CURRENT_VERSION
Aktuální verze obslužné rutiny zpráv pro strukturu voleb vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQMHBO_VERSION_1.

Volby (MQLONG)

Struktura popisovače zpráv pro strukturu voleb vyrovnávací paměti-pole Volby

Tyto volby řídí akci MQMHBUF.

Je třeba určit následující volbu:

MQMHBO_PROPERTIES_IN_MQRFH2
Při převádění vlastností z manipulátorů zpráv do vyrovnávací paměti je převeďte do formátu MQRFH2 .

Volitelně můžete také zadat následující volbu. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

VLASTNOSTI MQMHBO_DELETE_PROPERTIES
Vlastnosti, které jsou přidány do vyrovnávací paměti, se odstraní z popisovače zprávy. Pokud se nezdaří volání, 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
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

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
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

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)

Jedná se o identifikátor struktury; hodnota musí být:

MQOD_STRUCTION_ID
Identifikátor struktury deskriptoru objektu.

Pro programovací jazyk C je také definována konstanta MQOD_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQOD_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQOD_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být jedna z následujících:

MQOD_VERSION_1
Struktura deskriptoru objektu Version-1 .

MQOD_VERSION_2
Struktura deskriptoru objektu Version-2 .

MQOD_VERSION_3
Struktura deskriptoru objektu Version-3 .

MQOD_VERSION_4
Struktura deskriptoru objektu Version-4 .

Všechny verze jsou podporovány ve všech prostředích produktu IBM MQ V7.0 .

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

VERZE AKTUÁLNÍ_VERZE
Aktuální verze struktury deskriptoru objektu.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQOD_VERSION_1.

ObjectType (MQLONG)

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)

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.

Název ObjectQMgr(MQCHAR48)

Jedná se o název správce front, ve kterém je definován objekt *ObjectName* . Znaky, které jsou platné v názvu, jsou stejné jako ty, které jsou platné pro *ObjectName* (viz “ObjectName (MQCHAR48)” na stránce 483). Název, který je zcela prázdný až k prvnímu znaku null nebo konec pole označuje správce front, ke kterému je aplikace připojena (lokální správce front).

Pro typy označených objektů platí následující body:

- Pokud *ObjectType* je MQOT_TOPIC, MQOT_NAMELIST, MQOT_PROCESS nebo MQOT_Q_MGR, *ObjectQMgrName* musí být prázdný nebo název lokálního správce front.
- Je-li *ObjectName* názvem modelové fronty, správce front vytvoří dynamickou frontu s atributy modelové fronty a vrátí se do pole *ObjectQMgrName* název správce front, ve kterém je fronta vytvořena; toto je název lokálního správce front. Modelovou frontu lze zadat pouze v rámci volání MQOPEN. Modelová fronta není na volání MQPUT1 platná.
- Je-li *ObjectName* název fronty klastru a *ObjectQMgrName* je prázdný, místo určení zpráv odeslaných pomocí manipulátoru fronty vráceného voláním MQOPEN je zvoleno správcem front (nebo uživatelskou procedurou pracovní zátěže klastru, pokud je instalována), jak je uvedeno níže:
  - Je-li zadána hodnota MQOO_BIND_ON_OPEN, správce front při zpracování volání MQOPEN vybere konkrétní instanci fronty klastru a všechny zprávy odeslané s použitím tohoto popisovače fronty budou odeslány do této instance.
  - Je-li zadána hodnota MQOO_BIND_NOT_FIXED, může správce front zvolit jinou instanci cílové fronty (umístěné v jiném správci front v klastru) pro každé následující volání MQPUT, které používá tento popisovač fronty.

  Pokud aplikace potřebuje odeslat zprávu do *specifické* instance fronty klastru (tj. instance fronty, která se nachází na konkrétním správci front v klastru), musí aplikace určit název správce front v poli *ObjectQMgrName* . Tím se lokální správce front odešle k odeslání zprávy do určeného cílového správce front.
- Je-li *ObjectName* název sdílené fronty, která jsou vlastněny skupinou sdílení front, do níž patří lokální správce front, *ObjectQMgrName* může být název skupiny sdílení front, název lokálního správce front nebo prázdný; zpráva se umístí do stejné fronty, podle toho, která z těchto hodnot je uvedena.

  Skupiny sdílení front jsou podporovány pouze v systému z/OS.
- Je-li *ObjectName* název sdílené fronty, která je vlastněna vzdálenou skupinou sdílení front (tj. skupina sdílení front, do níž lokální správce front nenáleží), musí být hodnotou *ObjectQMgrName* název skupiny sdílení front. Můžete použít název správce front, který patří do této skupiny, ale to může zpozdit zprávu v případě, že tento konkrétní správce front není k dispozici, když zpráva dorazí do skupiny sdílení front.
- Je-li otevíraný objekt rozdělovník (to znamená, že *RecsPresent* je větší než nula), *ObjectQMgrName* musí být prázdný nebo řetězec s hodnotou null. 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, je-li *ObjectName* název modelové fronty, a vstupní pole pouze ve všech ostatních případech. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

DynamicQName (MQCHAR48)

Jedná se o název dynamické fronty, která má být vytvořena voláním MQOPEN. To má význam pouze v případě, že *ObjectName* uvádí název modelové fronty; ve všech ostatních případech je *DynamicQName* ignorován.

Znaky, které jsou platné v názvu, jsou stejné jako znaky pro *ObjectName*, až na to, že hvězdička je také platná. Název, který je prázdný (nebo jeden z mezer, který se vyskytuje pouze před prvním znakem null) není platný, pokud *ObjectName* je název modelové fronty.

Je-li posledním nemezerovaným znakem v názvu hvězdička (*), nahradí správce front hvězdičku řetězcem znaků, který zaručuje, že název generovaný pro danou frontu je jedinečný v lokálním správci

front. Pro povolení dostatečného počtu znaků je hvězdička platná pouze v pozicích 1 až 33. Po hvězdičce nesmí být žádné jiné znaky než mezery nebo znak null.

Je platný pro hvězdičku, aby se vyskytla v první znakové pozici. V takovém případě se jméno skládá pouze ze znaků generovaných správcem front.

V systému z/OSnepoužívejte v první znakové pozici název s hvězdičkou, protože ve frontě nejsou prováděny žádné kontroly zabezpečení s úplným názvem, který je generován automaticky.

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 bez mezer v jiných programovacích jazycích.

ID AlternateUserID (MQCHAR12)

Zadáte-li MQOTE_ALTERNATE_USER_AUTHORITY pro volání MQOPEN nebo MQPMO_ALTERNATE_USER_AUTHORITY pro volání MQPUT1 , bude toto pole obsahovat alternativní identifikátor uživatele, který se používá ke kontrole oprávnění pro otevřené místo identifikátoru uživatele, pod kterým momentálně běží aplikace. Některé kontroly se však i nadále provádějí s aktuálním identifikátorem uživatele (například kontroly kontextu).

Je-li zadáno MQOO_ALTERNATE_USER_AUTHORITY nebo MQPMO_ALTERNATE_USER_AUTHORITY a toto pole je zcela prázdné až na první znak null nebo na konci pole, může být otevření úspěšné pouze v případě, že není k otevření tohoto objektu s použitím uvedených voleb potřebná žádná autorizace uživatele.

Není-li zadán parametr MQOO_ALTERNATE_USER_AUTHORITY ani MQPMO_ALTERNATE_USER_AUTHORITY, bude toto pole ignorováno.

V označeném prostředí existují následující rozdíly:

- V systému z/OSse ke kontrole autorizace pro otevření používají pouze prvních 8 znaků produktu *AlternateUserId* . Avšak, aktuální identifikátor uživatele musí být autorizován k uvedení tohoto konkrétního alternativního identifikátoru uživatele; pro tuto kontrolu se použijí všech 12 znaků alternativního identifikátoru uživatele. Identifikátor uživatele musí obsahovat pouze znaky povolené externím správcem zabezpečení.

  Je-li pro frontu zadán parametr *AlternateUserId* , může správce front při vkládání zpráv následně použít hodnotu. Pokud volby MQPMO_ * _CONTEXT zadané v volání MQPUT nebo MQPUT1 způsobí, že správce front vygeneruje informace o kontextu identity, umístí správce front *AlternateUserId* do pole *UserIdentifier* v záhlaví MQMD příslušné zprávy místo aktuálního identifikátoru uživatele.
- V jiných prostředích se produkt *AlternateUserId* používá pouze pro kontroly řízení přístupu k otevířenému objektu. Je-li objektem fronta, *AlternateUserId* neovlivňuje obsah pole *UserIdentifier* v 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 řetězec s hodnotou null v C a 12 prázdných znaků v jiných programovacích jazycích.

RecsPresent (MQLONG)

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 jedno místo určení.

The value of *RecsPresent* must not be less than zero, and if it is greater than zero *ObjectType* must be MQOT_Q; the call fails with reason code MQRC_RECS_PRESENT_ERROR if these conditions are not satisfied.

V systému z/OSmusí být toto pole nula.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_2.

Počet KnownDestPočet (MQLONG)

Jedná se o počet front v seznamu distribucí, které se převáděly na lokální fronty a které byly úspěšně otevřeny. Tento počet nezahrnuje fronty, které se interpretují do vzdálených front (ačkoli lokální přenosová fronta je na počátku použita k uložení zprávy). Je-li tento parametr přítomen, je toto pole také nastaveno při otevření jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_1.

Počet UnknownDest(MQLONG)

Jedná se o počet front v seznamu distribucí, které se interpretují do vzdálených front a které byly úspěšně otevřeny. Je-li tento parametr přítomen, je toto pole také nastaveno při otevření jedné fronty, která není v rozdělovníku.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_1.

Počet InvalidDestPočet (MQLONG)

Jedná se o počet front v rozdělovníku, které se nepodařilo úspěšně otevřít. Je-li tento parametr přítomen, je toto pole také nastaveno při otevření jedné fronty, která není v rozdělovníku.

Poznámka: Je-li toto pole uvedeno, je nastaveno pouze v případě, že je parametr **CompCode** u volání MQOPEN nebo MQPUT1 MQCC_OK nebo MQCC_WARNING; není nastaven, pokud je parametrem **CompCode** hodnota MQCC_FAILED.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_1.

Posunutí ObjectRec(MQLONG)

Jedná se o posun v bajtech prvního záznamu objektu MQOR od začátku struktury MQOD. Odsazení může být kladné nebo záporné. *ObjectRecOffset* se používá pouze tehdy, když je otevíraný distribuční seznam. Pole je ignorováno, pokud *RecsPresent* je nula.

Když se otevírá distribuční seznam, musí být poskytnuto pole jednoho nebo více záznamů objektů MQOR, aby bylo možné určit názvy cílových front v rozdělovníku. To lze provést jedním ze dvou způsobů:

- Použitím pole offsetu *ObjectRecOffset*.

  V takovém případě aplikace musí deklarovat vlastní strukturu obsahující MQOD následovaný polem záznamů MQOR (s tolika prvky pole jako jsou potřeba) a nastavit proměnnou *ObjectRecOffset* na posun prvního prvku v poli od začátku operace MQOD. Ujistěte se, že je tento posun správný a má hodnotu, která může být umístěna v rámci MQLONG (nejvíce restriktivní programovací jazyk je COBOL, pro který je platný rozsah -999 999 999 až +999 999 999).

  Použijte *ObjectRecOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele takovým způsobem, který není přenosný do různých prostředí (například programovací jazyk COBOL).
- Použitím pole ukazatele *ObjectRecPtr*.

  V takovém případě může aplikace deklarovat pole struktury MQOR odděleně od struktury MQOD a nastavit *ObjectRecPtr* na adresu pole.

  Použijte *ObjectRecPtr* pro programovací jazyky, které podporují datový typ ukazatele, a to způsobem, který je přenosný do různých prostředí (například programovací jazyk C).

Zvolením jakékoliv techniky použijte jeden z produktů *ObjectRecOffset* a *ObjectRecPtr* . volání selže s kódem příčiny MQRC_OBJECT_RECORDS_ERROR, pokud jsou obě hodnoty nula nebo obě jsou nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_2.

Posunutí ResponseRec(MQLONG)

Jedná se o posun v bajtech prvního záznamu odezvy MQRR od začátku struktury MQOD. Odsazení může být kladné nebo záporné. *ResponseRecOffset* se používá pouze tehdy, když je otevíraný distribuční seznam. Pole je ignorováno, pokud *RecsPresent* je nula.

Je-li otevřen distribuční seznam, můžete zadat pole jednoho nebo více záznamů odpovědí MQRR, aby bylo možné identifikovat fronty, které se nepodařilo otevřít (pole*CompCode* v MQRR), a důvod pro každé selhání (pole*Reason* v MQRR). Data se vrátí v poli záznamů odpovědí ve stejném pořadí, v jakém se vyskytují názvy front 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 se nezdařily, nebo všechny selhaly, ale z různých důvodů); kód příčiny MQRC_MULTIPLE_REASONS z volání označuje tento případ. Pokud se stejný kód příčiny vztahuje na všechny fronty, je tento důvod vrácen v parametru **Reason** volání MQOPEN nebo MQPUT1 a záznamy odezvy nejsou nastaveny. Záznamy odezvy jsou volitelné, ale pokud jsou dodány, musí být *RecsPresent* z nich.

Záznamy odezvy lze poskytovat stejným způsobem jako záznamy objektů, a to buď uvedením offsetu v *ResponseRecOffset*, nebo zadáním adresy v *ResponseRecPtr* ; Podrobnosti o tom, jak to provést, viz “Posunutí ObjectRec(MQLONG)” na stránce 486. Avšak, nelze použít více než jeden z *ResponseRecOffset* a *ResponseRecPtr* ; volání selže s kódem příčiny MQRC_RESPONSE_RECORS_ERROR, pokud jsou oba nenulové.

Pro volání MQPUT1 jsou tyto záznamy odpovědí použity k vrácení informací o chybách, které se vyskytnou při odeslání zprávy do front v seznamu distribucí, a také o chybách, které se vyskytnou při otevření front. Kód dokončení a kód příčiny z operace put pro frontu nahrazují kód dokončení operací z otevřené operace pro tuto frontu pouze v případě, že kód dokončení z této fronty byl MQCC_OK nebo MQCC_WARNING.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_2.

ObjectRecPtr (MQPTR)

Jedná se o adresu prvního záznamu objektu MQOR. *ObjectRecPtr* se používá pouze tehdy, když je otevíraný distribuční seznam. Pole je ignorováno, pokud *RecsPresent* je nula.

Můžete použít buď *ObjectRecPtr* nebo *ObjectRecOffset* , abyste uvedli záznamy objektů, ale ne obojí; pro popis pole *ObjectRecOffset* , viz “Posunutí ObjectRec(MQLONG)” na stránce 486. Pokud nepoužíváte *ObjectRecPtr*, nastavte ji na nulový ukazatel nebo na null bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je 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 hodnotou je řetězec bajtů se všemi znaky null.

ResponseRecPtr (MQPTR)

Jedná se o adresu prvního záznamu odezvy MQRR. *ResponseRecPtr* se používá pouze tehdy, když je otevíraný distribuční seznam. Pole je ignorováno, pokud *RecsPresent* je nula.

Chcete-li určit záznamy odpovědí, ale ne obojí, použijte buď *ResponseRecPtr* nebo *ResponseRecOffset* ; podrobnosti viz “Posunutí ResponseRec(MQLONG)” na stránce 487. Pokud nepoužíváte *ResponseRecPtr*, nastavte ji na nulový ukazatel nebo na null bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je 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 hodnotou je řetězec bajtů se všemi znaky null.

AlternateSecurityId (MQBYTE40)

Jedná se o identifikátor zabezpečení předávaný s produktem *AlternateUserId* autorizační služby, aby bylo možné provést odpovídající kontroly autorizace. *AlternateSecurityId* se používá pouze tehdy, pokud:

- Funkce MQOO_ALTERNATE_USER_AUTHORITY je zadána v rámci volání MQOPEN nebo
- Funkce MQPMO_ALTERNATE_USER_AUTHORITY je zadána v rámci volání MQPUT1 .

a pole *AlternateUserId* není zcela prázdné až na první znak null nebo na konec pole.

V systému Windowslze produkt *AlternateSecurityId* použít k zadání identifikátoru zabezpečení (SID) produktu Windows , který jednoznačně identifikuje produkt *AlternateUserId*. Identifikátor SID pro uživatele lze získat ze systému Windows pomocí volání rozhraní API LookupAccountName ( ) Windows .

V systému z/OSje toto pole ignorováno.

Pole *AlternateSecurityId* má následující strukturu:

- První bajt je binární celé číslo obsahující dlouhá data, která následují; hodnota vylučuje samotný bajt. Není-li uveden žádný identifikátor zabezpečení, je délka nula.
- Druhý bajt označuje typ identifikátoru zabezpečení, který je přítomný; jsou možné následující hodnoty:

  ID_BEZPEČNOSTNÍHO_ZABEZPEČENÍ MQSID_NT_ID_
  Identifikátor zabezpečení produktu Windows .

  MQSIDT_NONE
  Žádný identifikátor zabezpečení.
- Třetí a následující bajty až do délky definované prvním bytem obsahují vlastní identifikátor zabezpečení.
- Zbývající bajty v poli jsou nastaveny na binární nulu.

Můžete použít následující speciální hodnotu:

MQSID_NONE
Není uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQSID_NONE_ARRAY; hodnota 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 *Version* je menší než MQOD_VERSION_3.

ResolvedQName (MQCHAR48)

Jedná se o název cílové fronty poté, co název lokálního správce front interpretuje název. Vrácený název je název fronty, která existuje ve správci front identifikovaném příkazem *ResolvedQMgrName*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je otevřena jediná fronta pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Je-li otevřený objekt jakýkoli z následujících, *ResolvedQName* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřena pro procházení, vstup nebo výstup
- Distribuční seznam
- Fronta aliasů, která odkazuje na objekt tématu (místo toho se odkazuje na ResObjectString).
- Fronta aliasů, 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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_3.

Název ResolvedQMgr(MQCHAR48)

Jedná se o název cílového správce front poté, co lokální správce front vyřeší daný název. Vrácený název je název správce front, který vlastní frontu určenou produktem *ResolvedQName*. *ResolvedQMgrName* může být název lokálního správce front.

Pokud *ResolvedQName* je sdílená fronta, kterou vlastní skupina sdílení front, do níž patří lokální správce front, *ResolvedQMgrName* je název skupiny sdílení front. Pokud je fronta vlastníkem některé jiné skupiny sdílení front, může být produktem *ResolvedQName* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (charakter vráceného výsledku je určen definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je otevřena jediná fronta pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Je-li otevřený objekt jakýkoli z následujících, *ResolvedQMgrName* je nastaven na mezery:

- Nejedná se o frontu
- Fronta, ale neotevřena pro procházení, vstup nebo výstup
- Fronta klastru s uvedeným parametrem MQOO_BIND_NOT_FIXED (nebo s MQOO_BIND_AS_Q_DEF, pokud 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 řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_3.

ObjectString (MQCHARV)

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)

Toto je řetězec používaný k poskytnutí kritérií výběru použitých při načítání zpráv z fronty.

Parametr *SelectionString* nesmí být zadán v následujících případech:

- Pokud *ObjectType* není MQOT_Q
- Není-li otevíraná fronta otevřena pomocí jedné z voleb MQOO_BROWSE nebo MQOO_INPUT_ *

Je-li v těchto případech zadán příkaz *SelectionString* , volání selže s kódem příčiny MQRC_SELECTOR_INVALID_FOR_TYPE.

Pokud je parametr *SelectionString* zadán nesprávně, v souladu s popisem způsobu použití struktury “MQCHARV-Řetězec délky proměnné” na stránce 294 , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SELECTION_STRING_ERROR. Maximální délka *SelectionString* je MQ_SELECTOR_LENGTH.

Použití produktu *SelectionString* je popsáno v tématu Selektory.

Řetězec ResObject(MQCHARV)
Pole Řetězec ResObjectje dlouhé jméno objektu poté, co správce front vyřeší název zadaný v poli *ObjectName* .

Toto pole je vráceno pouze pro témata a aliasy front, které odkazují na objekt tématu.

Pokud je v produktu *ObjectString* zadán dlouhý název objektu a v produktu *ObjectName*není k dispozici nic, vrátí se hodnota vrácená v tomto poli stejná jako hodnota uvedená v části *ObjectString*.

Je-li toto pole vynecháno (toto pole je ResObjectString.VSBufSize je nula), pak se *ResObjectString* nevrátí, ale délka bude vrácena v ResObjectString.VSLength.

Je-li délka vyrovnávací paměti (poskytnutá v objektu ResObjectStrng.VSBufSize) kratší než úplná hodnota *ResObjectString*, řetězec bude zkrácen a vrátí se jako počet znaků nejvíce vpravo, kolik se vejde do zadané vyrovnávací paměti.

Pokud je parametr *ResObjectString* zadán nesprávně, v souladu s popisem způsobu použití struktury MQCHARV , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_RES_OBJECT_STRING_ERROR.

ResolvedType (MQLONG)
Typ vyřešeného (základního) objektu, který se otevře.

Možné hodnoty jsou:

MQOT_Q
Vyřešený objekt je fronta. Tato hodnota platí, je-li fronta otevřena přímo nebo když je otevřena fronta aliasů odkazující na frontu.

MQOT_TOPIC
Vyřešený objekt je téma. Tato hodnota platí, je-li téma otevřeno přímo nebo při otevření fronty aliasů ukazujících 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.

Použití

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 506. 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)

To je stejné jako pole *ObjectName* ve struktuře MQOD (podrobnosti viz MQOD), kromě následujících:

- Musí se jednat o název fronty.
- Nesmí se jednat o název modelové fronty.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název ObjectQMgr(MQCHAR48)

To je stejné jako pole *ObjectQMgrName* ve struktuře MQOD (podrobnosti viz MQOD).

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je řetězec s hodnotou null v 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 507. Pole v MQPD

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
StrucId (identifikátor struktury)	ID_STRUC_MQPD_STRUC_ID	'PD'

Tabulka 507. Pole v MQPD (pokračování)

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
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)

Jedná se o identifikátor struktury; hodnota musí být:

ID_STRUKTURY OBJEKTU MQPD_BEAN
Identifikátor pro strukturu deskriptoru vlastností.

Pro programovací jazyk C je také definována konstanta **MQPD_STRUC_ID_ARRAY** ; tato hodnota má stejnou hodnotu jako **MQPD_STRUC_ID**, ale je pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQPD_STRUC_ID**.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQPD_VERSION_1
Struktura deskriptoru vlastností Version-1 .

Následující konstanta uvádí 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)

Hodnota musí být:

MQPD_NONE
Nejsou zadány žádné volby

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQPD_NONE.

Podpora (MQLONG)

Toto pole popisuje, jaká úroveň podpory pro vlastnost zprávy je vyžadována správce front, aby byla zpráva obsahující tuto vlastnost vložena do fronty. Toto platí pouze pro vlastnosti definované IBM MQ; podpora pro všechny ostatní vlastnosti je volitelná.

Pole je automaticky nastaveno na správnou hodnotu, je-li správce front znám vlastnost definovaná uživatelem IBM MQ. Není-li vlastnost rozpoznána, je objekt MQPD_SUPPORT_OPTIONAL přiřazen. When a queue manager receives a message containing an IBM MQ-defined property that the queue manager recognizes as being incorrect, the queue manager corrects the value of the *Support* field.

Při nastavení definované vlastnosti IBM MQpomocí volání MQSETMP na obslužné rutiny zprávy, kde byla nastavena volba MQCMHO_NO_VALIDATION, se *Support* stane vstupním polem. To umožňuje

aplikaci umístit vlastnost definované IBM MQs správnou hodnotou, kde tato vlastnost není podporována připojeným správcem front, ale kde je zpráva určena ke zpracování v jiném správci front.

Hodnota MQPD_SUPPORT_OPTIONAL je vždy přiřazena vlastnostem, které nejsou definovanými vlastnostmi produktu IBM MQ.

Pokud správce front produktu IBM WebSphere MQ 7.0 , který podporuje vlastnosti zprávy, obdrží vlastnost obsahující nerozpoznanou hodnotu *Support* , bude s touto vlastností zacházeno jako s následujícím způsobem:

- Parametr MQPD_SUPPORT_REQUIRED byl zadán, pokud jsou některé z nerozpoznaných hodnot obsaženy v objektu MQPD_REJECT_UNSUP_MASK.
- Parametr MQPD_SUPPORT_REQUIRED_IF_LOCAL byl zadán, pokud jsou některé z nerozpoznaných hodnot obsaženy v proměnné MQPD_ACCEPT_UNSUP_IF_XMIT_MASK
- Příkaz MQPD_SUPPORT_OPTIONAL byl zadán jiným způsobem.

Je vrácena jedna z následujících hodnot volání MQINQMP nebo jedna z hodnot může být zadána při použití volání MQSETMP pro popisovač zprávy, kde je nastavena volba MQCMHO_NO_VALIDATION:

PODPORA MQPD_SUPPORT_OPTIONAL
Vlastnost je přijata správcem front, i když není podporována. Vlastnost může být vyřazena, aby byla zpráva přetékat do správce front, který nepodporuje vlastnosti zprávy. Tato hodnota je také přiřazena vlastnostem, které nejsou IBM MQdefinované.

POŽADOVÁNA PODPORA MQPD_SUPPORT_REQUIRED
Podpora pro vlastnost je povinná. Zpráva je odmítnuta správcem front, který nepodporuje vlastnost definovaná uživatelem IBM MQ. 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 definovaná uživatelem IBM MQ, je-li 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 v rámci volání MQINQMP a vstupní pole pro volání MQSETMP, pokud byl popisovač zprávy vytvořen s použitím volby MQCMHO_NO_VALIDATION. Počáteční hodnota tohoto pole je MQPD_SUPPORT_OPTIONAL.

Kontext (MQLONG)

Tato vlastnost popisuje kontext zprávy, do níž daná vlastnost patří.

When a queue manager receives a message containing an IBM MQ-defined property that the queue manager recognizes as being incorrect, the queue manager corrects the value of the *Context* field.

Je možné zadat následující volbu:

KONTEXT MQPD_USER_CONTEXT
Vlastnost je přidružena ke kontextu uživatele.

K nastavení vlastnosti přidružené k kontextu uživatele pomocí volání MQSETMP není vyžadována žádná speciální autorizace.

Ve správci front produktu IBM WebSphere MQ 7.0 je vlastnost přidružená ke kontextu uživatele uložena, jak je popsáno pro MQOO_SAVE_ALL_CONTEXT. Volání MQPUT s uvedeným parametrem MQPMO_PASS_ALL_CONTEXT způsobí, že se vlastnost zkopíruje z uloženého kontextu do nové zprávy.

Není-li dříve popsaná volba vyžadována, 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 z volání MQINQMP. Počáteční hodnota tohoto pole je MQPD_NO_CONTEXT.

CopyOptions (MQLONG)

Popisuje, do kterého typu zpráv má být vlastnost zkopírována. Toto je výstupní pouze pole pro rozpoznané vlastnosti definované IBM MQ ; IBM MQ nastavuje příslušnou hodnotu.

When a queue manager receives a message containing an IBM MQ defined property that the queue manager recognizes as being incorrect, the queue manager corrects the value of the *CopyOptions* field.

Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát 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 se okopíruje 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 se zkopíruje do všech typů následujících zpráv.

Výchozí volba: Pro dodání výchozí sady voleb kopírování lze zadat následující volbu:

MQCOPY_DEFAULT
Tato vlastnost se okopíruje do zprávy, která se předá, do zprávy sestavy nebo do zprávy přijaté odběratelem při publikování zprávy.

To je ekvivalentní zadání kombinace voleb MQCOPY_FORWARD, plus MQCOPY_REPORT a MQCOPY_PUBLISH.

Pokud žádná z voleb, které jsou popsány dříve, není povinná, použijte následující volbu:

MQCOPY_NONE
Tuto hodnotu použijte, chcete-li označit, že nejsou zadány žádné další volby kopírování; mezi touto vlastností a následujícími zprávami neexistuje žádný vztah. Tato hodnota je vždy vrácena pro vlastnosti deskriptoru zpráv.

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole z 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 508. 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.		
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

Tabulka 508. Pole v MQPMO (pokračování)		
Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
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)

Jedná se o identifikátor struktury; hodnota musí být:

MQPMO_STRUC_ID
Identifikátor struktury voleb put-message.

Pro programovací jazyk C je také definována konstanta MQPMO_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQPMO_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQPMO_STRUC_ID.

Verze (MQLONG)
Číslo verze struktury.

Hodnota musí být jedna z následujících:

MQPMO_VERSION_1
Struktura volby put-message Version-1 .

Tato verze je podporována ve všech prostředích.

MQPMO_VERSION_2
Struktura voleb vložených zpráv Version-2 .

Tato verze je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

a pro IBM MQ MQI clients připojené k těmto systémům.

MQPMO_VERSION_3
Struktura voleb vložených zpráv Version-3 .

Tato verze je podporována ve všech prostředích.

Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQPMO_AKTUÁLNÍ_VERZE
Aktuální verze struktury voleb put-message.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQPMO_VERSION_1.

Volby MQPMO (MQLONG)
Pole Volby řídí činnost volání **MQPUT** a **MQPUT1** .

Volba rozsahu. Můžete zadat libovolné volby MQPMO nebo žádnou z nich. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace, které nejsou platné, jsou zaznamenány; jakékoli jiné kombinace jsou platné.

Následující volba určuje rozsah odeslaných publikací:

MQPMO_SCOPE_QMGR

Publikování se odešle pouze na odběratele, kteří se přihlásili k odběru tohoto správce front. Publikování není předáno žádným vzdáleným správcům front publikování/odběru, kteří provedli odběr u tohoto správce front, který potlačí jakékoli chování nastavené pomocí atributu tématu PUBSCOPE.

Poznámka: Pokud není nastaveno, rozsah publikování je určen atributem tématu PUBSCOPE.

Volby publikování. Následující volby řídí způsob, jakým jsou zprávy publikovány v tématu:

MQPMO_SUPPRESS_REPLYTO

Jakékoli informace zadané v polích *ReplyToQ* a *ReplyToQMgr* deskriptoru MQMD této publikace nejsou předány odběratelům. Je-li tato volba použita s volbou sestavy, která vyžaduje *ReplyToQ*, volání selže s MQRC_MISSING_REPLY_TO_Q.

MQPMO_RETAIN

Odeslaná publikování má být uchována správcem front. Toto uchování umožňuje odběrateli požádat o kopii této publikace po době publikování pomocí volání MQSUBRQ. Umožňuje také odeslání publikování aplikacím, které učiní jejich odběr po datu, kdy byla tato publikace vytvořena (pokud se nerozhodnou neodeslat ji pomocí volby MQSO_NEW_PUBLICATIONS_ONLY). Je-li aplikace odeslána publikování, která byla uchována, je označena vlastností zprávy MQIsRetained této publikace.

V každém uzlu stromu témat může být zachováno pouze jedno publikování. Pokud tedy již existuje zachované publikování pro toto téma publikované kteroukoli jinou aplikací, bude tato publikace nahrazena touto publikací. Je proto lepší vyhnout se tomu, aby zprávy uchovaly více než jeden vydavatel v rámci stejného tématu.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr obsahovat zástupný znak v tématu, v takovém případě se může počet zachovaných publikování shodovat (u různých uzlů ve stromu témat) a některé publikace mohou být odeslány do žádající aplikace. Další podrobnosti naleznete v popisu volání příkazu “MQSUBRQ-Požadavek na odběr” na stránce 788 .

Informace o interakci zachovaných publikování s úrovněmi odběrů naleznete v tématu Zachytávání publikací.

Je-li tato volba použita a publikování nelze zadržet, zpráva se nepublikuje a volání selže s hodnotou MQRC_PUT_NOT_RETAILED.

MQPMO_NOT_OWN_SUBS
Sděluje správci front, že aplikace nemá odesílat žádné z jejích publikací do odběrů, které vlastní. Odběry jsou považovány za vlastněné stejnou aplikací, pokud jsou popisovače připojení stejné.

MQPMO_WARN_IF_NO_SUBSP_MATCHED
Pokud publikování neodpovídá žádnému odběru, vraťte kód dokončení (*CompCode*) MQCC_WARNING a kód příčiny MQRC_NO_SUBS_MATCHED.

Je-li operace vložení vrácena MQRC_NO_SUBS_MATCHED, publikování nebylo doručeno do žádných odběrů. Je-li však v operaci vložení zadána volba MQPMO_RETAIN, bude zpráva uchována a doručena do všech následně definovaných odpovídajících odběrů.

Odběr v tématu se shoduje se zveřejněním, pokud je splněna některá z následujících podmínek:

- Zpráva je doručena do fronty odběru
- Zpráva by byla doručena do fronty odběru, ale problém s frontou znamená, že zpráva nemůže být vložena do fronty, a proto byla umístěna do fronty nedoručených zpráv nebo byla vyřazena.
- Je definován výstupní bod směrování, který potlačí doručení zprávy na odběr.

Odběr v tématu se neshoduje s touto publikací, pokud jsou splněny některé z následujících podmínek:

- Odběr obsahuje řetězec výběru, který se neshoduje s publikováním
- Odběr specifikovaného objektu MQSO_PUBLICATION_ON_REQUEST
- Publikování nebylo doručeno, protože byla v operaci vložení zadána volba MQPMO_NOT_OWN_SUBS a odběr odpovídá identitě vydavatele.

Volby synchronizačního bodu. Následující volby se týkají účasti volání MQPUT nebo MQPUT1 v rámci jednotky práce:

MQPMO_SYNCPOINT
Požadavek má fungovat v rámci běžných protokolů jednotky práce. Zpráva není viditelná mimo pracovní jednotku, dokud se jednotka práce nepotvrdí. Je-li jednotka práce zálohována, zpráva se odstraní.

Není-li zadáno MQPMO_SYNCPOINT a MQPMO_NO_SYNCPOINT, zahrnutí požadavku na vložení do protokolů jednotek práce je určeno prostředím, které spouští správce front, a nikoli prostředím, kde je aplikace spuštěna. V systému z/OSse požadavek na vložení nachází v rámci pracovní jednotky. Ve všech ostatních prostředích se požadavek na vložení nenachází v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nemusí aplikace, kterou chcete nastavit na port, tuto volbu standardně povolit; explicitně zadejte buď MQPMO_SYNCPOINT nebo MQPMO_NO_SYNCPOINT.

Neuvádějte MQPMO_SYNCPOINT k MQPMO_NO_SYNCPOINT.

MQPMO_NE_SYNCPOINT
Požadavek má fungovat mimo běžné protokoly jednotek práce. Zpráva je okamžitě k dispozici a nelze ji odstranit tím, že zazálohujete jednotku práce.

Není-li zadáno MQPMO_NO_SYNCPOINT a MQPMO_SYNCPOINT, zahrnutí požadavku na vložení do protokolů jednotek práce je určeno prostředím, v němž je spuštěn správce front, a nikoli prostředí, v němž je aplikace spuštěna. V systému z/OSse požadavek na vložení nachází v rámci pracovní jednotky. Ve všech ostatních prostředích se požadavek na vložení nenachází v rámci pracovní jednotky.

Vzhledem k těmto rozdílům nemusí aplikace, kterou chcete nastavit na port, tuto volbu standardně povolit; explicitně zadejte buď MQPMO_SYNCPOINT nebo MQPMO_NO_SYNCPOINT.

Neuvádějte MQPMO_NO_SYNCPOINT a MQPMO_SYNCPOINT.

Volby identifikátoru zprávy a korelačního identifikátoru. Následující volby vyžadují, aby správce front vygeneroval nový identifikátor zprávy nebo identifikátor korelace:

MQPMO_NOVÉ_ID_ZPRÁVY
Správce front nahradí obsah pole *MsgId* v produktu MQMD novým identifikátorem zprávy. Tento identifikátor zprávy je odeslán se zprávou a vrácen do aplikace na výstupu z volání MQPUT nebo MQPUT1 .

Volbu MQPMO_NEW_MSG_ID lze zadat také v případě, že je zpráva vložena do distribučního seznamu; podrobnosti naleznete v popisu pole *MsgId* ve struktuře MQPMR.

Použití této volby zmírňuje aplikaci nutnosti resetovat pole *MsgId* na hodnotu MQMI_NONE před každým voláním MQPUT nebo MQPUT1 .

MQPMO_NOVÉ_ KOREL_ID
Správce front nahradí obsah pole *CorrelId* v MQMD novým identifikátorem korelace. Tento korelační identifikátor se odešle se zprávou a vrátí se aplikaci na výstup z volání MQPUT nebo MQPUT1 .

Volbu MQPMO_NEW_CORREL_ID lze zadat také v případě, že je zpráva vložena do distribučního seznamu; podrobnosti naleznete v popisu pole *CorrelId* ve struktuře MQPMR.

Funkce MQPMO_NEW_CORREL_ID je užitečná v situacích, kdy aplikace vyžaduje jedinečný identifikátor korelace.

Volby skupiny a segmentu. Následující volby se vztahují ke zpracování zpráv ve skupinách a segmentech logických zpráv. Přečtěte si níže uvedené definice, které vám pomohou porozumět této volbě.

Upozornění: Segmentované nebo seskupené zprávy nemůžete používat s publikováním/odběrem.

Fyzická zpráva
Jedná se o nejmenší jednotku informací, které lze umístit do fronty nebo z ní odebrat; často odpovídá informacím zadaným nebo načteným při volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zprávy (MQMD). Obecně jsou fyzické zprávy rozlišeny odlišnými hodnotami identifikátoru zprávy (pole*MsgId* v MQMD), ačkoli správce front toto není vynucen.

Logická zpráva
Logická zpráva je jediná jednotka informací o aplikaci pouze pro platformy jiné než z/OS . Pokud nejsou k dispozici omezení systému, je logická zpráva stejná jako fyzická zpráva. Avšak kde jsou

logické zprávy extrémně velké, omezení systému by mohla učinit vhodné nebo nezbytné k rozdělení logické zprávy do dvou nebo více fyzických zpráv, nazývaných *segmenty*.

Logická zpráva, která byla rozdělena na segmenty, se skládá ze dvou nebo více fyzických zpráv, které mají stejný nenulový identifikátor skupiny (pole*GroupId* v MQMD) a stejné pořadové číslo zprávy (pole*MsgSeqNumber* v MQMD). Segmenty jsou odlišeny lišícími hodnotami pro offset segmentu (pole*Offset* v MQMD), který poskytuje odchylku dat ve fyzické zprávě od začátku dat v logické zprávě. Protože každý segment je fyzická zpráva, segmenty v logické zprávě mají obvykle odlišné identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale jejíž segmentace byla povolena odesílající aplikací, má také identifikátor skupiny, který není null, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s identifikátorem skupiny, pokud tato logická zpráva nepatří do žádné skupiny zpráv. Logické zprávy, pro které byla funkce 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 identifikátor skupiny bez hodnoty null. Logické zprávy ve skupině jsou rozlišeny odlišnými hodnotami pro pořadové číslo zprávy, což je celé číslo v rozsahu od 1 do *n*, kde *n* je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentována, ve skupině je více než *n* fyzických zpráv.

MQPMO_LOGICAL_ORDER
Tato volba sděluje správci front, jak aplikace vkládá zprávy do skupin a segmentů logických zpráv. Může být zadán pouze na volání MQPUT; není platný na 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 příkazu MQPMO_LOGICAL_ORDER naleznete v tématu Logické a fyzické řazení .

Volby kontextu. Následující volby řídí zpracování kontextu zprávy:

MQPMOTO_NE_KONTEXT
Kontext identity i původ jsou nastaveny tak, aby neoznačovaly žádný kontext. To znamená, že pole kontextu v MQMD jsou nastavena na:

- Mezery pro znaková pole
- Hodnoty null pro bajtová pole
- Nuly pro číselná pole

MQPMO_VÝCHOZÍ_KONTEXT
Zpráva má mít k sobě přidružené informace o kontextu, pro identitu i pro původ. Správce front nastaví pole kontextu v deskriptoru zpráv následujícím způsobem:

Tabulka 509. Výchozí hodnoty informací o kontextu pro pole MQMD

Pole v MQMD	Použitá hodnota
UserIdentifier	Určeno z prostředí, je-li to možné; jinak nastavte prázdné znaky.
AccountingToken	Je-li to možné, určeno z prostředí; v opačném případě nastavte hodnotu MQACT_NONE.
ApplIdentityData	Nastavit na mezery.

Tabulka 509. Výchozí hodnoty informací o kontextu pro pole MQMD (pokračování)

Pole v MQMD	Použitá hodnota
PutApplType	Určeno z prostředí.
PutApplName	Určeno z prostředí, je-li to možné; jinak nastavte prázdné znaky.
PutDate	Nastavte na datum, kdy je zpráva vložena.
PutTime	Nastavení na čas, kdy je zpráva vložena.
ApplOriginData	Nastavit na mezery.

Další informace o kontextu zprávy viz téma Kontext zprávy.

Jedná se o výchozí hodnoty a akce, pokud nejsou zadány žádné volby kontextu.

KONTEXT MQPMO_PASS_IDENTITY_CONTEXT
Zpráva má k sobě přidružené informace o kontextu. Kontext identity je převzat z manipulátoru fronty uvedeného v poli *Context* . Informace o kontextu výchozího bodu je vygenerováno správcem front stejným způsobem, jakým je pro hodnoty MQPMO_DEFAULT_CONTEXT (viz předchozí tabulka). Další informace o kontextu zprávy viz téma Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_PASS_IDENTITY_CONTEXT (nebo s volbou, která jej označuje). Pro volání MQPUT1 je stejná kontrola autorizace provedena jako volání MQOPEN s volbou MQOO_PASS_IDENTITY_CONTEXT.

MQPMO_PASS_ALL_CONTEXT
Zpráva má k sobě přidružené informace o kontextu. Kontext je převzat z manipulátoru fronty uvedeného v poli *Context* . Další informace o kontextu zprávy naleznete v tématu Řízení informací o kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_PASS_ALL_CONTEXT (nebo s volbou, která jej označuje). Pro volání MQPUT1 bude pro volání MQOPEN s volbou MQOO_PASS_ALL_CONTEXT provedena stejná kontrola autorizace jako pro volání MQOPEN.

KONTEXT MQPMO_SET_IDENTITY_CONTEXT
Zpráva má k sobě přidružené informace o kontextu. Aplikace určuje kontext identity ve struktuře MQMD. Informace o kontextu výchozího bodu je vygenerováno správcem front stejným způsobem, jakým je pro hodnoty MQPMO_DEFAULT_CONTEXT (viz předchozí tabulka). Další informace o kontextu zprávy viz téma Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_SET_IDENTITY_CONTEXT (nebo s volbou, která jej označuje). Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_SET_IDENTITY_CONTEXT.

MQPMO_SET_ALL_CONTEXT
Zpráva má k sobě přidružené informace o kontextu. Aplikace určuje identitu, původ a kontext uživatele ve struktuře MQMD. Další informace o kontextu zprávy viz téma Kontext zprávy.

Pro volání MQPUT musí být fronta otevřena s volbou MQOO_SET_ALL_CONTEXT. Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou MQOO_SET_ALL_CONTEXT.

Můžete určit pouze jednu z voleb kontextu MQPMO_ * _CONTEXT. Pokud nezadáte žádný, předpokládá se hodnota MQPMO_DEFAULT_CONTEXT.

Volby vlastností. Následující volba souvisí s vlastnostmi zprávy:

MQPMOD_MD_FOR_OUTPUT_ONLY
Parametr deskriptoru zpráv musí být použit pouze pro výstup, aby vrátil deskriptor zprávy, který byl vložen. Pole deskriptoru zpráv přidružená k polím *NewMsgHandle*, *OriginalMsgHandle*nebo obě tato pole musí být použita pro vstup do struktury **MQPMO** .

Pokud není poskytnut platný popisovač zprávy, pak se volání nezdaří 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 hodnoty MQPMO_ASYNC_RESPONSE a MQPMO_SYNC_RESPONSE, předpokládá se hodnota MQPMO_RESPONSE_AS_Q_DEF nebo MQPMO_RESPONSE_AS_TOPIC_DEF.

MQPMO_ASYNC_RESPONSE
Volba MQPMO_ASYNC_RESPONSE vyžaduje, aby byla operace MQPUT nebo MQPUT1 dokončena bez čekání aplikace na dokončení volání správcem front. Použití této volby může zlepšit výkon systému zpráv, zejména u aplikací používajících vazby klienta. Aplikace může periodicky kontrolovat pomocí příkazu MQSTAT, zda k chybě došlo během předchozích asynchronních volání.

Při použití této volby budou v produktu MQMD zaručena pouze následující pole:

- ApplIdentityData
- PutApplType
- PutApplName
- ApplOriginData

Navíc, pokud je zadán jeden nebo oba parametry MQPMO_NEW_MSG_ID nebo MQPMO_NEW_CORREL_ID jako volby, jsou dokončeny také vrácené hodnoty MsgId a CorrelId . (MQPMO_NEW_MSG_ID lze implicitně zadat tak, že zadáte prázdné pole MsgId).

Byla dokončena pouze předchozí uvedená pole. Další informace, které by normálně byly vráceny ve struktuře MQMD nebo MQPMO, nejsou definovány.

Při požadavku na asynchronní odezvu vložení pro hodnoty MQPUT1jsou názvy ResolvedQName a ResolvedQMgrvrácené ve struktuře MQOD nedefinované.

Při požadavku na asynchronní odezvu vložení pro volání MQPUT nebo MQPUT1nemusí CompCode a příčina MQCC_OK a MQRC_NONE nezbytně 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 z operací vkládání zkontrolovat kód CompCode a kódy příčiny a také použít příkaz MQSTAT při zadávání dotazů na asynchronní informace o chybě.

Ačkoli se úspěch nebo selhání jednotlivých operací MQPUT nebo MQPUT1 okamžitě nevrátí, může být první chyba, která se vyskytla při asynchronním volání, později určena voláním MQSTAT.

Pokud se nepodaří doručit trvalou zprávu pod synchronizačním bodem s použitím asynchronní odezvy vložení a pokusíte transakci potvrdit, potvrzení se nezdaří a transakce bude vrácena s kódem dokončení MQCC_FAILED a z důvodu MQRC_BACKED_OUT. Aplikace může volání MQSTAT volat k určení příčiny předchozího selhání operace MQPUT nebo MQPUT1 .

MQPMO_SYNC_RESPONSE
Uvedení tohoto typu odezvy vložení zajistí, aby byla operace MQPUT nebo MQPUT1 vždy vydána 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 na objektu fronty nebo tématu.

MQPMO_ODEZVA_NA_DOBA_Q_DEF
Je-li tato hodnota zadána pro volání MQPUT, použije se použitý typ odezvy vložení z hodnoty DEFPRESP zadané ve frontě při jejím prvním otevření aplikací.

- Je-li fronta fronta klastru a tato hodnota je určena pro volání MQPUT, použije se použitý typ odezvy vložení z atributu **DEFPRESP** definovaného ve správci front *destination* , který vlastní konkrétní instanci fronty, na 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 fronta klastru a tato hodnota je určena pro volání MQPUT, použije se použitý typ odezvy vložení z atributu **DEFPRESP** definovaného ve správci front *local* , a to i v případě, že je cílový správce front vzdálený.

Je-li klientská aplikace připojena ke správci front na úrovni dřívější než IBM WebSphere MQ 7.0, chová se tak, jako by byla zadána hodnota MQPMO_SYNC_RESPONSE.

Je-li tato volba zadána pro volání MQPUT1 , hodnota atributu DEFPRESP není známa před tím, než je požadavek odeslán na server. Při výchozím nastavení volání MQPUT1 používá funkci MQPMO_SYNCPOINT pro MQPMO_ASYNC_RESPONSE a v případě použití MQPMO_NO_SYNCPOINT se chová jako pro MQPMO_SYNC_RESPONSE. Toto výchozí chování však můžete potlačit nastavením vlastnosti Put1DefaultAlwaysSync v konfiguračním souboru klienta, viz stanza CHANNELS v konfiguračním souboru klienta.

MQPMO_RESPONSE_AS_TOPIC_DEF
MQPMO_RESPONSE_AS_TOPIC_DEF je synonymem pro MQPMO_RESPONSE_AS_Q_DEF pro použití s objekty témat.

Další volby. Následující volby řídí kontrolu autorizace, co se stane, když správce front přechází do klidového stavu, a řeší názvy front a správců front:

MQPMO_ALTERNATE_USER_AUTHORITY
Funkce MQPMO_ALTERNATE_USER_AUTHORITY udává, že pole *AlternateUserId* v parametru **ObjDesc** volání MQPUT1 obsahuje identifikátor uživatele, který má být použit k ověření oprávnění k vkládání zpráv do fronty. Volání může proběhnout pouze v případě, že je produkt *AlternateUserId* autorizován k otevření fronty s použitím zadaných voleb bez ohledu na to, zda je k tomu oprávnění identifikátoru uživatele, pod kterým je aplikace spuštěna, k tomu oprávněn. (To však neplatí pro zadané volby kontextu, které jsou vždy zkontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.)

Tato volba je platná pouze s voláním MQPUT1 .

UVÁDĚNÍ MQPMO_FAIL_IF_QUIESCING
Tato volba vynutí selhání volání MQPUT nebo MQPUT1 v případě, že je správce front ve stavu uvedení do klidového stavu.

V systému z/OStato volba také vynutí selhání volání MQPUT nebo MQPUT1 , pokud se připojení (pro aplikaci CICS nebo IMS) nachází ve stavu uvedení do klidového stavu.

Volání vrací kód dokončení MQCC_FAILED s kódem příčiny MQRC_Q_MGR_QUIESCING nebo MQRC_CONNECTION_QUIESCING.

MQPMOD_RESOLVE_LOKÁLNÍ_Q
Tato volba se používá k vyplnění *ResolvedQName* ve struktuře MQPMO s názvem lokální fronty, do níž je zpráva vložena, a *ResolvedQMgrName* s názvem lokálního správce front, který je hostitelem lokální fronty. Další informace o funkci MQPMO_RESOLVE_LOCAL_Q naleznete v tématu MQOO_RESOLE_LOCAL_Q.

Máte-li oprávnění k vložení do fronty, máte oprávnění k uvedení tohoto příznaku na volání MQPUT; není třeba žádné speciální oprávnění.

Výchozí volba. Pokud nepotřebujete žádné 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 definován pro dokumentaci programu podpory; není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze zjistit.

MQPMO_NONE je vstupní pole. Počáteční hodnota pole *Options* je MQPMO_NONE.

Časový limit (MQLONG)

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je -1.

Kontext (MQHOBJ)

Je-li zadáno MQPMO_PASS_IDENTITY_CONTEXT nebo MQPMO_PASS_ALL_CONTEXT, musí toto pole obsahovat popisovač vstupní fronty, z níž jsou informace o kontextu přidružené ke zprávě, která má být vložena, přijata.

Není-li zadán parametr MQPMO_PASS_IDENTITY_CONTEXT ani MQPMO_PASS_ALL_CONTEXT, bude toto pole ignorováno.

Toto je vstupní pole. Počáteční hodnota tohoto pole je `0`.

Počet KnownDestPočet (MQLONG)

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v seznamu distribuce, které jsou lokálními frontami. Tento počet nezahrnuje zprávy odeslané do front, které se interpretují do vzdálených front (ačkoli lokální přenosová fronta je na počátku použita 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 *Version* je menší než MQPMO_VERSION_1.

Toto pole není definováno v systému z/OS , protože distribuční seznamy nejsou podporovány.

Počet UnknownDest(MQLONG)

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v rozdělovníku, které se interpretují do vzdálených front. Zprávy, které správce front dočasně uchovává v seznamu položek rozdělovníku jako počet jednotlivých míst určení, 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 *Version* je menší než MQPMO_VERSION_1.

Toto pole není definováno v systému z/OS , protože distribuční seznamy nejsou podporovány.

Počet InvalidDestPočet (MQLONG)

Jedná se o počet zpráv, které nebylo možné odeslat do front v seznamu distribuce. Počet zahrnuje fronty, které se nepodařilo otevřít, a také fronty, které byly úspěšně otevřeny, ale pro které došlo k selhání operace vložení. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Poznámka: Toto pole je nastaveno, pokud je parametr **CompCode** na volání MQPUT nebo MQPUT1 MQCC_OK nebo MQCC_WARNING; může být nastaven, pokud je parametrem **CompCode** MQCC_FAILED, ale nespoléhejte se na tento kód v aplikačním kódu.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud *Version* je menší než MQPMO_VERSION_1.

Toto pole není definováno v systému z/OS , protože distribuční seznamy nejsou podporovány.

ResolvedQName (MQCHAR48)

Jedná se o název cílové fronty po provedení rozpoznání názvu pomocí lokálního správce front. Vrácený název je název fronty, která existuje ve správci front identifikovaném příkazem *ResolvedQMgrName*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objektem distribuční seznam nebo téma, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název ResolvedQMgr(MQCHAR48)

Jedná se o název cílového správce front po provedení rozpoznání názvu pomocí lokálního správce front. Vrácený název je název správce front, který vlastní frontu, kterou identifikuje produkt *ResolvedQName*, a může to být název lokálního správce front.

Pokud *ResolvedQName* je sdílená fronta, kterou vlastní skupina sdílení front, do níž patří lokální správce front, *ResolvedQMgrName* je název skupiny sdílení front. Pokud je fronta vlastníkem některé jiné skupiny sdílení front, může být produktem *ResolvedQName* název skupiny sdílení front nebo název správce

front, který je členem skupiny sdílení front (charakter vráceného výsledku je určen definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objektem distribuční seznam nebo téma, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

RecsPresent (MQLONG)

Jedná se o počet vložených záznamů zpráv MQPMR nebo záznamů odpovědí MQRR, které byly poskytnuty aplikací. Toto číslo může být větší než nula pouze v případě, že je zpráva vložena do distribučního seznamu. Záznamy zpráv a záznamy odpovědí jsou volitelné; aplikace nemusí poskytovat žádné záznamy, nebo může poskytnout záznamy pouze jednoho typu. Avšak, pokud aplikace poskytuje záznamy obou typů, musí poskytnout záznamy *RecsPresent* každého typu.

Hodnota *RecsPresent* nemusí být stejná jako počet míst určení v rozdělovníku. Je-li zadáno příliš mnoho záznamů, přebytečné nejsou použity; je-li uvedeno příliš málo záznamů, použijí se výchozí hodnoty pro vlastnosti zprávy pro ty cíle, které nevloží záznamy zpráv (viz *PutMsgRecOffset*).

Je-li *RecsPresent* menší než nula nebo je větší než nula, ale zpráva se nedistribuuje na distribuční seznam, volání selže s kódem příčiny MQRC_REC_PRESENT_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQPMO_VERSION_2.

PutMsgRecFields (MQLONG)

Toto pole obsahuje příznaky, které označují, která pole MQPMR se nacházejí v záznamech vložených zpráv poskytnutých aplikací. Volbu *PutMsgRecFields* používejte pouze v případě, že je zpráva vložena do distribučního seznamu. Pole je ignorováno, pokud *RecsPresent* je nula, nebo obě *PutMsgRecOffset* a *PutMsgRecPtr* jsou nula.

Pro pole, která jsou přítomná, používá správce front pro každou cílovou hodnotu hodnoty z polí v odpovídajícím záznamu vložení zprávy. U nepřítomných polí používá správce front hodnoty z struktury MQMD.

Pomocí jednoho nebo více následujících příznaků určete, která pole se nacházejí v záznamech vložených zpráv:

MQPMRF_ID_ZPRÁVY
Zobrazí se pole identifikátoru zprávy.

MQPMRF_CORREL_ID
Pole identifikátoru korelace je přítomno.

ID SKUPINY MQPMRF_GROUP_ID
Pole identifikátoru skupiny je přítomno.

ZPĚTNÁ VAZBA MQPMRF_FEEDBACK
Je přítomno pole zpětné vazby.

MQPMRF_ACCOUNTING_TOKEN
Pole Účetní-token je přítomno.

Zadáte-li tento příznak, zadejte buď MQPMO_SET_IDENTITY_CONTEXT, nebo MQPMO_SET_ALL_CONTEXT v poli *Options* ; pokud tato podmínka není splněna, volání selže s kódem příčiny MQRC_PMO_RECORD_FLAGS_ERROR.

Nejsou-li přítomna žádná pole MQPMR, lze zadat následující:

MQPMRF_NONE
Nejsou přítomna žádná pole záznamu vložení zprávy.

Je-li tato hodnota uvedena, musí být buď *RecsPresent* nula, nebo obě *PutMsgRecOffset* a *PutMsgRecPtr* musí být nula.

Funkce MQPMRF_NONE je definována pro dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Pokud příkaz *PutMsgRecFields* obsahuje příznaky, které nejsou platné, nebo jsou zadány záznamy zpráv, ale *PutMsgRecFields* má hodnotu MQPMRF_NONE, volání selže s kódem příčiny MQRC_PMO_RECORD_FLAGS_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je MQPMRF_NONE. Toto pole je ignorováno, pokud *Version* je menší než MQPMO_VERSION_2.

PutMsgRecOffset (MQLONG)

Jedná se o posun v bajtech prvního záznamu vložení zprávy MQPMR ze začátku struktury MQPMO. Odsazení může být kladné nebo záporné. *PutMsgRecOffset* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *RecsPresent* je nula.

Když je zpráva vložena do distribučního seznamu, může být poskytnuto pole jednoho nebo více záznamů vložení zpráv MQPMR, aby bylo možné určit určité vlastnosti zprávy pro každý cíl jednotlivě; tyto vlastnosti jsou:

- Identifikátor zprávy
- Identifikátor korelace
- Identifikátor skupiny
- Hodnota zpětné vazby
- Token evidence

Nemusíte uvádět všechny tyto vlastnosti, ale jakoukoli vámi vybranou dílčí sadu specifikujte pole ve správném pořadí. Další podrobnosti naleznete v popisu struktury MQPMR.

Obvykle musí existovat tolik záznamů o vložení zpráv, protože při otevření distribučního seznamu jsou záznamy objektů zadány příkazem MQOD; každý záznam vložení zprávy poskytuje vlastnosti zprávy pro frontu označenou odpovídajícím záznamem objektu. Fronty v rozdělovníku, které se nedaří otevřít, musí stále mít přidělené záznamy zpráv pro ně na odpovídajících pozicích v poli, ačkoli jsou vlastnosti zprávy 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 zprávy pro místa určení, které nepřijaly záznamy zpráv, jsou převzaty z odpovídajících polí v deskriptoru zpráv MQMD. Pokud existuje více záznamů vložení zpráv než záznamů objektů, přebytek se nepoužije (ačkoli musí být stále možné k nim přistupovat). Záznamy zpráv o vložení jsou volitelné, ale pokud jsou dodány, musí být *RecsPresent* z nich.

Poskytněte záznamy vložení zpráv podobným způsobem jako záznamy objektů v MQOD, a to buď zadáním posunu v *PutMsgRecOffset*, nebo uvedením adresy v *PutMsgRecPtr* ; Podrobnosti o tom, jak to provést, naleznete v poli *ObjectRecOffset* , které je popsáno v"MQOD-Popisovač objektu" na stránce 476.

Nelze použít více než jeden z *PutMsgRecOffset* a *PutMsgRecPtr* ; volání selže s kódem příčiny MQRC_PUT_MSG_RECORS_ERROR, pokud jsou obě nenulové.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQPMO_VERSION_2.

Posunutí ResponseRec(MQLONG)

Jedná se o posun v bajtech prvního záznamu odezvy MQRR od začátku struktury MQPMO. Odsazení může být kladné nebo záporné. *ResponseRecOffset* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *RecsPresent* je nula.

Při umístění zprávy do rozdělovníku můžete zadat pole jednoho nebo více záznamů odpovědí MQRR, které budou identifikovat fronty, do kterých nebyla zpráva úspěšně odeslána (pole*CompCode* v MQRR), a důvod

pro každé selhání (pole*Reason* v MQRR). Je možné, že zpráva nebyla odeslána, protože došlo k otevření fronty, nebo došlo k selhání 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 se nezdařily, nebo všechny selhaly, ale z různých důvodů); kód příčiny MQRC_MULTIPLE_REASONS z volání označuje tento případ. Pokud se stejný kód příčiny vztahuje na všechny fronty, je tento důvod vrácen v parametru **Reason** volání MQPUT nebo MQPUT1 a záznamy odezvy nejsou nastaveny.

Obvykle existuje tolik záznamů odpovědí jako jsou záznamy objektů zadané příkazem MQOD při otevření distribučního seznamu; je-li to nutné, každý záznam odpovědi je nastaven na kód dokončení a kód příčiny pro vložení do fronty označené odpovídajícím záznamem objektu. Fronty v rozdělovníku, které se nedaří otevřít, musí mít stále alokovány záznamy odpovědí na odpovídajících pozicích v poli, ačkoli jsou nastaveny na kód dokončení a kód příčiny, který je výsledkem operace otevření, spíše než operace vložení.

Počet záznamů odezvy se může lišit od počtu záznamů objektů. Pokud existuje méně záznamů odezev než záznamů objektů, aplikace nemusí být schopna identifikovat všechna místa určení, pro které došlo k selhání operace vložení, nebo příčiny selhání. Pokud existuje více záznamů odezev než záznamů objektů, přebytek se nepoužije (ačkoli musí být stále možné k nim přistupovat). Záznamy odezvy jsou volitelné, ale pokud jsou dodány, musí být *RecsPresent* z nich.

Poskytněte záznamy odezvy podobným způsobem jako záznamy objektů v MQOD, a to buď zadáním posunu v *ResponseRecOffset*, nebo uvedením adresy v *ResponseRecPtr* ; Podrobnosti o tom, jak to provést, naleznete v poli *ObjectRecOffset* , které je popsáno v “MQOD-Popisovač objektu” na stránce 476. Avšak použijte ne více než jeden z *ResponseRecOffset* a *ResponseRecPtr* ; volání selže s kódem příčiny MQRC_RESPONSE_RECORS_ERROR, pokud jsou obě nenulové.

Pro volání MQPUT1 musí být toto pole nulové. Důvodem je to, že informace o odezvě (je-li požadována) jsou vraceny v záznamech odpovědí určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQPMO_VERSION_2.

PutMsgRecPtr (MQPTR)

Jedná se o adresu prvního záznamu vložení zprávy MQPMR. Volbu *PutMsgRecPtr* používejte pouze v případě, že je zpráva vložena do distribučního seznamu. Pole je ignorováno, pokud *RecsPresent* je nula.

Můžete použít buď *PutMsgRecPtr* nebo *PutMsgRecOffset* k zadání vložených záznamů zpráv, ale ne obojí; podrobnosti viz “PutMsgRecOffset (MQLONG)” na stránce 510. Pokud nepoužíváte *PutMsgRecPtr*, nastavte ji na nulový ukazatel nebo na null bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je 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 hodnotou je řetězec bajtů se všemi znaky null.

ResponseRecPtr (MQPTR)

Jedná se o adresu prvního záznamu odezvy MQRR. *ResponseRecPtr* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *RecsPresent* je nula.

Chcete-li určit záznamy odpovědí, ale ne obojí, použijte buď *ResponseRecPtr* nebo *ResponseRecOffset* ; podrobnosti viz “Posunutí ResponseRec(MQLONG)” na stránce 510. If you do not use *ResponseRecPtr* set it to the null pointer or null bytes.

Pro volání MQPUT1 musí být toto pole ukazatelem null nebo null bajtů. Důvodem je to, že informace o odezvě (je-li požadována) jsou vraceny v záznamech odpovědí určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null. Toto pole je ignorováno, pokud *Version* je 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 hodnotou je řetězec bajtů se všemi znaky null.

OriginalMsgHandle (MQHMSG)

Toto je volitelný odkaz na zprávu. Možná byla dříve načtena z fronty. Použití tohoto manipulátoru je předmětem hodnoty pole *Action* ; viz také NewMsgHandle.

Obsah původní obslužné rutiny zprávy nebude změněn pomocí volání **MQPUT** nebo **MQPUT1** .

Toto je vstupní pole. Počáteční hodnota tohoto pole je **MQHM_NONE**. Toto pole je ignorováno, pokud je verze nižší než **MQPMO_VERSION_3**.

NewMsgPopisovač (MQHMSG)

Jedná se o volitelný úchyt pro zprávu, na kterou se vztahuje hodnota pole Akce. Definuje vlastnosti zprávy a přepisuje hodnoty parametru *OriginalMsgHandle*, pokud je zadán.

Při návratu z volání **MQPUT** nebo **MQPUT1** odráží obsah manipulátoru zprávu, která byla ve skutečnosti vložena.

Toto je vstupní pole. Počáteční hodnota tohoto pole je **MQHM_NONE**. Toto pole je ignorováno, pokud je verze nižší než **MQPMO_VERSION_3**.

Akce (MQLONG)

Uvádí typ prováděné operace a vztah mezi původní zprávou uvedenou v poli Popisovač OriginalMsga novou zprávou uvedenou v poli Popisovač NewMsg. Vlastnosti zprávy jsou zvoleny správcem front podle hodnoty uvedené akce.

Obsah deskriptoru zpráv můžete zadat pomocí parametru MsgDesc na volání MQPUT nebo MQPUT1 . Případně je možné nezadat parametr MsgDesc nebo zadat, že se jedná o výstup-pouze zahrnující MQPMO_MD_FOR_OUTPUT_ONLY v poli Volby struktury MQPMO.

Není-li zadán parametr MsgDesc nebo je-li zadán pouze na výstupu, je deskriptor zprávy pro novou zprávu naplněn daty z polí pro obsluhu zpráv MQPMO podle pravidel popsaných v tomto tématu.

Nastavení kontextu a předávání aktivit popsané v tématu Řízení informací o kontextu nabývají účinku po sestavení deskriptoru zprávy.

Je-li zadána nesprávná hodnota akce, volání selže s kódem příčiny MQRC_ACTION_ERROR.

Může být uvedena některá z následujících akcí:

NOVÁ HODNOTA MQACTP_NEW
Probíhá vkládání nové zprávy a tento program nezadá žádný vztah k předchozí zprávě. Deskriptor zprávy se skládá z následujících kroků:

- Je-li zadán příkaz MsgDesc v rámci volání MQPUT nebo MQPUT1 a MQPMO_MD_FOR_OUTPUT_ONLY není v adresáři MQPMO.Optionsse používají jako nemodifikované popisovače zpráv.
- Není-li zadán parametr MsgDesc nebo MQPMO_MD_FOR_OUTPUT_ONLY, je uveden v záhlaví MQPMO.Options pak správce front vygeneruje deskriptor zprávy pomocí kombinace vlastností z ovladače OriginalMsga NewMsgHandle. Jakákoli pole deskriptoru zprávy explicitně nastavená na novém popisovači zprávy mají přednost před těmi, které jsou v původním popisovači zprávy.

Data zprávy jsou převzata z parametru MQPUT nebo MQPUT1 .

MQACTP_FORWARD
Posílá se dříve načtená zpráva. Původní popisovač zprávy určuje zprávu, která byla dříve načtena.

Nový popisovač zprávy určuje jakékoliv změny vlastností (včetně libovolného v deskriptoru zpráv) v původním popisovači zprávy.

Deskriptor zprávy se skládá z následujících kroků:

- Je-li zadán příkaz MsgDesc v rámci volání MQPUT nebo MQPUT1 a MQPMO_MD_FOR_OUTPUT_ONLY není v adresáři MQPMO.Optionsse používají jako nemodifikované popisovače zpráv.
- Není-li zadán parametr MsgDesc nebo MQPMO_MD_FOR_OUTPUT_ONLY, je uveden v záhlaví MQPMO.Options pak správce front vygeneruje deskriptor zprávy pomocí kombinace vlastností z ovladače OriginalMsga NewMsgHandle. Jakákoli pole deskriptoru zprávy explicitně nastavená na novém popisovači zprávy mají přednost před těmi, které jsou v původním popisovači zprávy.
- Je-li hodnota MQPMO_NEW_MSG_ID nebo MQPMO_NEW_CORREL_ID zadána v objektu MQPMO.Options, pak jsou tyto volby dodrženy.

Vlastnosti zprávy se skládají z následujících hodnot:

- Všechny vlastnosti z původního popisovače zprávy, které mají MQCOPY_FORWARD, v MQPD.CopyOptions
- Všechny vlastnosti z nové obslužné rutiny zpráv. Pro každou vlastnost v novém popisovači zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nové obslužné rutiny zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy vlastnost v novém popisovači zprávy má 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 MQPUT nebo MQPUT1 .

MQACTP_REPLY
Odpověď se provádí na dříve načtenou zprávu. Původní popisovač zprávy určuje zprávu, která byla dříve načtena.

Nový popisovač zprávy určuje jakékoliv změny vlastností (včetně libovolného v deskriptoru zpráv) v původním popisovači zprávy.

Deskriptor zprávy se skládá z následujících kroků:

- Je-li zadán příkaz MsgDesc v rámci volání MQPUT nebo MQPUT1 a MQPMO_MD_FOR_OUTPUT_ONLY není v adresáři MQPMO.Optionsse používají jako nemodifikované popisovače zpráv.
- Není-li zadán parametr MsgDesc nebo MQPMO_MD_FOR_OUTPUT_ONLY, je uveden v záhlaví MQPMO.Options, pak jsou počáteční pole deskriptoru zpráv zvolena takto:

Tabulka 510. Transformace popisovače zprávy odpovědi

Pole v MQMD	**Použitá hodnota**
Sestava	Pokud MQRO_PASS_DISCARD_AND_EXPIRY a MQRO_DISCARD_MSG jsou nastaveny: MQRO_DISCARD_MSG jinak MQRO_NONE
MsgType	MQMT_REPLY
Vypršení	Pokud MQRO_PASS_DISCARD_AND_EXPIRY je nastaveno: Zkopírováno ze vstupní zprávy jinak MQEI_UNLIMITED
Zpětná vazba	MQFB_NONE

Tabulka 510. Transformace popisovače zprávy odpovědi (pokračování)	
Pole v MQMD	**Použitá hodnota**
MsgId	Je-li hodnota MQPMO_NEW_MSG_ID nastavena: Je vygenerován nový identifikátor zprávy else if MQRO_PASS_MSG_ID is set: Zkopírováno ze vstupní zprávy jinak MQMI_NONE
CorrelId	Je-li MQPMO_NEW_CORREL_ID nastaveno: Je vygenerován nový korelační identifikátor else if MQRO_COPY_MSG_ID_TO_CORREL_ID je nastaven: Zkopírováno z pole MsgId v Vstupní zpráva jinak je-li hodnota MQRO_PASS_CORREL_ID nastavena: Zkopírováno z pole CorrelId v 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 zpráv je pak upraven novým popisovačem zprávy-jakákoli pole deskriptoru zprávy explicitně nastavená jako vlastnosti v novém popisovači zprávy mají přednost před poli deskriptoru zpráv, jak bylo popsáno výše.

Vlastnosti zprávy se skládají z následujících hodnot:

- Všechny vlastnosti z popisovače původní zprávy, které mají MQCOPY_REPLY, v MQPD.CopyOptions
- Všechny vlastnosti z nové obslužné rutiny zpráv. Pro každou vlastnost v novém popisovači zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nové obslužné rutiny zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy vlastnost v novém popisovači zprávy má 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 MQPUT/MQPUT1 .

SESTAVA MQACTP_REPORT

Sestava je generována jako výsledek dříve načtené zprávy. Původní popisovač zprávy určuje zprávu, která způsobí generování sestavy.

Nový popisovač zprávy určuje jakékoliv změny vlastností (včetně libovolného v deskriptoru zpráv) v původním popisovači zprávy.

Deskriptor zprávy se skládá z následujících kroků:

- Je-li zadán příkaz MsgDesc v rámci volání MQPUT nebo MQPUT1 a MQPMO_MD_FOR_OUTPUT_ONLY není v adresáři MQPMO.Optionsse používají jako nemodifikované popisovače zpráv.
- Není-li zadán parametr MsgDesc nebo MQPMO_MD_FOR_OUTPUT_ONLY, je uveden v záhlaví MQPMO.Options a pak počáteční pole deskriptoru zpráv jsou vybrána takto:

Tabulka 511. Transformace popisovače zprávy sestavy

Pole v MQMD	**Použitá hodnota**
Sestava	Pokud MQRO_PASS_DISCARD_AND_EXPIRY a MQRO_DISCARD_MSG je nastaveno: MQRO_DISCARD_MSG jinak MQRO_NONE
MsgType	SESTAVA MQMT_REPORT
Vypršení	Pokud MQRO_PASS_DISCARD_AND_EXPIRY je nastaveno: Zkopírováno ze vstupní zprávy jinak MQEI_UNLIMITED
MsgId	Je-li hodnota MQPMO_NEW_MSG_ID nastavena: Je vygenerován nový identifikátor zprávy else if MQRO_PASS_MSG_ID is set: Zkopírováno ze vstupní zprávy jinak MQMI_NONE
CorrelId	Je-li MQPMO_NEW_CORREL_ID nastaveno: Je vygenerován nový korelační identifikátor else if MQRO_COPY_MSG_ID_TO_CORREL_ID je nastaven: Zkopírováno z pole MsgId v Vstupní zpráva jinak je-li hodnota MQRO_PASS_CORREL_ID nastavena: Zkopírováno z pole CorrelId v Vstupní zpráva jinak MQCI_NONE
BackoutCount	0
ReplyToQ	Mezery
ReplyToQMgr	Mezery
OriginalLength	Nastavit na *BufferLength*

- Deskriptor zpráv je pak upraven novým popisovačem zprávy-jakákoli pole deskriptoru zprávy explicitně nastavená jako vlastnosti v novém popisovači zprávy mají přednost před poli deskriptoru zpráv, jak bylo popsáno výše.

Vlastnosti zprávy se skládají z následujících hodnot:

- Všechny vlastnosti z popisovače původní zprávy, které mají MQCOPY_REPORT v MQPD.CopyOptions

- Všechny vlastnosti z nové obslužné rutiny zpráv. Pro každou vlastnost v novém popisovači zprávy, která má stejný název jako vlastnost v původním popisovači zprávy, je hodnota převzata z nové obslužné rutiny zprávy. Jedinou výjimkou z tohoto pravidla je speciální případ, kdy vlastnost v novém popisovači zprávy má stejný název jako vlastnost v původním popisovači zprávy, ale hodnota vlastnosti je null. V tomto případě je vlastnost odebrána ze zprávy.

Pole Názor ve výsledném deskriptoru MQMD představuje sestavu, která má být generována. Hodnota zpětné vazby MQFB_NONE způsobí, že volání MQPUT nebo MQPUT1 selže 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 MQMD nebo MQPUT1 .

- Je-li zpětná vazba MQFB_COA, MQFB_COD nebo MQFB_EXPIRATION, pak je zkontrolována hodnota sestavy.
- Je-li některý z následujících případů pravdivý, použije se úplná data zprávy z vyrovnávací paměti o délce BufferLength .
  - Feedback je MQFB_EXPIRATION and Report contains 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 tato hodnota menší než 100) z vyrovnávací paměti.
  - Zpětná vazba je MQFB_EXPIRATION a Report 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 spolu se zprávou zahrnuta žádná uživatelská data.
- V případě, že zpětná vazba má jinou hodnotu než výše uvedená hodnota, budou použita vyrovnávací paměť a hodnota BufferLength jako normální.

Odvození uživatelských dat popsaných v předchozím seznamu je také zobrazeno v následující tabulce:

Tabulka 512. Zdroj uživatelských dat

	MQFB_COA	**MQFB_COD**	**MQFB_EXPIRATION**
MQRO_EXPIRATION_WITH_FULL_DATA	Není	Není	Vyrovnávací paměť (Bufferlength)
MQRO_COD_WITH_FULL_DATA	Není	Vyrovnávací paměť (Bufferlength)	Není
MQRO_COA_WITH_FULL_DATA	Vyrovnávací paměť (Bufferlength)	Není	Není
MQRO_EXPIRATION_WITH_DATA	Není	Není	Vyrovnávací paměť (prvních 100 bajtů)
MQRO_CED_WITH_DATA	Není	Vyrovnávací paměť (prvních 100 bajtů)	Není
MQRO_COA_WITH_DATA	Vyrovnávací paměť (prvních 100 bajtů)	Není	Není

PubLevel (MQLONG)

Počáteční hodnota tohoto pole je 9. Úroveň odběru, na kterou je tato publikace zaměřena. Tato publikace je určena pouze pro odběry s nejvyšší úrovní SubLevel nižší nebo rovnou této hodnotě. Tato hodnota musí

být v rozsahu nula až 9; nula je nejnižší úroveň. Pokud však byla publikace zachována, není již dostupná odběratelům na vyšší úrovni, protože je znovu publikována na úrovni PubLevel 1.

Další informace najdete 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 513. 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)

Jedná se o identifikátor zprávy, který má být použit pro zprávu odeslanou do fronty názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MsgId* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta 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 dvě z těchto míst určení nemají stejný identifikátor zprávy).

Je-li zadáno MQPMO_NEW_MSG_ID, nové identifikátory zpráv jsou generovány pro všechna místa určení v seznamu distribucí bez ohledu na to, zda mají záznamy MQPMR. To se liší od způsobu zpracování operace MQPMO_NEW_CORREL_ID (viz pole *CorrelId*).

Jedná se o vstupní/výstupní pole.

CorrelId (MQBYTE24)

Jedná se o identifikátor korelace, který má být použit pro zprávu odeslanou do fronty názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *CorrelId* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta místa určení, která nemají záznam MQPMR obsahující pole *CorrelId* .

Je-li zadán parametr MQPMO_NEW_CORREL_ID, bude vygenerován a použit *jediný* nový korelační identifikátor, který bude použit pro všechna místa určení v seznamu distribucí bez ohledu na to, zda mají záznamy MQPMR. To se liší od způsobu zpracování MQPMO_NEW_MSG_ID (viz pole *MsgId*).

Jedná se o vstupní/výstupní pole.

GroupId (MQBYTE24)

GroupId je identifikátor skupiny, který se má použít pro zprávu odeslanou do fronty s názvem, který byl určen příslušným prvkem v poli struktur MQOR poskytnutém na volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *GroupId* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta místa určení, která nemají záznam MQPMR obsahující pole *GroupId* . Hodnota je zpracována tak, jak je dokumentováno ve Fyzickém pořadí na frontě, ale s následujícími rozdíly:

- Položka GroupId se vytvoří z identifikátoru QMName a z časového razítka. Proto zachovat jedinečné názvy správce front GroupId také jedinečné. Také nenastavujte hodiny na počítači se správci front.
- V případech, kdy se použije nový identifikátor skupiny, vygeneruje správce front jiný identifikátor skupiny pro každé místo určení (to znamená, že žádná dvě místa určení nemají stejný identifikátor skupiny).
- V takových případech, kdy se hodnota v poli použije, volání selže s kódem příčiny MQRC_GROUP_ID_ERROR

Jedná se o vstupní/výstupní pole.

Zpětná vazba (MQLONG)

Jedná se o kód zpětné vazby, který má být použit pro zprávu odeslanou do fronty názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *Feedback* v produktu MQMD pro vložení do jedné fronty.

Není-li toto pole k dispozici, bude použita hodnota v produktu MQMD.

Toto je vstupní pole.

AccountingToken (MQBYTE32)

Jedná se o účtovací token, který má být použit pro zprávu odeslanou do fronty s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *AccountingToken* v produktu MQMD pro vložení do jedné fronty. Chcete-li získat informace o obsahu tohoto pole, prohlédněte si popis *AccountingToken* v “MQMD-Deskriptor zpráv” na stránce 419 .

Není-li toto pole k dispozici, bude použita hodnota v produktu 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 514. 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
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

Tabulka 514. Pole v MQRFH pro MQRFH (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.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)

Jedná se o identifikátor struktury; hodnota musí být:

MQRFH_STRUCT
Identifikátor pro pravidla a formátování struktury záhlaví.

Pro programovací jazyk C je také definován konstantní MQRFH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQRFH_STRUC_ID, ale je to pole znaků namísto řetězce.

Počáteční hodnota tohoto pole je MQRFH_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQRFH_VERSION_1
Pravidla Version-1 a formátovací struktura záhlaví.

Počáteční hodnota tohoto pole je MQRFH_VERSION_1.

StrucLength (MQLONG)

Jedná se o délku struktury MQRFH v bajtech, včetně pole *NameValueString* na konci struktury. Tato délka nezahrnuje žádná uživatelská data, která následují za polem *NameValueString* .

Aby se zabránilo problémům při převádění uživatelských dat v některých prostředích, musí být *StrucLength* násobkem čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. o délce kromě pole *NameValueString* :

PEVNÉ PRODLOUŽENÍ MQRFH_STRUCLOTH_FIXED
Délka pevné části struktury MQRFH.

Počáteční hodnota tohoto pole je MQRFH_STRUC_LENGTH_FIXED.

Kódování (MQLONG)

Tato hodnota určuje číselné kódování dat, která jsou následující: *NameValueString* ; Nevztahuje se na číselná data ve struktuře MQRFH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG)

Tato hodnota určuje identifikátor znakové sady dat, která následují za *NameValueString* ; Nevztahuje se na znaková data v samotné struktuře MQRFH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru MQMD MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8)

Uvádí název formátu dat, která následují za *NameValueString*.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole *Format* v produktu MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

Je možné zadat následující:

MQRFH_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQRFH_NONE.

NameValueString (MQCHARn)

Jedná se o znakový řetězec proměnné délky obsahující dvojice název-hodnota ve formuláři:

```
name1 value1 name2 value2 name3 value3 ...
```

Každý název nebo hodnota musí být oddělena od sousedního názvu nebo hodnoty jedním nebo více prázdnými znaky; tyto mezery nejsou významné. Název nebo hodnota může obsahovat významné mezery tak, že se k názvu nebo hodnotě připojí dvojité uvozovky; všechny znaky mezi otevřenou dvojitými uvozovkami a odpovídajícími uzavíraní dvojitými uvozovkami se považují za významné. V následujícím příkladu je název `FAMOUS_WORDS`a hodnota je `Hello World`:

```
FAMOUS_WORDS "Hello World"
```

Název nebo hodnota může obsahovat jiné znaky než znak null (které se chová jako oddělovač pro *NameValueString*). Aby však mohla aplikace pomoci s interoperabilitou, aplikace může omezit názvy na následující znaky:

- První znak: velká nebo malá písmena (A až Z, nebo a až z) nebo podtržítko.
- Následné znaky: velká nebo malá abecední, desetinná číslice (0 až 9), podtržítko, pomlčka nebo tečka.

Pokud název nebo hodnota obsahuje jednu nebo více dvojitých uvozovek, musí být název nebo hodnota ohraničena dvojitými uvozovkami a každá dvojitá uvozovka v řetězci musí být zdvojená:

```
Famous_Words "The program displayed ""Hello World"""
```

Názvy a hodnoty rozlišují velikost písmen, 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 převádění uživatelských dat v některých prostředích, vytvořte tuto délku více než čtyři. Pad *NameValueString* s mezerami na tuto délku, nebo ukončete jej dříve umístěním znaku null za posledním významným znakem v řetězci. Nulový znak a bajty po něm až do zadané délky *NameValueString*jsou ignorovány.

Poznámka: Vzhledem k tomu, že délka tohoto pole není pevná, je pole vynecháno z deklarací struktury, které jsou poskytovány 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

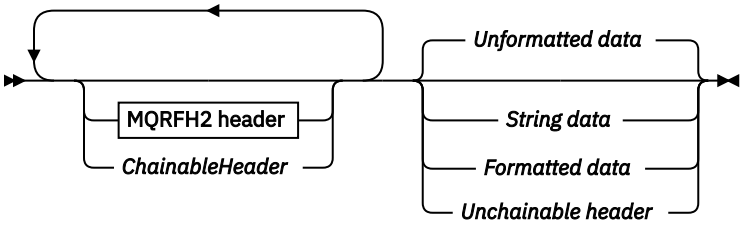
Všechny systémy IBM MQ a IBM MQ MQI clients připojené k těmto systémům.

Název formátu

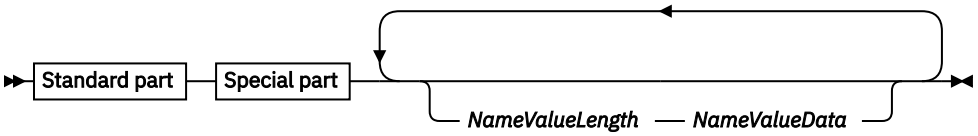
MQFMT_RF_HEADER_2

Syntax

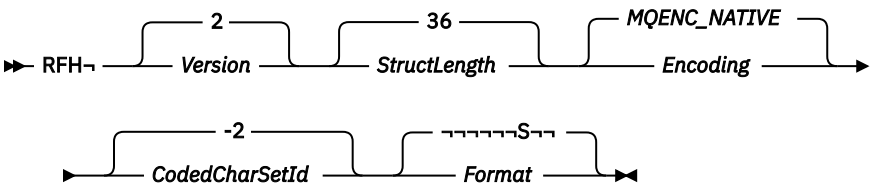
IBM MQ Message



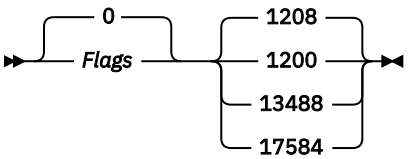
MQRFH2 header



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 515. 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

Tabulka 515. Pole v MQRFH2 pro MQRFH2 (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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)

Jedná se o identifikátor struktury; hodnota musí být:

MQRFH_STRUCT
Identifikátor pro pravidla a formátování struktury záhlaví.

Pro programovací jazyk C je také definován konstantní MQRFH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQRFH_STRUC_ID, ale je to pole znaků namísto řetězce.

Počáteční hodnota tohoto pole je MQRFH_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQRFH_VERSION_2
Version-2 pravidla a formátování struktury záhlaví.

Počáteční hodnota tohoto pole je MQRFH_VERSION_2.

StrucLength (MQLONG)

Jedná se o délku v bajtech struktury MQRFH2 , včetně polí *NameValueLength* a *NameValueData* na konci struktury. Je platný pro více párů polí *NameValueLength* a *NameValueData* na konci struktury, v posloupnosti:

```
length1, data1, length2, data2, ...
```

StrucLength nezahrnuje žádná uživatelská data, která by mohla následovat za posledním polem *NameValueData* na konci struktury.

Chcete-li se vyvarovat problémů s převodem uživatelských dat v některých prostředích, musí být *StrucLength* násobkem čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. o délce kromě 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)

Určuje číselné kódování dat, která následují za posledním polem *NameValueData* . Nevztahuje se na číselná data ve struktuře MQRFH2 .

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG)

Tato hodnota určuje identifikátor znakové sady dat, která následuje za posledním polem *NameValueData* . Nevztahuje se na znaková data ve struktuře MQRFH2 .

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru MQMD MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_INHERIT.

Formát (MQCHAR8)

Uvádí jméno formátu dat, která následuje za posledním polem *NameValueData* .

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole *Format* v produktu MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

Počáteční hodnota tohoto pole je MQRFH_NONE. MQRFH_NONE musí být zadaný.

MQRFH_NONE
Žádné vlajky.

MQRFH_INTERNAL
Záhlaví MQRFH2 obsahuje interně nastavené vlastnosti.

MQRFH_INTERNAL je určen pro použití správcem front.

Prvních 16 bitů, MQRFH_FLAGS_RESTRICTED_MASK, jsou rezervovány pro nastavení příznaků správce front. Parametry, které může uživatel nastavit, jsou definovány v posledních 16 bitech.

NameValueCCSID (MQLONG)

Tato hodnota určuje identifikátor kódované znakové sady pro data v poli *NameValueData* . To se liší od znakové sady jiných 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
13488	UTF-16, verze Unicode, podmnožina 2.0
17584	UTF-16, verze Unicode 3.0 dílčí sada (obsahuje symbol Euro)
1208	UTF-8, nejnovější podporovaná verze Unicode

Pro znakové sady UTF-16 musí být kódování (pořadí bajtů) produktu *NameValueData* stejné jako kódování ostatních polí ve struktuře MQRFH2 .

Znaky nad rámec Unicode Basic Multilingual Plane (ty, které jsou vyšší než U + FFFF), představované v souboru UTF-16 pomocí náhradních kódových bodů (X'D800'až X'DFFF'), nebo 4 bajty v UTF-8, nejsou podporovány.

Poznámka: Pokud *NameValueCCSID* nemá jednu z výše uvedených hodnot a struktura MQRFH2 vyžaduje převod na volání MQGET, volání bude dokončeno s kódem příčiny MQRC_SOURCE_CCSID_ERROR a zpráva je vrácena nekonverzovanou.

Počáteční hodnota tohoto pole je 1208.

NameValueDélka (MQLONG)
Délka odpovídajícího pole NameValueData

Určuje délku dat v poli *NameValueData* v bajtech. *NameValueLength* musí být násobkem čtyř.

Poznámka: Pole *NameValueLength* a *NameValueData* jsou volitelná, ale pokud jsou přítomna, musí se objevit jako pár a být sousedící. Dvojice polí se mohou opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože tato pole jsou volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

Data NameValueData (MQCHARn)
Pole *NameValueData* je pole s proměnnou délkou, které obsahuje složku obsahující dvojice název-hodnota vlastností zprávy. Složka je řetězec znaků s proměnnou délkou, který obsahuje data zakódovaná pomocí syntaxe jako XML. Délka znakového řetězce v bajtech je dána polem *NameValueLength* , která předchází poli *NameValueData* . Délka musí být násobkem čtyř.

Pole *NameValueDélka* a *NameValueData* jsou volitelná, ale pokud jsou přítomna, musí se vyskytovat jako dvojice a musí být sousední. Dvojice polí se mohou opakovat tolikrát, kolikrát je třeba, například:

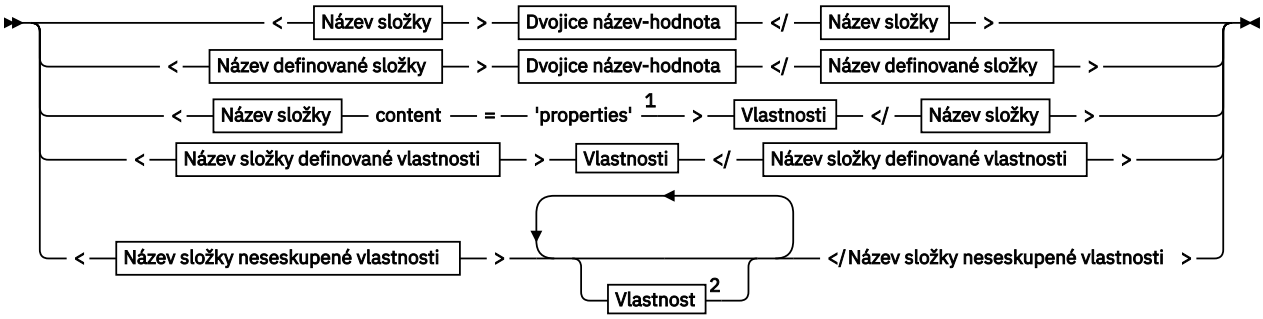
```
length1 data1 length2 data2 length3 data3
```

Položka *NameValueData* není převedena na znakovou sadu zadanou v rámci volání MQGET . I když je zpráva načtena s volbou MQGMO_CONVERT v akci *NameValueData* zůstane ve své původní znakové sadě. Hodnota *NameValueData* je však převedena na kódování zadané ve výzvě MQGET .

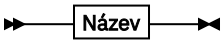
Notes:

- Protože tato pole jsou volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.
- Termíny "definované" a "vyhrazené" se používají v diagramu syntaxe. Hodnota "Definováno" znamená, že název je používán produktem IBM MQ. "Vyhrazeno" znamená, že název je vyhrazen pro budoucí použití produktem IBM MQ.

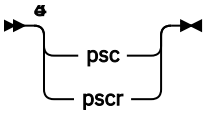
NameValueData syntaxe



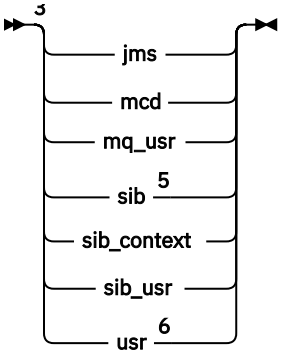
Název složky



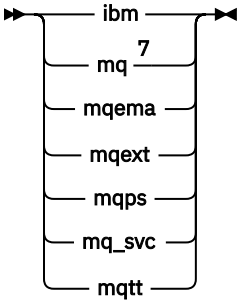
Název definované složky



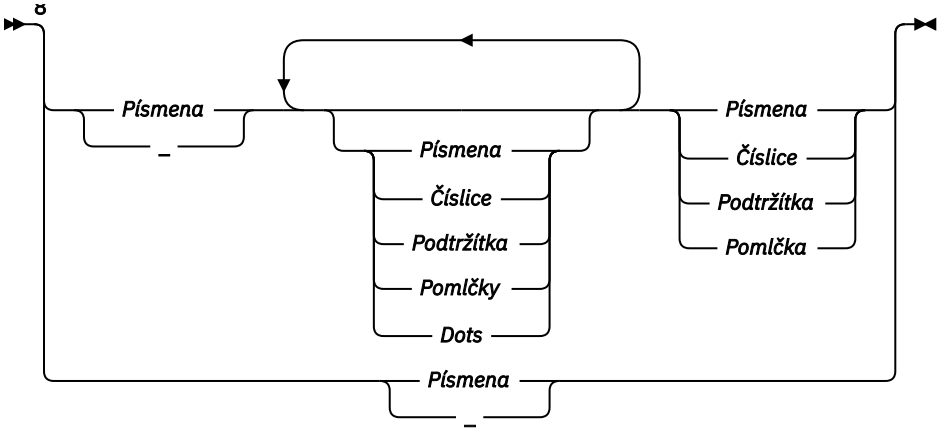
Název složky definované vlastnosti



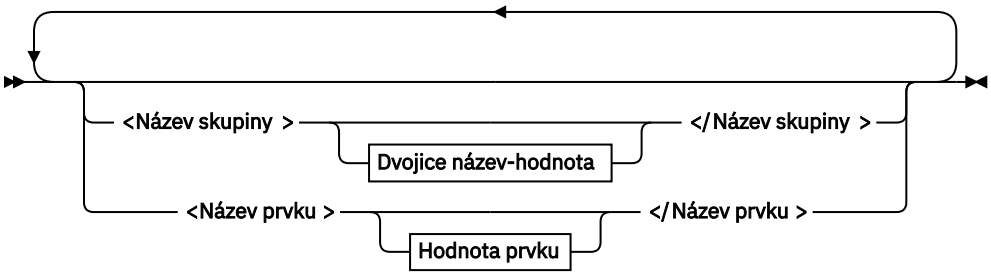
Název složky neseskupené vlastnosti



Název

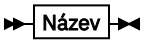


Dvojice název-hodnota

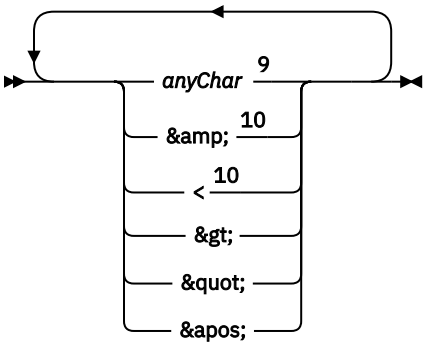


Název skupiny

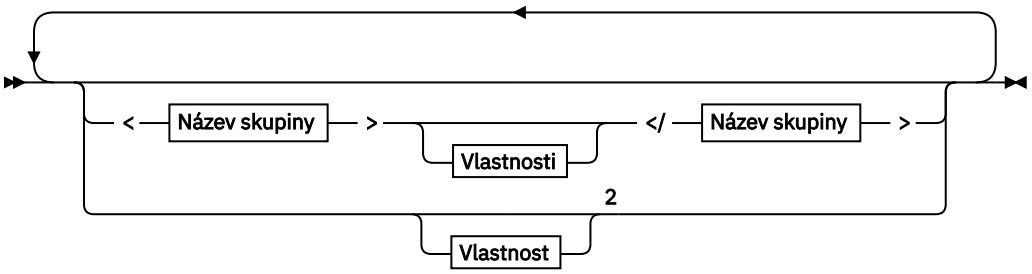
Název prvku



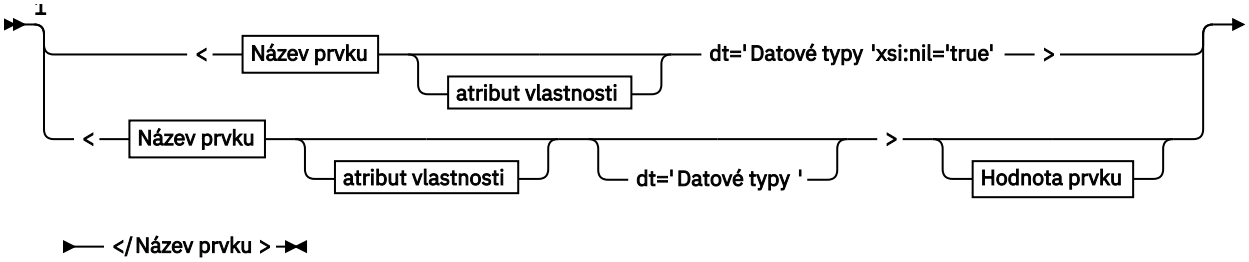
Hodnota prvku



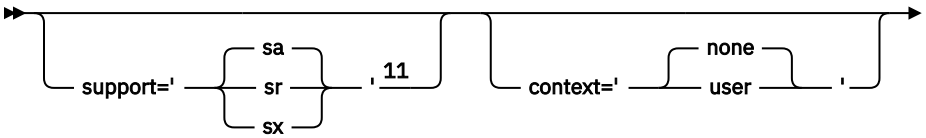
Vlastnosti

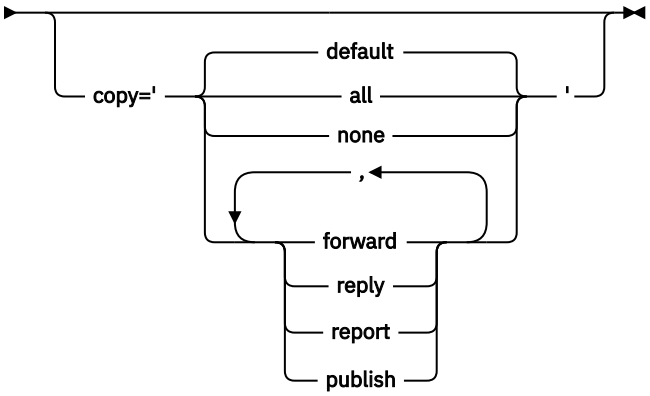


Vlastnost

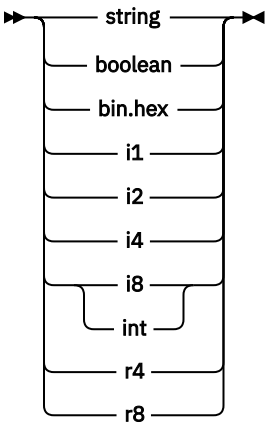


atribut vlastnosti





Datové typy



Poznámky:

[1] Dvojité uvozovky nebo jednoduché uvozovky jsou platné.

[2] Nepoužijte neplatný název vlastnosti, viz “Neplatný název vlastnosti” na stránce 544. Použijte vyhrazený název vlastnosti pouze pro její definovaný účel; viz “Definované názvy vlastností” na stránce 544.

[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 jsou významné pouze vlastnosti v prvním záhlaví MQRFH2 .

[6] V produktu MQRFH2nesmí být více než jedna složka `usr` . Vlastnosti ve složce `usr` se nesmí vyskytovat více než jednou.

[7] Významné jsou pouze vlastnosti v první složce `mq` . Je-li pořadač UTF-8, jsou podporovány pouze jednobajtové UTF-8 znaky. Jediný netisknuznakový znak je Unicode U+0020.

[8] Platné znaky jsou definovány ve specifikaci XML W3C a skládají se hlavně z kategorií Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` a Nd. Viz Znakové kategorie Unicode.

[9] Všechny znaky jsou významné. Počáteční a koncové mezery jsou součástí hodnoty prvku.

[10] Nepoužít neplatný znak; viz “Neplatné znaky” na stránce 543. Použijte řídicí posloupnost, spíše než tyto neplatné znaky.

[11] Atribut vlastnosti podpory je platný pouze ve složce `mq` .

Název složky

Položka *NameValueData* obsahuje jedinou 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 každá obsahuje více polí *NameValueData* .

Pořadí záhlaví MQRFH2 a pořadí polí *NameValueData* nečiní žádný rozdíl v logickém obsahu složky. Je-li stejná složka přítomna více než jednou ve zprávě, složka je analyzována jako celek. Pokud se stejná vlastnost vyskytuje ve více instancích stejné složky, je analyzována jako seznam.

Správná analýza MQRFH2 není ovlivněna alternativními způsoby, jak může být složka fyzicky uložena ve zprávě.

Toto pravidlo se neřídí čtyřmi složkami. Analyzována je pouze první instance složky `mq`, `sib`, `sib_contexta` `sib_usr` .

Pokud se stejná vlastnost vyskytuje v kombinaci obsahu zřetězených záhlaví MQRFH2 více než jednou, bude analyzována pouze první instance této vlastnosti. Je-li vlastnost nastavena pomocí volání API, jako např. MQSETMPa přidáno do MQRFH2 přímo aplikací, volání rozhraní API má přednost.

Název složky je název složky obsahující dvojice název-hodnota nebo skupiny. Skupiny a dvojice název-hodnota mohou být ve stromu složek promíchána ve stejné úrovni. Viz Obrázek 1 na stránce 534. Nekombinujte název skupiny a název prvku, viz Obrázek 2 na stránce 534.

```
<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 rezervovaný název složky, viz “Neplatný název cesty” na stránce 543 a “Vyhrazená složka nebo název složky vlastností” na stránce 543. Použít definovaný název složky pouze pro její definovaný účel; viz “Název definované složky” na stránce 535.

Přidáte-li atribut `'content=properties'` do značky názvu složky, stane se tato složka složkou vlastností; viz Obrázek 3 na stránce 534.

```
<myFolder></myfolder>
<myPropertyFolder contents='properties'></myPropertyFolder>
```

Obrázek 3. Příklad složky a složky vlastností

Názvy složek rozlišují velikost písmen. Názvy složek a názvy složek vlastností sdílejí stejný obor názvů. Musí mít odlišné názvy. Například, `Folder1` v Obrázek 4 na stránce 534 musí být jiný název než `Folder2` v Obrázek 5 na stránce 534.

```
<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 názvu a hodnoty v různých složkách mají různé obory názvů. `Property1` v Obrázek 5 na stránce 534 je jiná vlastnost na `Property1` v Obrázek 6 na stránce 534.

```
<Folder3 content='properties'><Property1>value</Property1></Folder3>
```

Obrázek 6. `Folder3` prostor jmen

Složky vlastností se liší od složek bez vlastností ve dvou důležitých aspektech:

1. Složky vlastností obsahují vlastnosti a složky bez vlastností obsahují dvojice název-hodnota. Složky se mírně liší, syntakticky.

2. Pro přístup k vlastnostem zprávy použijte definovaná rozhraní, jako jsou vlastnosti zprávy MQI nebo JMS vlastností zprávy. Rozhraní zajišťují, že složky vlastností v produktu MQRFH2 jsou dobře formované. Mezi správci front na různých platformách a v různých verzích je interoperabilní složka vlastností interoperabilní.

Vlastnost MQI MQI je robustním způsobem čtení a zápisu MQRFH2a odstraňuje potíže se syntaktickou analýzou správného objektu MQRFH2 .

Název definované složky

Definovaným názvem složky je název složky, která je vyhrazená pro použití produktem IBM MQnebo jiným produktem. Nevytvářejte složku se stejným názvem a nepřidávejte své vlastní dvojice název-hodnota do složek. Definované složky jsou psc a pscr.

psc a pscr se používají ve frontě publikování/odběru.

Segmentovaná zpráva s názvem MQMF_SEGMENT nebo MQMF_SEGMENTATION_ALLOWED nemůže obsahovat MQRFH2 s definovaným názvem složky. MQPUT selže s kódem příčiny 2443, MQRC_SEGMENTATION_NOT_ALLOWED.

Název složky definované vlastnosti

Definované jméno složky vlastností představuje název složky vlastností, kterou používá produkt IBM MQnebo jiný produkt. Názvy složek a jejich obsah naleznete v tématu Složky vlastností. Definované názvy složek vlastností jsou podmnožinou všech názvů složek vyhrazených IBM MQ. Viz téma “Vyhrazená složka nebo název složky vlastností” na stránce 543.

Jakýkoli prvek uložený ve složce vlastností je vlastnost. Prvek, který je uložen ve složce vlastností, nesmí mít atribut content='properties' .

Vlastnosti můžete přidat pouze k definovaným složkám vlastností usr, mq_usra sib_usr. V jiných složkách vlastností, jako jsou mq a sib, IBM MQ ignoruje nebo vyhodí vlastnosti, které nerozezná.

Popis každé definované složky vlastností obsahuje seznam vlastností, které IBM MQ definuje a které mohou být použity aplikačním programem. K některým vlastnostem se přistupuje nepřímo nastavením nebo získáním vlastnosti JMS a k některým z nich lze přistupovat přímo pomocí volání MQI MQSETMP a MQINQMP .

Definované složky vlastností také obsahují další vlastnosti, které produkt IBM MQ rezervoval, ale ke kterým aplikacím nemají přístup. Názvy vyhrazených vlastností nejsou uvedeny v seznamu. Ve složkách vlastností usr, mq_usra sib_usr nejsou k dispozici žádné vyhrazené vlastnosti. Ale nevytvářejte vlastnosti s neplatnými názvy vlastností, viz “Neplatný název vlastnosti” na stránce 544.

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 516. 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>
JMSCorrelation	jms.Cid	string	<jms><Cid>*correlationId*</Cid></jms>

Tabulka 516. Název vlastnosti JMS, synonymum, datový typ a složka (pokračování)			
Synonymum vlastnosti	**Název vlastnosti**	**Datový typ**	**Složka**
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 517. 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

Produkt mq_usr obsahuje vlastnosti definované aplikací, které nejsou vystaveny jako uživatelem definované vlastnosti produktu JMS . Vlastnosti, které nesplňují požadavky produktu JMS , mohou být umístěny do této složky.

Vlastnosti můžete vytvořit ve složce mq_usr . Vlastnosti, které vytvoříte v produktu mq_usr , jsou stejně jako vlastnosti, které vytváříte v nových složkách s atributem content='properties' .

sib

Produkt `sib` obsahuje vlastnosti systémové zprávy sběrnice pro integraci služeb produktu WebSphere Application Server (WAS/SIB). Vlastnosti produktu `sib` nejsou vystaveny jako vlastnosti produktu JMS pro aplikace produktu IBM MQ JMS , protože se nejedná o podporované typy. Některé vlastnosti produktu `sib` například nelze vystavit jako vlastnosti produktu JMS , protože se jedná o bajtová pole. Některé vlastnosti produktu `sib` jsou vystaveny pro aplikace WAS/SIB jako vlastnosti produktu `JMS_IBM_*` ; tyto vlastnosti zahrnují vlastnosti dopředného a zpětného cesty směrování.

Nepřidávejte své vlastní vlastnosti do složky `sib`.

sib_context

`sib_context` obsahuje vlastnosti zprávy systému WAS/SIB, které nejsou vystaveny uživatelským aplikacím WAS/SIB nebo jako vlastnosti produktu JMS . `sib_context` obsahuje vlastnosti zabezpečení a transakcí, 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 uživatelské vlastnosti produktu JMS , protože nejsou podporované typy. Produkt `sib_usr` je vystaven aplikacím WAS/SIB v rozhraní produktu `SIMessage` . Viz Vývoj integrace služeb.

Typ vlastnosti `sib_usr` musí být `bin.hex`a hodnota musí být ve správném formátu. Pokud aplikace IBM MQ zapíše zadaný prvek `bin.hex` do složky v chybném formátu, obdrží aplikaci `IOException`. Pokud datový typ vlastnosti není `bin.hex` , aplikace přijme `ClassCastException`.

Nepokoušejte se zpřístupnit uživatelské vlastnosti produktu JMS pro 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, umístí se do složky `usr` .

Tabulka 518. `usr` - název vlastnosti, 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 s názvem 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 neseskupené vlastnosti

ibm

ibm obsahuje vlastnosti, které jsou používány pouze produktem IBM MQ.

Tabulka 519. 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 jsou používány pouze produktem IBM MQ.

Na vlastnosti ve složce mq se vztahují následující omezení:

- Vlastnosti MQse budou ignorovat pouze ve vlastnostech v první významné složce produktu mq ve zprávě; vlastnosti ve všech ostatních složkách produktu mq ve zprávě se budou ignorovat.
- Ve složce jsou povoleny pouze jednobajtové znaky UTF-8 . Vícebajtový znak ve složce může způsobit selhání syntaktické analýzy a zprávu, která má být odmítnuta.
- Nepoužívejte řídicí řetězce ve složce. S únikovým řetězcem se zachází jako se skutečnou hodnotou prvku.
- Jako bílý znak ve složce je považován pouze znak Unicode U+0020 . Všechny ostatní znaky jsou považovány za významné a mohou způsobit selhání syntaktické analýzy složky a zpráva, která má být odmítnuta.

Pokud selže syntaktická analýza složky mq , nebo pokud složka tato omezení nepozoruje, je zpráva odmítnuta s kódem příčiny 2527, MQRC_RFH_RESTRICTED_FORMAT_ERR.

Nepřidávejte své vlastní vlastnosti do složky mq.

mqema

mqema obsahuje vlastnosti, které jsou používány pouze produktem WebSphere Application Server. Složka byla nahrazena 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 520. mqext - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>

Tabulka 520. mqext - název vlastnosti, synonymum, datový typ a složka (pokračování)			
Synonymum vlastnosti	**Název vlastnosti**	**Datový typ**	**Složka**
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 521. 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>
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

Produkt mq_svc obsahuje vlastnosti používané produktem SupportPac MA93.

Nepřidávejte své vlastní vlastnosti do složky mq_svc.

mqtt

mqtt obsahuje vlastnosti používané produktem MQ Telemetry

Tabulka 522. `mqtt` - název vlastnosti, 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 s názvem `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 syntaktických diagramech popisuje "dvojice název-hodnota" obsah běžné složky. Obyčejná složka obsahuje skupiny a prvky. Prvek je dvojice název-hodnota. Skupina obsahuje prvky a další skupiny.

Pokud jde o stromy, prvky jsou listové uzly a skupiny jsou vnitřní uzly. Interní uzel a složka, která je kořenovým uzlem, mohou obsahovat kombinaci interních uzlů a koncových uzlů. Uzel nemůže být zároveň zároveň vnitřním uzlem a koncovým uzlem; viz Obrázek 2 na stránce 534.

Vlastnosti

V syntaktických diagramech popisuje "Vlastnosti" obsah složky vlastností. Složka vlastností obsahuje skupiny a vlastnosti. Vlastnost je dvojice názvu a hodnoty s volitelným atributem datového typu. Skupina obsahuje vlastnosti a další skupiny.

Pokud jde o stromy, vlastnosti jsou listové uzly a skupiny jsou vnitřní uzly. Interní uzel a složka vlastností, která je kořenovým uzlem, mohou obsahovat kombinaci interních uzlů a koncových uzlů. Uzel nemůže být zároveň zároveň vnitřním uzlem a koncovým uzlem; viz Obrázek 2 na stránce 534.

Vlastnost

Vlastnost zprávy je dvojice název-hodnota ve složce vlastností. Volitelně může obsahovat atribut datového typu a atribut vlastnosti; v tomto případě viz následující kód. Je-li atribut datového typu vynechán, typ vlastnosti je `string`.

```
<pf><p1 dt='i8'>value</p1></pf>
```

The name of a message property is its full path name, with the XML-like, <> syntax, replaced by dots. Například `myPropertyFolder1.myGroup1.myGroup2.myProperty1` je mapováno 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 produktu Obrázek 7 na stránce 541 jsou mapovány na složku vlastností v produktu Obrázek 8 na stránce 541 .

```
 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 více názvů vlastností

Název

Název musí začínat písmenem *Letter* nebo *Underscore*. Nesmí obsahovat *dvojtečku*, nesmí končit v *období* a obsahovat pouze *Písmena*, *Číslice*, *Podtržítka*, *Pomlčka*a *Dots*. Platné znaky jsou definovány ve specifikaci XML W3C a skládají se hlavně z kategorií Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` a Nd. Viz Znakové kategorie Unicode.

Úplná cesta k vlastnosti nebo dvojici název-hodnota nesmí porušit pravidlo popsané v “Neplatný název cesty” na stránce 543. 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 lze umístit do kořenové složky složky. Použijte skupiny, pokud pomáhá organizovat vlastnosti a dvojice název-hodnota.

Název prvku

Název prvku má stejnou syntaxi jako název.

Hodnota prvku

Hodnota prvku zahrnuje veškerý bílý prostor mezi značkou <*Element name*> a značkou </*Element name*> . Nepoužívejte dva znaky < a & v hodnotě. Poté nahraďte produkty < a `&`.

Atributy vlastností

Pole vlastností mapují pole deskriptoru vlastností. Mapování jsou následující:

Podpora

sa (výchozí)
`MQPD_SUPPORT_OPTIONAL`

sr
`MQPD_SUPPORT_REQUIRED`

sx
MQPD_SUPPORT_REQUIRED_IF_LOCAL

Kontext

žádná (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 parametr `default` v kombinaci s dalšími volbami. Hodnota `default` je stejná jako `forward` + `report` + `publish`.

Není
MQPD_COPY_NONE

Nepoužívejte žádné v kombinaci s dalšími volbami.

Atributy vlastností Podpora jsou použitelné pouze pro vlastnosti ve složce `mq` .

Atributy vlastností Kontext a CopyOptions jsou použitelné pro všechny složky vlastností.

Datové typy

Datové typy produktu MQRFH2 jsou mapovány na typy vlastností zpráv takto:

Tabulka 523. 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[]

Jakýkoli prvek bez datového typu se předpokládá, že je typu `string`.

Hodnota null je indikována atributem prvku `xsi:nil='true'`. Nepoužívejte atribut `xsi:nil='false'` pro jiné hodnoty než null. Následující vlastnost má například hodnotu null:

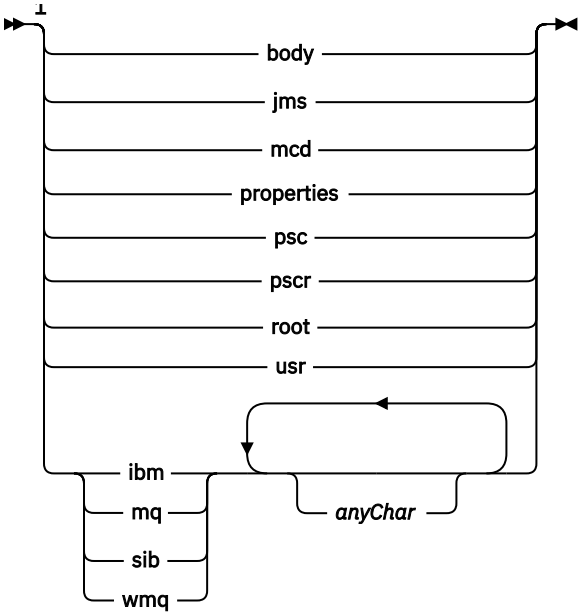
```
<NullProperty xsi:nil='true'></NullProperty>
```

Vlastnost typu byte nebo znakový řetězec může mít prázdnou hodnotu. Prázdná hodnota je reprezentována prvkem MQRFH2 s hodnotou prvku nulové délky. Např. následující vlastnost má prázdnou hodnotu:

```
<EmptyProperty></EmptyProperty>
```

Vyhrazená složka nebo název složky vlastností

Omezte název složky nebo složky vlastností tak, aby se nezačínala libovolným z následujících řetězců. Předpony jsou vyhrazeny pro názvy složek nebo vlastností vytvořených produktem IBM.

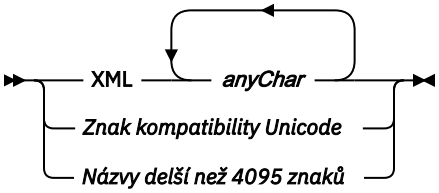


Poznámky:

[1] Vyhrazená složka nebo název vlastnosti obsahuje libovolnou kombinaci malých a velkých písmen.

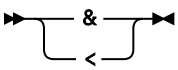
Neplatný název cesty

Omezte úplnou cestu dvojice název-hodnota nebo vlastnosti nezahrnujte některý z následujících řetězců.



Neplatné znaky

Vždy používejte esc sekvence & a < místo literálů "&" a "<".

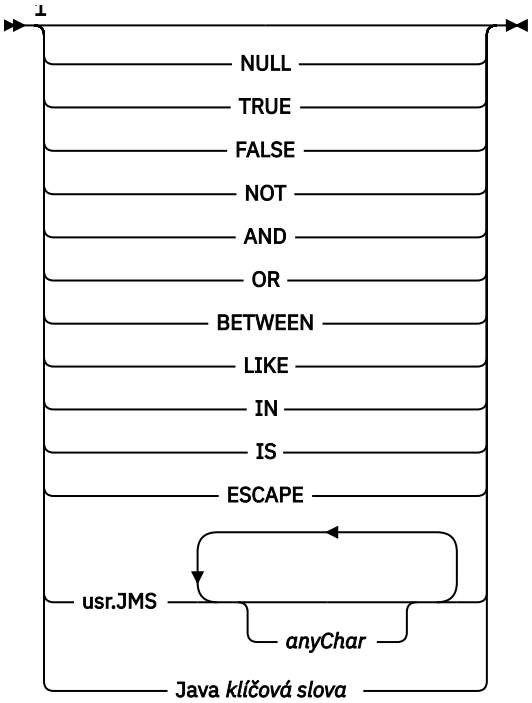


Definované názvy vlastností

Definované názvy vlastností jsou názvy vlastností, které jsou definovány produktem IBM MQnebo jinými produkty a které používají aplikace IBM MQ a uživatelské aplikace. Definované vlastnosti existují pouze v definovaných složkách vlastností. Definované názvy vlastností jsou popsány v popisech složek vlastností; viz Složky vlastností.

Neplatný název vlastnosti

Nevytvářejte názvy vlastností, které se shodují s následujícím pravidlem. Pravidlo se vztahuje na úplnou cestu k vlastnosti, která pojmenovává vlastnost, a nikoli pouze názvu prvku vlastnosti.



Poznámky:

[1] Neplatný název vlastnosti může obsahovat libovolnou kombinaci malých a velkých písmen.

Neplatné atributy

Složky vlastností a vlastnosti mohou zahrnovat pouze podporované “Atributy vlastností” na stránce 541 a “Datové typy” na stránce 542.

Jakékoli nepodporované atributy podobné XML, například názvy s hodnotami řetězce v uvozovkách, které jsou obsaženy ve složkách vlastností nebo ve vlastnostech, mohou být odebrány.

Atributy podobné formátu XML, které jsou obsaženy ve složkách mimo vlastnost nebo v prvcích mimo vlastnost, které zůstávají v záhlaví 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).

Pouľití

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 524. Pole v MQRMH pro MQRMH

Název a popis pole	**Název konstanty**	**Počáteční hodnota konstanty (pokud existuje)**
StrucId (identifikátor struktury)	MQRMH_STRUC_ID	'RMH¬'
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

Tabulka 524. Pole v MQRMH pro MQRMH (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.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)

Jedná se o identifikátor struktury; hodnota musí být:

MQRMH_STRUCTION_ID
Identifikátor struktury záhlaví zprávy odkazu.

Pro programovací jazyk C je také definována konstanta MQRMH_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQRMH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQRMH_STRUC_ID.

Verze (MQLONG)

Číslo verze struktury. Hodnota musí být:

MQRMH_VERSION_1
Struktura záhlaví referenční zprávy Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

MQRMH_AKTUÁLNÍ_VERZE
Aktuální verze struktury záhlaví zprávy odkazu.

Počáteční hodnota tohoto pole je MQRMH_VERSION_1.

StrucLength (MQLONG)
Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadná data.

Počáteční hodnota tohoto pole je nula.

Kódování (MQLONG)

Určuje číselné kódování hromadných dat. Nevztahuje se na číselná data v samotné struktuře MQRMH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

CodedCharSetId (MQLONG)

Tato hodnota určuje identifikátor znakové sady pro hromadný data. Nevztahuje se na znaková data v samotné struktuře MQRMH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech po této struktuře jsou ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Nepoužívejte MQCCSI_INHERIT, je-li hodnota pole `PutApplType` v deskriptoru 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)

Uvádí název formátu hromadných dat.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole *`Format`* v produktu MQMD.

Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

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 nebo nereprezentuje poslední část objektu. Servisní dokumentace programu MQRMHF_NOT_LAST. Není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je MQRMHF_NOT_LAST.

ObjectType (MQCHAR8)

Jedná se o název, který může uživatelská procedura pro zprávy použít k rozeznání typů referenční zprávy, které podporuje. Název se musí shodovat se stejnými pravidly jako pole *Format* , viz “Formát (MQCHAR8)” na stránce 550.

Počáteční hodnota tohoto pole je 8 mezer.

ID ObjectInstance(MQBYTE24)

Toto pole použijte k identifikaci určité instance objektu. Pokud není potřeba, nastavte ji na následující hodnotu:

MQOII_NONE
Není uveden žádný identifikátor instance objektu. Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQOII_NONE_ARRAY; hodnota má stejnou hodnotu jako MQOII_NONE, ale je to pole znaků namísto řetězce.

Délka tohoto pole je dána hodnotou MQ_OBJECT_INSTANCE_INSTANCE_LENGTH. Počáteční hodnota tohoto pole je MQOII_NONE.

Délka SrcEnv(MQLONG)

Délka dat zdrojového prostředí. Je-li toto pole nula, nejsou žádná data o zdrojovém prostředí a *SrcEnvOffset* je ignorován.

Počáteční hodnota tohoto pole je 0.

Odchylka SrcEnv(MQLONG)

Toto pole určuje posun dat o zdrojovém prostředí ze začátku struktury MQRMH. Data o zdrojovém prostředí mohou být určena tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Například na Windows mohou být data zdrojového prostředí cestou k adresáři objektu obsahujícího data typu bulk. Pokud však tvůrce neznají data o zdrojovém prostředí, musí uživatelská procedura pro předání zprávy určit všechny potřebné informace o prostředí.

Délka dat zdrojového prostředí je dána *SrcEnvLength* ; Pokud je tato délka nula, nejsou žádná data o zdrojovém prostředí a *SrcEnvOffset* je ignorován. Je-li tato možnost přítomna, musí se zdrojová data prostředí zcela nacházet v rozmezí *StrucLength* bajtů od začátku struktury.

Aplikace nesmí předpokládat, že data prostředí jsou spuštěna okamžitě po posledním pevném poli ve struktuře nebo že je souvislá s libovolnými daty adresovaným poli *SrcNameOffset*, *DestEnvOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

Délka SrcName(MQLONG)

Délka názvu zdrojového objektu. Je-li toto pole nula, neexistuje žádné jméno zdrojového objektu a *SrcNameOffset* se ignoruje.

Počáteční hodnota tohoto pole je 0.

Posunutí SrcName(MQLONG)

Toto pole určuje posun názvu zdrojového objektu od začátku struktury MQRMH. Jméno zdrojového objektu může být zadáno tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Pokud však tvůrce neznají název zdrojového objektu, musí uživatelská procedura zprávy identifikovat objekt, ke kterému má být přístup.

Délka názvu zdrojového objektu je dána *SrcNameLength* ; je-li tato délka nula, neexistuje žádné jméno zdrojového objektu a *SrcNameOffset* je ignorován. Je-li tento název zadán, musí být název zdrojového objektu zcela umístěn v rozmezí *StrucLength* bajtů od začátku struktury.

Aplikace nesmí předpokládat, že název zdrojového objektu je souvislý s libovolní z dat adresovaných poli *SrcEnvOffset*, *DestEnvOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

DestEnvDélka (MQLONG)

Jedná se o délku dat cílového prostředí. Je-li toto pole nula, nejsou k dispozici žádná data cílového prostředí a *DestEnvOffset* je ignorován.

Offset DestEnv(MQLONG)

Toto pole určuje posun dat cílového prostředí ze začátku struktury MQRMH. Data cílového prostředí mohou být uvedena tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Například, na Windows data cílového prostředí mohou být cesta k adresáři objektu, kam se mají hromadná data uložit. Pokud však tvůrce neznáme data cílového prostředí, je zodpovědností uživatelského ukončovacího programu pro zprávy, aby určil, že jsou potřebné informace o prostředí.

Délka dat cílového prostředí je dána produktem *DestEnvLength* ; je-li tato délka nula, nejsou žádná data cílového prostředí a *DestEnvOffset* je ignorován. Je-li tento parametr zadán, musí být data cílového prostředí plně umístěna v rozmezí *StrucLength* bajtů od začátku struktury.

Aplikace nesmí předpokládat, že data cílového prostředí sousedí s libovolní z dat řešených poli *SrcEnvOffset*, *SrcNameOffset*a *DestNameOffset* .

Počáteční hodnota tohoto pole je 0.

DestName(MQLONG)

Délka názvu cílového objektu. Je-li toto pole nula, neexistuje žádný název cílového objektu a *DestNameOffset* je ignorován.

Offset DestName(MQLONG)

Toto pole určuje posun názvu cílového objektu od začátku struktury MQRMH. Jméno cílového objektu může být zadáno tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Pokud však tvůrce nezná název cílového objektu, zodpovídá za identifikaci objektu, který má být vytvořen nebo upraven, je zodpovědný za uživatelskou proceduru zprávy.

Délka názvu cílového objektu je dána *DestNameLength* ; je-li tato délka nula, neexistuje žádný název cílového objektu a *DestNameOffset* je ignorován. Je-li tento parametr zadán, musí být název cílového objektu zcela umístěn v rozmezí *StrucLength* bajtů od začátku struktury.

Aplikace nesmí předpokládat, že název cílového objektu je souvislý s libovolní z dat adresovaných poli *SrcEnvOffset*, *SrcNameOffset*a *DestEnvOffset* .

Počáteční hodnota tohoto pole je 0.

Délka DataLogical(MQLONG)

Pole *DataLogicalLength* určuje délku hromadného dat, na kterou odkazuje struktura MQRMH.

Pokud jsou hromadná data ve skutečnosti přítomna ve zprávě, data začínají na offset *StrucLength* bajtů od začátku struktury MQRMH. Délka celé zprávy minus *StrucLength* udává délku hromadného datového souboru.

Pokud jsou data přítomna ve zprávě, *DataLogicalLength* uvádí množství dat, která jsou relevantní. Normální případ je určen pro *DataLogicalLength* , aby měl stejnou hodnotu jako délka dat přítomných ve zprávě.

Pokud struktura MQRMH představuje zbývající data v objektu (počínaje určeným logickým posunutím), můžete použít hodnotu nula pro *DataLogicalLength*, pokud se hromadná data ve skutečnosti ve zprávě neprezentují.

Nejsou-li k dispozici žádná data, je konec MQRMH totožný s koncem zprávy.

Počáteční hodnota tohoto pole je 0.

Ofset DataLogicalOffset (MQLONG)

Toto pole určuje dolní posun dat hromadného objektu od začátku objektu, jehož součástí jsou hromadné datové formuláře. Posunutí hromadných dat od začátku objektu se nazývá *logický posun*. Toto není fyzický posun hromadných dat od začátku struktury MQRMH; tento posun je dán parametrem *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 offsetu o 1 000 000 000. Je to tedy hodnota v rozsahu od 0 do 999 999 999.
- *DataLogicalOffset2* představuje výsledek, který se získá, když je logický offset rozdělen do 1 000 000 000. Jedná se tedy o počet úplných násobků 1 000 000 000, které existují v logickém posunu. Počet násobků je v rozsahu od 0 do 999 999 999.

Počáteční hodnota tohoto pole je 0.

DataLogicalOffset2 (MQLONG)

Toto pole určuje horní posun hromadných dat od začátku objektu, jehož součástí jsou hromadné datové formuláře. Je to hodnota v rozsahu od 0 do 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 525. 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)

Jedná se o kód dokončení, který je výsledkem operace otevření nebo vložení pro frontu s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je MQCC_OK.

Příčina (MQLONG)

Jedná se o kód příčiny, který je výsledkem operace otevření nebo vložení pro frontu s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci 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

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 526. 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

Tabulka 526. 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_2.		
KeyResetPočet (počet resetů tajného klíče TLS)	MQSCO_RESET_COUNT_DEFAULT	0
"FipsRequired (MQLONG)" na stránce 560 (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
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

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 */

  };
```

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 **
```

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 */
```

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 315

Struktura MQCNO umožňuje aplikaci určit volby související s připojením ke správci front. Struktura je vstupní/výstupní parametr volání MQCONNX.

StrucId (MQCHAR4)

Jedná se o identifikátor struktury; hodnota musí být:

ID_KONSTRUKCE_MQSCO_
Identifikátor struktury voleb konfigurace TLS.

Pro programovací jazyk C je také definován konstantní MQSCO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQSCO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSCO_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQSCO_VERSION_1
Struktura konfiguračních voleb TLS Version-1 TLS.

MQSCO_VERSION_2
Struktura konfiguračních voleb Version-2 TLS.

MQSCO_VERSION_3
Struktura konfiguračních voleb TLS Version-3 .

MQSCO_VERSION_4
Struktura konfiguračních voleb Version-4 TLS.

MQSCO_VERSION_5
Struktura konfiguračních voleb TLS Version-5 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSCO_
Aktuální verze struktury voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSCO_VERSION_1.

ALW *KeyRepository (MQCHAR256)*

Toto pole je relevantní pouze pro produkt IBM MQ MQI clients spuštěný na systémech AIX, Linux, and Windows . Určuje umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty. Soubor databáze klíčů musí mít název souboru ve tvaru `zzz.kdb`, kde `zzz` je vybratelný uživatelem. Pole *KeyRepository* obsahuje cestu k tomuto souboru, spolu s názvem souboru stem (všechny znaky v názvu souboru, ale ne včetně konečné `.kdb`). Přípona souboru `.kdb` se přidá automaticky.

Ke každému souboru databáze klíčů je přidružen *soubor stash hesel*. Tato zadržuje kódovaná hesla, která umožňují programový přístup k databázi klíčů. Soubor pro uložení hesla se musí nacházet ve stejném adresáři a musí mít stejný soubor jako databáze klíčů a musí končit příponou `.sth`.

Pokud má pole *KeyRepository* například hodnotu `/xxx/yyy/key`, musí být soubor databáze klíčů `/xxx/yyy/key.kdba` soubor pro uložení hesla musí být `/xxx/yyy/key.sth`, kde `xxx` a `yyy` představují názvy adresářů.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Hodnota není kontrolována; pokud došlo k chybě při přístupu k úložišti klíčů, volání selže s kódem příčiny MQRC_KEY_REPOSITORY_ERROR.

Chcete-li spustit TLS připojení 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 řetězec s hodnotou null v jazyce C a prázdné znaky v jiných programovacích jazycích.

CryptoHardware (MQCHAR256)

Toto pole poskytuje podrobnosti konfigurace kryptografického hardwaru připojeného k systému klienta.

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 PKCS #11 , například IBM 4960 nebo IBM 4764, musí být uvedena cesta k ovladači PKCS #11 , návěští tokenu PKCS #11 a řetězce hesel tokenu PKCS #11 , přičemž každý z nich končí 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
```

Návěští tokenu PKCS #11 se musí shodovat s popiskem, se kterým jste nakonfigurovali hardware.

Není-li požadována žádná konfigurace kryptografického hardwaru, nastavte pole na prázdné nebo null.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Pokud hodnota není platná, nebo vede k selhání při konfiguraci kryptografického hardwaru, volání selže s kódem příčiny MQRC_CRYPTO_HARDWARE_ERROR.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_SSL_TYPTO_HARDWARE_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v jazyce C a prázdné znaky v jiných programovacích jazycích.

AuthInfoRecCount (MQLONG)

Jedná se o počet záznamů ověřovacích informací (MQAIR) adresovaných poli *AuthInfoRecPtr* nebo *AuthInfoRecOffset* . Další informace viz"MQAIR-záznam ověřovacích informací" na stránce 269. Hodnota musí být nula nebo větší. Není-li hodnota platná, volání selže s kódem příčiny MQRC_AUTH_INFO_INFO_COUNT_ERROR.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AuthInfoRecOffset (MQLONG)

Jedná se o posun v bajtech prvního záznamu ověřovacích informací od začátku struktury MQSCO. Odsazení může být kladné nebo záporné. Pole je ignorováno, pokud *AuthInfoRecCount* je nula.

K zadání záznamů MQAIR můžete použít buď *AuthInfoRecOffset* nebo *AuthInfoRecPtr* , ale ne obojí; podrobnosti najdete v popisu pole *AuthInfoRecPtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AuthInfoRecPtr (PMQAIR)

Toto je adresa prvního záznamu ověřovacích informací. Pole je ignorováno, pokud *AuthInfoRecCount* je nula.

Pole záznamů MQAIR můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *AuthInfoRecPtr*

  V takovém případě může aplikace deklarovat pole záznamů MQAIR, které jsou odděleny od struktury MQSCO, a nastavit proměnnou *AuthInfoRecPtr* na adresu pole.

  Zvažte použití *AuthInfoRecPtr* pro programovací jazyky, které podporují datový typ ukazatele v módě, který je přenosný do různých prostředí (například programovací jazyk C).
- Použití pole offsetu *AuthInfoRecOffset*

  V takovém případě musí aplikace deklarovat složenou strukturu obsahující MQSCO, za kterou následuje pole záznamů MQAIR, a nastavit proměnnou *AuthInfoRecOffset* na hodnotu offsetu prvního záznamu v poli od začátku struktury MQSCO. Ujistěte se, že je tato hodnota správná a že má hodnotu, která může být umístěna v rámci MQLONG (nejvíce omezující programovací jazyk je COBOL, pro který je platný rozsah -999 999 999 až +999 999 999).

  Zvažte použití *AuthInfoRecOffset* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele v módě, který není přenosný do různých prostředí (například programovací jazyk COBOL).

Ať už vyberete jakoukoli techniku, lze použít pouze jedno z *AuthInfoRecPtr* a *AuthInfoRecOffset* ; volání selže s kódem příčiny MQRC_AUTH_INFO_INFO_ERROR, pokud jsou oba nenulová.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null.

Poznámka: Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Počet KeyReset(MQLONG)

Představuje celkový počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než bude znovu vyjednán tajný klíč.

Počet bajtů zahrnuje řídicí informace odeslané agentem MCA.

Určíte-li počet obnovení tajných klíčů TLS v rozsahu od 1 bajtu do 32 KB, budou kanály TLS používat počet obnovení tajných klíčů 32 KB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se mohly vyskytnout u malých hodnot resetu tajného klíče TLS.

Toto je vstupní pole. Hodnota je číslo v rozsahu od 0 do 999 999 999, přičemž výchozí hodnota je 0. Použijte hodnotu 0, abyste označili, že tajné klíče nejsou nikdy znovu vyjednávány.

FipsRequired (MQLONG)

Produkt IBM MQ lze konfigurovat s šifrovacím hardwarem tak, aby používané šifrovací moduly byly moduly poskytované hardwarovým produktem. 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" . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C certificate a měli by si být vědomi jakýchkoli 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: Je-li to možné, pokud jsou konfigurovány pouze specifikace CipherSpecs standardu FIPS, klient MQI odmítne připojení, která určují jinou specifikaci než FIPS CipherSpec s hodnotou 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)

Toto pole Uvádí, zda se použije šifrování vyhovující Suite B a jaká úroveň síly je použita. Hodnota může být jedna nebo více hodnot:

- MQ_SUITE_B_NONE

  Šifrování kompatibilní se sadou Suite B se nepoužívá.
- MQ_SUITE_B_128_BIT

  Používá se zabezpečení odolnosti standardu Suite B 128 bitů.
- MQ_SUITE_B_192_BIT

  Je použito 192bitové zabezpečení pevnosti sady Suite B.

Poznámka: Použití hodnoty MQ_SUITE_B_NONE s jakoukoli jinou hodnotou v tomto poli je neplatné.

Zásada CertificateVal(MQLONG)

Toto pole uvádí, jaký typ zásady ověření certifikátu se použije. Pole může být nastaveno na jednu z následujících hodnot:

MQ_CERT_VAL_POLICY_ANY
Použít všechny zásady ověření platnosti certifikátů podporované knihovnou SSL (Secure Sockets Layer). Přijměte řetěz certifikátů, pokud některý ze zásad považuje řetězec certifikátů za platný.

MQ_CERT_VAL_POLICY_RFC5280
Použijte pouze zásadu ověření certifikátu vyhovujícího standardu RFC5280 . Toto nastavení poskytuje přísnější validaci než nastavení ANY, ale odmítá některé starší digitální certifikáty.

Počáteční hodnota tohoto pole je MQ_CERT_VAL_POLICY_ANY.

CertificateLabel (MQCHAR64)
Toto pole uvádí podrobnosti o použité návěští certifikátu.

IBM MQ inicializuje výchozí hodnotu pro pole *CertificateLabel* jako prázdné místo.

To je interpretováno za běhu jako výchozí hodnota a je zpětně kompatibilní.

Například při určení verze MQSCO menší než 5.0nebo použití výchozí hodnoty mezer pro pole *CertificateLabel* použije předexistující výchozí hodnota `ibmwebspheremq`*`user_id`*.

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
- IBM i
- Linux
- Windows
- 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
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

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.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)

Jedná se o identifikátor struktury; hodnota musí být:

ID_STRUKTURY OBJEKTU MQSD_STRUCT
Identifikátor struktury deskriptoru odběru.

Pro programovací jazyk C je také definována konstanta MQSD_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQSD_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSD_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQSD_VERSION_1
Struktura deskriptoru odběru Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSD_AKTUÁLNÍ_VERZE
Aktuální verze struktury deskriptoru odběru.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSD_VERSION_1.

Volby (MQLONG)

Tato volba poskytuje volby pro řízení akce volání MQSUB.

Je třeba určit alespoň jednu z následujících voleb:

- MQSO_ALTER
- MQSO_RESUME
- VYTVOŘENÉ MQSO_CREATE

Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace, které nejsou platné, jsou uvedeny v tomto tématu; všechny ostatní kombinace jsou platné.

Volby přístupu nebo vytvoření: Volby přístupu a vytvoření řídí, zda je odběr vytvořen, nebo zda je vrácen nebo změněn existující odběr. Musíte uvést alespoň jednu z těchto voleb.

Tabulka 527. Platné kombinace voleb přístupu a vytvoření

Kombinace možností	**Notes**
VYTVOŘENÉ MQSO_CREATE	Vytvoří odběr, pokud takový odběr neexistuje. Tato kombinace selže, pokud existuje odběr.
MQSO_RESUME	Pokračuje ve stávajícím odběru. Tato kombinace selže, pokud neexistuje žádný odběr.
MQSO_CREATE + MQSO_RESUME	Vytvoří odběr, pokud jeden neexistuje a obnoví odpovídající, pokud existuje. Tato kombinace je užitečná, pokud se používá v aplikaci, která je spouštěna určitý počet opakování.
MQSO_ALTER (viz poznámka)	Pokračuje ve stávajícím odběru a mění všechna pole tak, aby se shodovala s odpovídajícími poli specifikovanou v rámci MQSD. Tato kombinace selže, pokud neexistuje žádný odběr.
MQSO_CREATE + MQSO_ALTER (viz poznámka)	Vytvoří odběr, pokud neexistuje, a obnoví odpovídající, pokud existuje, tím, že pozmění všechna pole, která mají odpovídat hodnotě zadané v MQSD. Tato kombinace je užitečná kombinace, je-li použita v aplikaci, která chce zajistit, aby její odběr byl v určitém stavu, než budete pokračovat.

Poznámka:

Volby určené parametrem MQSO_ALTER mohou také určovat MQSO_RESUME, ale tato kombinace nemá žádný další účinek při specifikaci samotného MQSO_ALTER. MQSO_ALTER znamená MQSO_RESUME, protože volání funkce MQSUB pro změnu odběru znamená, že odběr bude také obnoven. Opak není pravda, nicméně: obnovení odběru neznamená, že je třeba jej změnit.

VYTVOŘENÉ MQSO_CREATE

Vytvořte nový odběr pro určené téma. Existuje-li odběr s použitím stejného produktu *SubName* , volání selže s funkcí MQRC_SUB_ALREADY_EXISTS. Toto selhání lze předejít kombinací volby MQSO_CREATE s MQSO_RESUME. *SubName* není vždy nutné. Další informace najdete v popisu tohoto pole.

Kombinace MQSO_CREATE s MQSO_RESUME vrátí popisovač do již existujícího odběru pro zadaný *SubName* , pokud je nalezen; pokud neexistuje žádný existující odběr, vytvoří se nový pomocí všech polí poskytnutých v MQSD.

MQSO_CREATE lze také kombinovat s příkazem MQSO_ALTER s podobným účinkem.

MQSO_RESUME

Vraťte popisovač na již existující odběr, který odpovídá určenému názvu produktu *SubName*. Nebyly provedeny žádné změny odpovídajících atributů odběrů a jsou vraceny ve výstupu ve struktuře MQSD. Jsou použita pouze následující pole MQSD: StrucId, Verze, Volby, AlternateUserID a AlternateSecurityID a SubName.

Volání selže s kódem příčiny MQRC_NO_SUBSCRIPTION, pokud odběr neexistuje odpovídající úplnému názvu odběru. Toto selhání lze předejít kombinací volby MQSO_CREATE s 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 uživatelů, je ID alternativního uživatele zaznamenáno jako ID uživatele, které vytvořil odběr namísto ID uživatele, pod kterým byl odběr proveden.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby MQSO_ANY_USERID a ID uživatele odběru se liší od ID aplikace, která požaduje zpracování na odběru, volání selže s kódem příčiny MQRC_IDENTITY_MISMATCH.

Pokud existuje odpovídající odběr a v současné době se používá, volání selže s klauzulí MQRC_SUBSCRIPTION_IN_USE.

Pokud odběr uvedený v položce SubName není platným odběrem pro pokračování nebo úpravu z aplikace, volání selže s položkou MQRC_INVALID_SUBSCRIPTION.

MQSO_RESUME je odvozeno příkazem MQSO_ALTER, takže jej není třeba kombinovat s touto volbou. Kombinování těchto dvou možností však nezpůsobí chybu.

MQSO_ALTER

Vrátit popisovač na již existující odběr s úplným názvem odběru, který odpovídá názvu zadanému názvem v produktu *SubName*. Všechny atributy odběru, které se liší od všech atributů uvedených ve struktuře 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. Pokusíte-li se změnit atribut, který nelze změnit, nebo chcete-li 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 odběr odpovídající úplnému názvu odběru neexistuje. Tomuto selhání se můžete vyhnout kombinací volby MQSO_CREATE s parametrem MQSO_ALTER.

Kombinace MQSO_CREATE s MQSO_ALTER vrací popisovač do již existujícího odběru pro zadaný *SubName* , pokud je nalezen; pokud neexistuje žádný existující odběr, vytvoří se nový pomocí všech polí poskytnutých v rámci MQSD.

ID uživatele odběru je ID uživatele, který vytvořil odběr, nebo pokud je později změněn jiným ID uživatele, jedná se o ID uživatele, který je nejnovější, úspěšnou změnou. Je-li použit identifikátor AlternateUsera pro tohoto uživatele je povoleno použití alternativních ID uživatelů, je ID alternativního

uživatele zaznamenáno jako ID uživatele, které vytvořil odběr namísto ID uživatele, pod kterým byl odběr proveden.

Pokud existuje odpovídající odběr, který byl vytvořen bez volby MQSO_ANY_USERID a ID uživatele odběru se liší od ID aplikace, která požaduje zpracování na odběru, volání selže s kódem příčiny MQRC_IDENTITY_MISMATCH.

Pokud existuje odpovídající odběr a v současné době se používá, volání selže s klauzulí MQRC_SUBSCRIPTION_IN_USE.

Pokud odběr uvedený v položce SubName není platným odběrem pro pokračování nebo úpravu z aplikace, volání selže s položkou MQRC_INVALID_SUBSCRIPTION.

Následující tabulka zobrazuje schopnost MQSO_ALTER změnit hodnoty atributu v MQSD a MQSUB.

Tabulka 528. Atributy v MQSD a MQSUB, které mohou být pozměněny

Deskriptor datového typu nebo volání funkce	Název pole	Může být tento atribut změněn pomocí MQSO_ALTER	Kód příčiny
MQSD.	Volby životnosti	Ne	MQRC_DURABILITY_NOT_ALTERABLE
MQSD.	Volby cíle	Ano	Není
MQSD.	Volby registrace	Ano (viz poznámka "1" na stránce 569)	MQRC_GROUPING_NOT_ALTERABLE, pokusíte-li se změnit MQSO_GROUP_SUB
MQSD.	Volby publikování	Ano (viz poznámka "2" na stránce 569)	Není
MQSD.	Volby zástupného znaku	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	Další volby	Ne (viz poznámka "3" na stránce 569)	Není
MQSD.	ObjectName	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	AlternateUserid	Ne (viz poznámka "4" na stránce 569)	Není
MQSD.	AlternateSecurityId	Ne (viz poznámka "4" na stránce 569)	Není
MQSD.	SubExpiry	Ano	Není
MQSD.	ObjectString	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	SubName	Ne (viz poznámka "5" na stránce 569)	Není
MQSD.	SubUserData	Ano	Není
MQSD.	SubCorrelId	Ano (viz poznámka "6" na stránce 569)	Funkce MQRC_GROUPING_NOT_ALTERABLE v rámci seskupeného odběru
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 569)	Funkce MQRC_GROUPING_NOT_ALTERABLE v rámci seskupeného odběru

Notes:

1. Objekt MQSO_GROUP_SUB nelze změnit.
2. Objekt 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 odebírané odběru
6. S výjimkou, je-li součástí seskupeného podobjektu (MQSO_GROUP_SUB)

Volby trvanlivosti: Následující volby řídí, jak trvalý odběr je. 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 funkce MQSO_RESUME je nastavena příslušná volba trvanlivosti.

MQSO_TRVALKA
Požadavek na odběr tohoto tématu zůstane zachován, dokud nebude explicitně odebrán pomocí funkce MQCLOSE s volbou MQCO_REMOVE_SUB. Není-li tento odběr explicitně odebrán, zůstane i po zavření tohoto připojení aplikací ke správci front.

Je-li požadován trvalý odběr tématu, které je definováno jako nepovolení trvalých odběrů, volání selže při volání MQRC_DURABILITY_NOT_ALLOWED.

MQSO_NON_DURABLE
Pokud je připojení aplikací ke správci front ukončeno, je požadavek na odběr tohoto tématu odebrán, pokud již není explicitně odebrán. MQSO_NON_DURABLE je protilehlý k volbě MQSO_DURABLE a je definován pro dokumentaci programu. Je-li uveden žádný, je to výchozí nastavení.

Volby cíle: Následující volba určuje cíl, do kterého jsou odesílána publikování pro téma, k jehož odběru je odebírán odběr. Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze změnit místo určení použité pro publikování pro odběr. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena, je-li to vhodné.

SPRAVOVANÉ MQSO_MANAGED

Požadujte, aby bylo místo určení, kam jsou publikace odesílány, spravováno správcem front.

Popisovač objektu vrácený v produktu *Hobj* představuje spravovanou frontu správce front a je určen pro použití s následujícími voláními MQGET, MQCB, MQINQ nebo MQCLOSE.

Ovladač objektu vrácený z předchozího volání MQSUB nemůže být zadán v parametru **Hobj** , pokud není zadán parametr MQSO_MANAGED.

MQSO_NO_MULTICAST

Požadavek na to, aby místo určení, kam jsou publikace odesílány, není skupinová adresa výběrového vysílání. Tato volba je platná pouze v kombinaci s volbou MQSO_MANAGED. Je-li v parametru **Hobj** poskytnuta obsluha pro frontu, nelze pro tento odběr použít výběrové vysílání a volba není platná.

Je-li téma definováno pouze pro povolení výběrového vysílání pomocí nastavení `MCAST(ONLY)` , pak se volání nezdaří s kódem příčiny MQRC_MULTICAST_REQUIRED.

Volba rozsahu platnosti: Následující volba určuje rozsah odběru, který má být proveden. Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze tuto volbu rozsahu odběru změnit. Při návratu z volání MQSUB pomocí MQSO-RESUME je nastavena příslušná volba rozsahu.

MQSO_SCOPE_QMGR
Tento odběr je proveden pouze v lokálním správci front. Do jiných správců front v síti není distribuován žádný odběr serveru proxy. K tomuto odběrateli jsou odeslány pouze publikování, která byla publikována v tomto správci front. Tím je potlačeno jakékoli chování nastavené pomocí atributu tématu SUBSCOPE.

Poznámka: Pokud není nastavena, je rozsah odběru určen atributem tématu SUBSCOPE.

Volby registrace: Následující volby řídí podrobnosti o registraci, která se provádí ve správci front pro tento odběr. Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze tyto volby registrace změnit. Při návratu z volání MQSUB pomocí funkce MQSO_RESUME jsou nastaveny příslušné volby registrace.

MQSO_GROUP_SUB

Tento odběr má být seskupen s jinými odběry stejné SubLevel pomocí stejné fronty a s uvedením stejného ID korelace, aby všechny publikace k tématům, které by způsobily více než jednu zprávu publikování, byly poskytnuty do skupiny odběrů kvůli překrývající se sadě používaných řetězců témat, způsobí, že bude do fronty doručena pouze jedna zpráva. Není-li tato volba použita, bude každý jedinečný odběr (identifikován názvem SubName) poskytnut spolu s kopií publikování, což může znamenat více než jednu kopii publikování, která může být umístěna do fronty sdílené počtem odběrů.

Pouze nejdůležitější předplatné ve skupině je poskytnuto spolu s kopií publikace. Nejvýznamnější odběr je založen na úplném názvu tématu až po bod, ve kterém je nalezen zástupný znak. Je-li ve skupině použita směs zástupných systémů, je důležitá pouze pozice zástupného znaku. Doporučuje se nekombinovat různé schéma zástupných znaků v rámci skupiny 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 určuje MQSO_NOT_OWN_PUBS a jedná se o publikování ze stejné aplikace, nebude do fronty doručeno žádné publikování.

Při změně odběru provedené s touto volbou pole, která implikují seskupení, Hobj na volání MQSUB (reprezentující frontu a název správce front) a ID SubCorrelnelze změnit. Pokus o změnu způsobí, že volání selže s MQRC_GROUPING_NOT_ALTERABLE.

Tato volba musí být kombinovaná s parametrem MQSO_SET_CORREL_ID s ID SubCorrel, která není nastavena na hodnotu MQCI_NONE, a nelze ji kombinovat s parametrem MQSO_MANAGED.

MQSO_ANY_USERID

Je-li zadáno MQSO_ANY_USERID, identita odběratele není omezena pouze na jedno ID uživatele. To umožňuje jakémukoli uživateli změnit nebo obnovit odběr, když mají odpovídající oprávnění. Pouze jeden uživatel může mít odběr v jednom okamžiku. Pokus o obnovení použití odběru, který je aktuálně používán jinou aplikací, způsobí, že volání selže při volání MQRC_SUBSCRIPTION_IN_USE.

Chcete-li tuto volbu přidat k existujícímu odběru, musí volání MQSUB (pomocí funkce MQSO_ALTER) pocházet ze stejného ID uživatele jako původní odběr samotný.

Pokud volání MQSUB odkazuje na existující odběr se sadou MQSO_ANY_USERID a ID uživatele se liší od původního odběru, volání se zdaří pouze v případě, že má nové ID uživatele oprávnění k odběru daného tématu. Při úspěšném dokončení jsou budoucí publikace k tomuto odběrateli vloženy do fronty odběratelů s použitím nového ID uživatele nastaveného ve zprávě publikování.

Nezadávejte parametry MQSO_ANY_USERID a MQSO_FIXED_USERID. Není-li zadán ani jeden z těchto parametrů, bude použita výchozí hodnota MQSO_FIXED_USERID.

ID UŽIVATELE MQSO_FIXED_USERID

Je-li zadáno MQSO_FIXED_USERID, může být odběr změněn nebo obnoven pouze posledním ID uživatele, aby mohl být změněn odběr. Pokud odběr nebyl změněn, jedná se o ID uživatele, který vytvořil daný odběr.

Pokud příkaz MQSUB odkazuje na existující odběr s nastaveným parametrem MQSO_ANY_USERID a pozmění odběr pomocí funkce MQSO_ALTER pro použití volby MQSO_FIXED_USERID, bude ID uživatele odběru nyní opraveno v tomto novém ID uživatele. Volání se zdaří pouze tehdy, má-li nové ID uživatele oprávnění přihlásit se k odběru tématu.

Pokud se ID uživatele, které není zaznamenáno jako vlastnící odběr, pokusí obnovit nebo změnit odběr MQSO_FIXED_USERID, volání selže s chybou MQRC_IDENTITY_MISMATCH. Vlastnící ID uživatele odběru lze zobrazit pomocí příkazu DISPLAY SBSTATUS.

Nezadávejte parametry MQSO_ANY_USERID a MQSO_FIXED_USERID. Není-li zadán ani jeden z těchto parametrů, bude použita výchozí hodnota MQSO_FIXED_USERID.

Volby publikování: Následující volby řídí způsob, jakým jsou publikacemi odeslány tomuto odběrateli. Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze tyto volby publikování změnit.

MQSO_NOT_OWN_PUBS
Sděluje zprostředkovateli, že aplikace nechce vidět žádná ze svých vlastních publikací. Publikace se považují za produkty pocházející ze stejné aplikace, jsou-li úchyty připojení stejné. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena, je-li to vhodné.

POUZE NOVÉ_VEŘEJNÉ_VEŘEJNÉ_PUBLIKOVÁNÍ
Při vytváření tohoto odběru se neuchovávají žádné aktuálně zachované publikace, pouze nové publikace. Tato volba se používá pouze v případě, že je zadán parametr MQSO_CREATE. Veškeré

následné změny odběru neovlivňují tok publikování, a proto budou všechny publikace, které byly uchovány v rámci tématu, odeslány odběrateli jako nové publikace.

Je-li tato volba zadána bez volání MQSO_CREATE, volání selže s chybou MQRC_OPTIONS_ERROR. Při návratu z volání MQSUB pomocí MQSO_RESUME není tato volba nastavena, i když byl odběr vytvořen pomocí této volby.

Není-li tato volba použita, budou dříve zachované zprávy odeslány do zadané cílové fronty. Pokud tato akce selže kvůli chybě, buď MQRC_RETAINED_MSG_Q_ERROR nebo MQRC_RETAINED_NOT_DELIVERED, dojde k selhání vytvoření odběru.

POŽADAVEK MQSO_PUBLICATIONS_ON_REQUEST
Nastavení této volby označuje, že odběratel bude požadovat informace konkrétně, když je to požadováno. Správce front neodesílá nevyžádané zprávy do odběratele. Zachované publikování (nebo možná více publikování v případě, že je v tématu uveden zástupný znak) se odešle odběrateli pokaždé, když je volání MQSUBRQ provedeno pomocí obslužné rutiny Hsub z předchozího volání MQSUB. Při volání MQSUB s použitím této volby nejsou odesílána žádná publikování. Při návratu z volání MQSUB pomocí MQSO_RESUME je tato volba nastavena, je-li to vhodné.

Tato volba není platná v kombinaci s úrovní SubLevel větší než 1.

Volby dopředného čtení: Následující volby řídí, zda jsou netrvalé zprávy odesílány aplikaci před tím, než je aplikace požaduje.

FUNKCE MQSO_READ_AHEAD_AS_Q_DEF
Pokud volání MQSUB používá spravovaný popisovač, použije se výchozí atribut dopředného čtení fronty modelu přidružené k tématu přihlášenému k určení, zda jsou zprávy odeslány aplikaci před tím, než je aplikace požaduje.

Toto je výchozí hodnota.

MQSO_NO_READ_AHEAD
Pokud volání MQSUB používá spravovaný popisovač, nebudou zprávy odeslány do aplikace dříve, než je aplikace požaduje.

MQSO_READ_AHEAD
Pokud volání MQSUB používá spravovanou obslužnou rutinu, mohou být aplikace odeslány do aplikace dříve, než je aplikace požaduje.

Poznámka:

Pro volby čtení napřed se vztahují následující poznámky:

1. Může být uvedena pouze jedna z těchto voleb. Jsou-li zadány funkce MQOO_READ_AHEAD a MQOO_NO_READ_AHEAD, vrátí se kód příčiny MQRC_OPTIONS_ERROR. Tyto volby jsou použitelné pouze v případě, že je zadán parametr MQSO_MANAGED.
2. Nejsou použitelné pro MQSUB, když je předána fronta, která již byla otevřena dříve. Čtení napřed nemusí být povoleno, je-li to požadováno. Volby MQGET použité při prvním volání MQGET mohou zabránit, aby bylo povoleno čtení napřed. Funkce dopředného čtení je také zablokována, když se klient připojuje ke správci front, kde není podporováno čtení napřed. Pokud aplikace není spuštěna jako klient produktu IBM MQ , jsou tyto volby ignorovány.

Volby zástupných znaků: Následující volby řídí, jak jsou zástupné znaky interpretovány v řetězci poskytnutém v poli ObjectString MQSD. Můžete uvést pouze jednu z těchto voleb. Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze tyto volby zástupného znaku změnit. Při návratu z volání MQSUB pomocí funkce MQSO_RESUME je nastavena příslušná volba zástupného znaku.

MQSO_WILDCARD_CHAR
Zástupné znaky fungují pouze na znacích v řetězci tématu.

Chování definované příkazem MQSO_WILDCARD_CHAR je zobrazeno v následující tabulce.

Tabulka 529. Jak jsou interpretovány zástupné znaky	
Speciální znak	**Chování**
Lomítko (/)	Žádný význam, jen další postava
Hvězdička (*)	Zástupný znak, nula nebo více znaků
Otazník (?)	Zástupný znak, 1 znak
Procento (%)	Únikový znak, který umožňuje použití znaků (*), (?) nebo (%) v řetězci a nebude interpretován jako speciální znak, například (% *), (%?) nebo (%%).

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

Vyhovuje odběrateli pomocí následujících témat:

```
*
/*
/ level0/level1/level2/level3/*
/ level0/level1/*/level3/level4
/ level0/level1/le?el2/level3/level4
```

Poznámka: Toto použití zástupných znaků poskytuje přesně význam poskytnutý v IBM MQ V6 a WebSphere MB V6 při použití formátovaných zpráv MQRFH1 pro publikování/odběr. Doporučuje se, aby toto nebylo použito pro nově vytvořené aplikace a používá se pouze pro aplikace, které byly dříve spuštěny proti této verzi a nebyly změněny tak, aby používaly výchozí chování zástupného znaku, jak je popsáno v MQSO_WILDCARD_TOPIC.

TÉMA MQSO_WILDCARD_TOPIC

Zástupné znaky fungují pouze na prvcích témat v řetězci tématu. Jedná se o výchozí chování, pokud není žádné zvoleno.

Chování požadované operací MQSO_WILDCARD_TOPIC je zobrazeno v následující tabulce:

Tabulka 530. Jak jsou interpretovány zástupné znaky	
Speciální znak	**Chování**
(/)	Oddělovač úrovně tématu
Znaménko čísla (#)	Zástupný znak: více úrovní tématu
Znaménko plus (+)	Zástupný znak: jedna úroveň tématu

Notes:

Znaky (+) a (#) nejsou považovány za zástupné znaky, jsou-li smíšeny s ostatními znaky (včetně samotných) v rámci úrovně tématu. V následujícím řetězci jsou znaky (#) a (+) považovány za běžné znaky.

```
level0/level1/#+/level3/level#
```

Například publikování na následující téma:

```
/level0/level1/level2/level3/level4
```

Vyhovuje odběrateli pomocí následujících témat:

```
#
/#
/ level0/level1/level2/level3/#
/ level0/level1/+/level3/level4
```

Další volby: Následující volby řídí způsob, jakým je volání rozhraní API vydáno spíše než odběr. Při návratu z volání MQSUB pomocí funkce MQSO_RESUME se tyto volby nezměnily. Další informace viz část "ID AlternateUserID (MQCHAR12)" na stránce 575.

OPRÁVNĚNÍ UŽIVATELE MQSO_ALTERNATE_USER_AUTHORITY

Pole ID AlternateUserobsahuje identifikátor uživatele, který se má použít k ověření tohoto volání MQSUB. Volání může být úspěšné pouze v případě, že je tento identifikátor AlternateUserautorizován k otevření objektu s uvedenými volbami přístupu bez ohledu na to, zda je identifikátor uživatele, pod kterým je aplikace spuštěna, oprávněn tak učinit.

ID_SADY_MQSO_SET_CORRELACE_

Předplatné má použít identifikátor korelace zadaný v poli *SubCorrelId* . Není-li tato volba zadána, bude identifikátor korelace automaticky vytvořen správcem front v době odběru a je vrácen aplikaci v poli *SubCorrelId* . Další informace viz "ID SubCorrel(MQBYTE24)" na stránce 577 .

Tuto volbu nelze kombinovat s funkcí MQSO_MANAGED.

KONTEXT MQSO_SET_IDENTITY_CONTEXT

Předplatné má použít účtovací token a data identity aplikace zadané v polích *PubAccountingToken* a *PubApplIdentityData* .

Je-li tato volba zadána, provede se stejná kontrola autorizace jako v případě, že k cílové frontě bylo přistupováno pomocí volání MQOPEN s MQOO_SET_IDENTITY_CONTEXT, s výjimkou případu, kdy je použita volba MQSO_MANAGED také v tom případě, že v cílové frontě není žádná kontrola autorizace.

Není-li tato volba zadána, budou k publikacím odeslaným pro tohoto odběratele přidruženy výchozí informace o kontextu:

Tabulka 531. Výchozí kontextové informace pro publikace odeslané k tomuto odběrateli

Pole v MQMD	**Použitá hodnota**
UserIdentifier	ID uživatele přidružené k odběru v době, kdy byl proveden odběr.
AccountingToken	Určeno z prostředí, je-li to možné; nastavte hodnotu MQACT_NONE, pokud není.
ApplIdentityData	Nastavit na prázdné znaky

Tato volba je platná pouze s MQSO_CREATE a MQSO_ALTER. Pokud se používá s MQSO_RESUME, pole *PubAccountingToken* a *PubApplIdentityData* se ignorují, takže tato volba nemá žádný efekt.

Je-li odběr změněn bez použití této volby, pokud dříve předplatné informace o kontextu identity, výchozí informace o kontextu se vygenerují pro změněný odběr.

Je-li odběr povolující použití jiných ID uživatelů s volbou MQSO_ANY_USERID obnoven jiným ID uživatele, bude vygenerován výchozí kontext identity pro nové ID uživatele, které nyní vlastní odběr, a budou doručena všechna následující publikování obsahující nový kontext identity.

MQSO_FAIL_IF_QUIESCING

Volání MQSUB selže, pokud se správce front nachází ve stavu uvedení do klidového stavu.
U z/OSu aplikací CICS nebo IMS tato volba také vynutí selhání volání MQSUB, pokud je připojení ve stavu uvedení do klidového stavu.

ObjectName (MQCHAR48)

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.

ID AlternateUserID (MQCHAR12)

Pokud uvedete MQSO_ALTERNATE_USER_AUTHORITY, toto pole obsahuje alternativní identifikátor uživatele, který se používá ke kontrole autorizace pro odběr a pro výstup do cílové fronty (zadané v parametru **Hobj** volání MQSUB), místo identifikátoru uživatele, pod kterým momentálně běží aplikace.

Je-li úspěšný, identifikátor uživatele uvedený v tomto poli se zaznamená jako identifikátor uživatele, který je vlastníkem, místo identifikátoru uživatele, pod kterým momentálně běží aplikace.

Je-li zadán parametr MQSO_ALTERNATE_USER_AUTHORITY a toto pole je zcela prázdné až na první znak null nebo na konci pole, může být odběr úspěšný pouze v případě, že není k odběru tohoto tématu s použitím zadaných voleb nebo cílové fronty pro výstup vyžadována žádná autorizace uživatele.

Není-li parametr MQSO_ALTERNATE_USER_AUTHORITY zadán, bude toto pole ignorováno.

V označeném prostředí existují následující rozdíly:

- V systému z/OSse ke kontrole autorizace pro odběr použije pouze prvních 8 znaků identifikátoru AlternateUser. Avšak, aktuální identifikátor uživatele musí být autorizován k uvedení tohoto konkrétního alternativního identifikátoru uživatele; pro tuto kontrolu se použijí všech 12 znaků alternativního

identifikátoru uživatele. Identifikátor uživatele musí obsahovat pouze znaky povolené externím správcem zabezpečení.

Při návratu z volání MQSUB pomocí funkce MQSO_RESUME se toto pole nezmění.

Toto je vstupní pole. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 12 prázdných znaků v jiných programovacích jazycích.

AlternateSecurityId (MQBYTE40)

Jedná se o identifikátor zabezpečení předávaný spolu s ID AlternateUserk autorizační službě, aby bylo možné provádět odpovídající kontroly autorizace.

ID AlternateSecurityID se používá pouze v případě, že je zadán parametr MQSO_ALTERNATE_USER_AUTHORITY a pole ID AlternateUsernení zcela prázdné do prvního znaku null nebo do konce pole.

Při návratu z volání MQSUB pomocí funkce MQSO_RESUME se toto pole nezmění.

Další informace naleznete v popisu “AlternateSecurityId (MQBYTE40)” na stránce 488 v datovém typu MQOD.

SubExpiry (MQLONG)

Jedná se o čas vyjádřený v desetinách sekundy, po jehož uplynutí vyprší platnost odběru. Po uplynutí tohoto intervalu nebudou k tomuto odběru odpovídat žádné další publikace. Jakmile dojde k vypršení platnosti odběru, publikování se již nebude odesílat do fronty. Avšak publikace, které již existují, nejsou žádným způsobem ovlivněny. *SubExpiry* nemá žádný vliv na vypršení platnosti publikace.

Je rozpoznána následující speciální hodnota:

MQEI_UNLIMITED
Odběr má neomezenou dobu platnosti.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, může dojít ke změně vypršení odběru.

Při návratu z volání MQSUB s použitím volby MQSO_RESUME je toto pole nastaveno na původní vypršení platnosti odběru a nikoli na zbývající dobu platnosti.

ObjectString (MQCHARV)

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)

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.

Data SubUserData (MQCHARV)

Určuje data uživatele odběru. Data poskytnutá na odběru v tomto poli budou zahrnuta jako vlastnost datové zprávy MQSubUserpro každou publikaci odeslanou do tohoto odběru.

Maximální délka *SubUserData* je 10240.

Pokud je parametr *SubUserData* zadán nesprávně, v souladu s popisem způsobu použití struktury MQCHARV , nebo pokud překročí 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 počáteční hodnoty v rámci struktury MQCHARV.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze změnit uživatelská data odběru.

Tato proměnná délka proměnné je vrácena ve výstupu z volání MQSUB s použitím volby MQSO_RESUME, je-li vyrovnávací paměť k dispozici a v produktu *VSBufLen*je k dispozici kladná délka vyrovnávací paměti. Není-li v rámci volání k dispozici žádná vyrovnávací paměť, bude v poli *VSLength* MQCHARV vrácena pouze délka data uživatele odběru. Je-li poskytnutá vyrovnávací paměť menší než prostor potřebný k vrácení pole, vrátí se ve vyrovnávací paměti pouze *VSBufLen* bajtů.

ID SubCorrel(MQBYTE24)

Toto pole obsahuje identifikátor korelace společný pro všechny publikace odpovídající tomuto odběru.

Upozornění: Identifikátor korelace může být předáván pouze mezi správci front v klastru publikování/odběru, ne v hierarchii.

Všechny publikace odeslané tak, aby odpovídaly tomuto odběru, obsahují tento korelační identifikátor v deskriptoru zpráv. Pokud více odběrů získává své publikace ze stejné fronty použitím identifikátoru MQGET podle identifikátoru korelace, lze získat pouze publikování pro specifický odběr, který má být získán. Tento korelační identifikátor může vygenerovat buď správce front, nebo uživatel.

Není-li určena 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. Vygenerovaný korelační identifikátor se skládá z 4bajtového identifikátoru produktu

(AMQX nebo CSQM buď v kódu ASCII, nebo EBCDIC), za nímž následuje implementace jedinečného řetězce specifický pro produkt.

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 každé publikaci pro tento odběr. V tomto případě, pokud pole obsahuje MQCI_NONE, je korelační identifikátor, který je nastaven v každé zprávě publikované pro tento odběr, korelační identifikátor vytvořený původním vložením zprávy.

Je-li zadána volba MQSO_GROUP_SUB a zadaný identifikátor korelace je shodný s existujícím seskupeným odběrem s použitím stejné fronty a překrývajícím se řetězcem tématu, je k dispozici pouze nejvýznamnější odběr ve skupině s kopií této publikace.

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, pak lze identifikátor korelace odběru změnit, pokud odběr není seskupeným odběrem, tj. byl vytvořen pomocí volby MQSO_GROUP_SUB, v takovém případě nelze změnit identifikátor korelace odběru.

Při návratu z volání MQSUB pomocí příkazu MQSO_RESUME je toto pole nastaveno na aktuální identifikátor korelace pro daný odběr.

PubPriority (MQLONG)

Jedná se o hodnotu, která bude v poli *Priority* deskriptoru zpráv (MQMD) všech publikovaných zpráv, odpovídajících tomuto odběru. Další informace o poli *Priority* v deskriptoru MQMD najdete v tématu “Priorita (MQLONG)” na stránce 448.

Hodnota musí být větší než nula nebo rovna nule; nula je nejnižší priorita. Mohou být použity také následující speciální hodnoty:

MQPRI_PRIORITY_AS_Q_DEF

Je-li fronta odběru uvedena v poli *Hobj* ve volání MQSUB a nejedná se o spravovaný popisovač, bude priorita zprávy převzata z atributu **DefPriority** této fronty. Je-li fronta fronta klastru nebo existuje více než jedna definice v cestě rozpoznání názvu fronty, pak se priorita určuje, když je zpráva publikování vložena do fronty, jak je popsáno pro “Priorita (MQLONG)” na stránce 448.

Pokud volání MQSUB používá spravovanou obslužnou rutinu, bude priorita zprávy převzata z atributu **DefPriority** ve frontě modelu přidružené k odběru tématu přihlášenému k odběru.

MQPRI_PRIORITY_AS_PUBLISHED

Priorita pro zprávu je priorita původní publikace. Toto je počáteční hodnota pole.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze změnit *Priority* ze všech budoucích zpráv publikování.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální prioritu používanou pro odběr.

Token PubAccounting(MQBYTE32)

Jedná se o hodnotu, která bude v poli *AccountingToken* deskriptoru zpráv (MQMD) všech publikovaných zpráv, odpovídajících tomuto odběru. *AccountingToken* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz téma Kontext zprávy. Další informace o poli *AccountingToken* v deskriptoru MQMD najdete v tématu “AccountingToken (MQBYTE32)” na stránce 455 .

Pro pole *PubAccountingToken* můžete použít následující speciální hodnotu:

MQACT_NONE

Není zadán žádný token účtování.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQACT_NONE_ARRAY; hodnota má stejnou hodnotu jako MQACT_NONE, ale je to pole znaků namísto řetězce.

Není-li volba MQSO_SET_IDENTITY_CONTEXT určena, vygeneruje správce front jako výchozí informace o kontextu správce front a toto pole je výstupní pole obsahující *AccountingToken* , které bude nastaveno v každé zprávě publikované pro tento odběr.

Je-li zadána volba MQSO_SET_IDENTITY_CONTEXT, generuje se token evidence uživatelem a toto pole je vstupním polem, které obsahuje sadu *AccountingToken* , jež má být nastavena v každé publikaci pro tento odběr.

Délka tohoto pole je dána hodnotou MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota tohoto pole je MQACT_NONE.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze změnit hodnotu parametru *AccountingToken* ve všech budoucích zprávách publikování.

Při návratu z volání MQSUB pomocí funkce MQSO_RESUME je toto pole nastaveno na aktuální *AccountingToken* , který se používá pro odběr.

PubApplIdentityData (MQCHAR32)

Jedná se o hodnotu, která je v poli *ApplIdentityData* deskriptoru zpráv (MQMD) všech publikovaných zpráv, odpovídajících tomuto odběru. *ApplIdentityData* je součástí kontextu identity zprávy. Další informace o kontextu zprávy viz téma Kontext zprávy. Další informace o poli *ApplIdentityData* v deskriptoru MQMD najdete v tématu “Data ApplIdentity(MQCHAR32)” na stránce 457 .

Není-li volba MQSO_SET_IDENTITY_CONTEXT určena, je hodnota *ApplIdentityData* , která je nastavena v každé zprávě publikované pro tento odběr, prázdná, jako výchozí kontextové informace.

Je-li zadána volba MQSO_SET_IDENTITY_CONTEXT, generuje se *PubApplIdentityData* uživatelem a toto pole je vstupní pole, které obsahuje *ApplIdentityData* , které má být nastaveno v každé publikaci pro tento odběr.

Délka tohoto pole je dána hodnotou MQ_APPL_IDENTITY_DATA_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 32 prázdných znaků v jiných programovacích jazycích.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, lze změnit *ApplIdentityData* ze všech budoucích zpráv publikování.

Při návratu z volání MQSUB pomocí funkce MQSO_RESUME je toto pole nastaveno na aktuální *ApplIdentityData* , který se používá pro odběr.

SelectionString (MQCHARV)

Jedná se o řetězec používaný k poskytnutí kritérií výběru používaných při odběru zpráv z tématu.

Tato proměnná délka proměnné bude vrácena ve výstupu z volání MQSUB s použitím volby MQSO_RESUME, je-li zadána vyrovnávací paměť, a také v parametru VSBufSizeje kladná délka vyrovnávací paměti. Není-li na volání k dispozici žádná vyrovnávací paměť, bude v poli VSLlength pole MQCHARV vrácena pouze délka řetězce výběru. Je-li poskytnutá vyrovnávací paměť menší než prostor potřebný k navrácení pole, vrátí se ve vyrovnávací paměti pouze bajty VSBufSize .

Pokud je parametr *SelectionString* zadán nesprávně, v souladu s popisem způsobu použití struktury “MQCHARV-Řetězec délky proměnné” na stránce 294 , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SELECTION_STRING_ERROR.

Použití SelectionString je popsáno v Selektory.

SubLevel (MQLONG)

Toto je úroveň přidružená k odběru. Publikace jsou k tomuto odběru doručeny pouze v případě, že jsou v sadě odběrů s nejvyšší hodnotou SubLevel menší nebo rovny hodnotě PubLevel použité v době

publikování. Pokud však byla publikace zachována, není již dostupná odběratelům na vyšší úrovni, protože je znovu publikována na úrovni PubLevel 1.

Hodnota musí být v rozsahu nula až 9. Nula je nejnižší úroveň.

Počáteční hodnota tohoto pole je 1.

Další informace viz Zachytávání publikací.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze změnit SubLevel .

Sloučení SubLevel s hodnotou větší než 1 s volbou MQSO_PUBLICATIONS_ON_REQUEST není povoleno.

Při návratu z volání MQSUB pomocí MQSO_RESUME je toto pole nastaveno na aktuální úroveň použitou pro odběr.

Řetězec ResObject(MQCHARV)

Jedná se o dlouhý název objektu poté, co správce front interpretuje název poskytnutý v produktu *ObjectName*.

Pokud je v produktu *ObjectString* zadán dlouhý název objektu a v produktu *ObjectName*není k dispozici nic, vrátí se hodnota vrácená v tomto poli stejná jako hodnota uvedená v části *ObjectString*.

Je-li toto pole vynecháno (toto pole je ResObjectString.VSBufSize je nula), pak se *ResObjectString* nevrátí, ale délka je vrácena jako ResObjectString.VSLength. Je-li délka kratší než úplný řetězec ResObject, bude oříznut a vrátí se jako počet znaků nejvíce vpravo, které se mohou vejít do zadané délky.

Pokud je parametr *ResObjectString* zadán nesprávně, v souladu s popisem 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 532. 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

Tabulka 532. Pole v objektu MQSMPO (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.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)

Jedná se o identifikátor struktury; hodnota musí být:

MQSMPO_STRUCTION_ID

Identifikátor pro nastavení struktury voleb vlastností zprávy.

Pro programovací jazyk C je také definována konstanta **MQSMPO_STRUC_ID_ARRAY** ; tato hodnota má stejnou hodnotu jako **MQSMPO_STRUC_ID**, ale je pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQSMPO_STRUC_ID**.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQSMPO_VERSION_1
Version-1 -nastavení struktury voleb vlastností zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSMPO_CURRENT_VERSION
Aktuální verze struktury voleb vlastností sady zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQSMPO_VERSION_1**.

Volby (MQLONG)

Volby umístění

Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti:

NEJPRVE LQSMPO_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 vlastností. Vlastnost, na kterou odkazuje kurzor vlastnosti, je ta, která byla naposledy dotazovaná pomocí volby MQIMPO_INQ_FIRST nebo volby MQIMPO_INQ_NEXT.

Kurzor vlastností se resetuje, když je popisovač zprávy znovu použit na volání MQGET nebo pokud je popisovač zprávy zadán v poli *MsgHandle* struktury MQGMO nebo MQPMO na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud vytvořen, nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti vymazán, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

FUNKCE MQSMPO_SET_PROP_BEFORE_CURSOR
Nastaví novou vlastnost před vlastností, na kterou ukazuje kurzor, který je uveden ve vlastnosti. Vlastnost, na kterou odkazuje kurzor vlastnosti, je ta, která byla naposledy dotazovaná pomocí volby MQIMPO_INQ_FIRST nebo volby MQIMPO_INQ_NEXT.

Kurzor vlastností se resetuje, když je popisovač zprávy znovu použit na volání MQGET nebo pokud je popisovač zprávy zadán v poli *MsgHandle* struktury MQGMO nebo MQPMO na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud vytvořen, nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti vymazán, 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 vlastnost kurzor vlastností. Vlastnost, na kterou odkazuje kurzor vlastnosti, je ta, která byla naposledy dotazovaná pomocí volby MQIMPO_INQ_FIRST nebo volby MQIMPO_INQ_NEXT.

Kurzor vlastností se resetuje, když je popisovač zprávy znovu použit na volání MQGET nebo pokud je popisovač zprávy zadán v poli *MsgHandle* struktury MQGMO nebo MQPMO na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud vytvořen, nebo pokud byl ukazatel vlastnosti na kurzor vlastnosti vymazán, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

VLASTNOST MQSMPO_APPEND_PROPERTY
Způsobí přidání nové vlastnosti po všech ostatních vlastnostech s odpovídající hierarchií. Pokud existuje alespoň jedna vlastnost, která odpovídá zadanému názvu, bude nová vlastnost přidána na konec po konci tohoto seznamu vlastností.

Tato volba umožňuje vytvoření seznamu vlastností se stejným názvem.

Pokud nepotřebujete žádné 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)

Kódování hodnoty vlastnosti, která má být nastavena, je-li hodnota číselná.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQENC_NATIVE**.

ValueCCSID (MQLONG)

Znaková sada hodnoty vlastnosti, která má být nastavena, je-li 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

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)

Jedná se o identifikátor struktury; hodnota musí být:

ID_STRUKTURY MQSRO_STRUCTION_ID
Identifikátor struktury Volby požadavku na odběr.

Pro programovací jazyk C je také definován konstantní MQSRO_STRUC_ID_ARRAY; má stejnou hodnotu jako MQSRO_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSRO_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury; hodnota musí být:

MQSRO_VERSION_1
Version-1 Struktura voleb požadavku na odběr.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSRO_CURRENT_VERSION
Aktuální verze struktury Volby požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSRO_VERSION_1.

Volby (MQLONG)

Musí být uvedena jedna z následujících voleb. Může být uvedena pouze jedna volba.

MQSRO_FAIL_IF_QUIESCING

Volání MQSUBRQ se nezdaří, je-li správce front ve stavu uvedení do klidového stavu. V produktu z/OSv případě aplikace CICS nebo IMS tato volba také vynutí selhání volání MQSUBRQ v případě, že se připojení nachází ve stavu uvedení do klidového stavu.

Výchozí volba: Pokud dříve popsaná volba není povinná, je třeba použít následující volbu:

MQSRO_NONE

Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

Funkce MQSRO_NONE pomáhá programovým dokumentaci. Ačkoli se nejedná o zamýšlené použití této volby, protože její hodnota je nulová, nelze toto použití detekovat.

NumPubs (MQLONG)

Jedná se o výstupní pole, které se vrátí do aplikace a označuje počet publikování odeslaných do fronty odběru jako výsledek tohoto volání. Přestože byl tento počet publikací odeslán jako výsledek tohoto volání, není zaručeno, že bude pro aplikaci k dispozici mnoho zpráv, zvláště pokud jde o netrvalé zprávy.

Pokud téma přihlášené k odběru zástupného znaku obsahovalo zástupný znak, může existovat více než jedna publikace. Pokud nebyly nalezeny žádné zástupné znaky v řetězci tématu, když byl vytvořen odběr představovaný *Hsub* , pak se v důsledku tohoto volání odešle nanejvýš jedna publikace.

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 533. 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.		

Tabulka 533. Pole v MQSTS (pokračování)

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
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-Načíst informace o stavu” na stránce 777
Použijte volání MQSTAT k získání informací o stavu. Typ vrácených informací o stavu je určen hodnotou typu uvedenou ve volání.

StrucId (MQCHAR4)

Identifikátor pro strukturu vykazování stavu, MQSTS.

StrucId je identifikátor struktury. Hodnota musí být:

ID_STRUKTURY MQSTS_
Identifikátor struktury vykazování stavu.

Pro programovací jazyk C je také definována konstanta MQSTS_STRUC_ID_ARRAY ; tato hodnota má stejnou hodnotu jako MQSTS_STRUC_ID, ale je pole znaků místo řetězce.

StrucId je vždy vstupní pole. Jeho počáteční hodnota je MQSTS_STRUC_ID.

Verze (MQLONG)

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 uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSTS_CURRENT_VERSION
Aktuální verze struktury vykazování stavu. Aktuální verze je MQSTS_VERSION_2.

Version je vždy vstupní pole. Jeho počáteční hodnota je MQSTS_VERSION_1.

CompCode (MQLONG)
Kód dokončení operace, která se vykazuje.

Interpretace parametru CompCode závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Jedná se o kód dokončení, který je výsledkem předchozí asynchronní operace put pro objekt uvedený v souboru ObjectName.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Pokud se připojení znovu připojuje nebo selhalo opětovné připojení, je to kód dokončení, který způsobil, že připojení začalo znovu navázat spojení.

Je-li připojení momentálně připojeno, hodnota je MQCC_OK.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nezdařilo znovu navázat spojení, je to kód dokončení, který způsobil selhání opětovného připojení.

Je-li připojení momentálně připojeno, nebo se znovu připojuje, hodnota je MQCC_OK.

CompCode je vždy výstupní pole. Jeho počáteční hodnota je MQCC_OK.

Příčina (MQLONG)
Kód příčiny operace, na kterou se hlásí operace.

Interpretace parametru Reason závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Jedná se o kód příčiny, který je výsledkem předchozí operace asynchronního vložení na objektu uvedeném v souboru ObjectName.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Pokud se připojení znovu připojuje nebo selhalo opětovné připojení, je to kód příčiny, který způsobil opětovné připojení k opětovnému připojení.

Je-li připojení momentálně připojeno, hodnota je MQRC_NONE.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nezdařilo znovu připojit, je to kód příčiny, který způsobil selhání opakovaného připojení.

Je-li připojení momentálně připojeno, nebo se znovu připojuje, hodnota je MQRC_NONE.

Reason je výstupní pole. Jeho počáteční hodnota je MQRC_NONE.

Počet PutSuccessCount (MQLONG)
Počet asynchronních operací vložení, které byly úspěšné.

Hodnota parametru PutSuccessCount závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení na objekt pojmenovaný ve struktuře MQSTS , která byla dokončena s MQCC_OK.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nula.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutSuccessCount je výstupní pole. Jeho počáteční hodnota je nula.

Počet operací PutWarning(MQLONG)

Počet asynchronních operací vložení, které skončily s varováním.

Hodnota parametru PutWarningCount závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení na objekt pojmenovaný ve struktuře MQSTS , která byla dokončena s MQCC_WARNING.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nula.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutWarningCount je výstupní pole. Jeho počáteční hodnota je nula.

Počet operací PutFailure(MQLONG)

Počet asynchronních operací vložení, které selhaly.

Hodnota parametru PutFailureCount závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Počet asynchronních operací vložení na objekt pojmenovaný ve struktuře MQSTS , která byla dokončena s MQCC_FAILED.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nula.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Nula.

PutFailureCount je výstupní pole. Jeho počáteční hodnota je nula.

ObjectType (MQLONG)

Typ objektu jmenovaného v *ObjectName* je ohlášen v.

Možné hodnoty parametru ObjectType jsou uvedeny v seznamu “MQOT_ * (typy objektů a rozšířené typy objektů)” na stránce 163.

ObjectType je výstupní pole. Jeho počáteční hodnota je MQOT_Q.

ObjectName (MQCHAR48)

Název objektu, u kterého se vykazuje zpráva.

Interpretace parametru ObjectName závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Jedná se o název fronty nebo tématu použitého v operaci put, jejíž selhání se vykazuje v polích *CompCode* a *Reason* ve struktuře MQSTS .

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Je-li připojení znovu připojováno, jedná se o název správce front přidruženého k připojení.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nezdařilo znovu připojit, jedná se o název objektu, který způsobil selhání opakovaného připojení. Příčina selhání se vykazuje v polích *CompCode* a *Reason* ve struktuře MQSTS .

ObjectName je výstupní pole. Jeho počáteční hodnota je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název ObjectQMgr(MQCHAR48)

Název vykazovaného správce front.

Interpretace parametru ObjectQMgrName závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Jedná se o název správce front, ve kterém je definován objekt *ObjectName* . Název, který je zcela prázdný až k prvnímu znaku null nebo konec pole označuje správce front, ke kterému je aplikace připojena (lokální správce front).

V 9.2.0 **PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION**

Multi

Pole **ObjectQMgrName** obsahuje název správce front, pro který je požadováno opětovné připojení, nebo prázdné pole, pokud není zadán správce front. Je-li to možné, klient se pokusí znovu připojit ke správci front s tímto názvem.

z/OS

Prázdné.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Pokud se připojení nezdařilo znovu připojit, jedná se o název objektu, který způsobil selhání opakovaného připojení. Příčina selhání se vykazuje v polích *CompCode* a *Reason* ve struktuře MQSTS .

ObjectQMgrName je výstupní pole. Jeho hodnota je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název ResolvedObject(MQCHAR48)

Název objektu uvedeného v souboru *ObjectName* poté, co název lokálního správce front vyřeší název.

Interpretace parametru ResolvedObjectName závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

ResolvedObjectName je název objektu uvedeného v souboru *ObjectName* poté, co lokální správce front vyřeší daný název. Vrácený název je název objektu, který existuje ve správci front identifikovaném příkazem *ResolvedQMgrName*.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Prázdné.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Prázdné.

ResolvedObjectName je výstupní pole. Jeho počáteční hodnota je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název ResolvedQMgr(MQCHAR48)

Název cílového správce front poté, co název lokálního správce front vyřeší název.

Interpretace parametru ResolvedQMgrName závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

ResolvedQMgrName je název cílového správce front poté, co lokální správce front vyřeší daný název. Vrácený název je název správce front, který vlastní objekt identifikovaný produktem *ResolvedObjectName*. *ResolvedQMgrName* může být název lokálního správce front.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Prázdné.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Prázdné.

ResolvedQMgrName je vždy výstupní pole. Jeho počáteční hodnota je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

ObjectString (MQCHARV)

Dlouhý název objektu, u kterého se vykazuje selhávající objekt. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru ObjectString závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Jedná se o dlouhý název objektu fronty nebo tématu použitého v operaci MQPUT , která se nezdařila.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Řetězec s nulovou délkou

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Jedná se o dlouhý název objektu objektu, který způsobil selhání opětovného připojení.

ObjectString je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

SubName (MQCHARV)

Název selhávajícího odběru. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru SubName závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Nulová délka řetězce.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nulová délka řetězce.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Název odběru, který způsobil selhání opětovného připojení. Není-li k dispozici žádný název odběru nebo selhání nesouvisí s odběrem, je to řetězec s nulovou délkou.

SubName je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

OpenOptions (MQLONG)

Objekt OpenOptions se používá k otevření objektu, který je hlášen. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Hodnota parametru OpenOptions závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Nula.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nula.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

OpenOptions použitý, když došlo k selhání. Příčina selhání se vykazuje v polích *CompCode* a *Reason* ve struktuře MQSTS .

OpenOptions je výstupní pole. Jeho počáteční hodnota je nula.

SubOptions (MQLONG)

SubOptions použil k otevření selhávajícího odběru. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru SubOptions závisí na hodnotě parametru MQSTAT **Type** .

CHYBA MQSTAT_TYPE_ASYNC_ERROR

Nula.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Nula.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

SubOptions použitý, když došlo k selhání. Pokud se 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).

Pouľití

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 600.

- 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 534. 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
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 535. 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
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)

Jedná se o identifikátor struktury. Hodnota musí být:

ID_STRUKTURY MQTM_STRUCT
Identifikátor pro strukturu zprávy spouštěče.

Pro programovací jazyk C je také definována konstanta MQTM_STRUC_ID_ARRAY; má stejnou hodnotu jako MQTM_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQTM_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQTM_VERSION_1
Číslo verze pro strukturu zprávy spouštěče.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQTM_AKTUÁLNÍ_VERZE
Aktuální verze struktury zprávy spouštěče.

Počáteční hodnota tohoto pole je MQTM_VERSION_1.

QName (MQCHAR48)

Jedná se o název fronty, pro kterou došlo k události spouštěče, a je použita aplikací spuštěnou aplikací pro monitor spouštěčů. Správce front inicializuje toto pole hodnotou atributu **QName** spuštěné fronty; podrobnosti o tomto atributu naleznete v příručce“Atributy pro fronty” na stránce 827.

Názvy, které jsou kratší než definovaná délka pole, jsou směrem doprava vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

ProcessName (MQCHAR48)

Jedná se o název objektu procesu správce front zadaný pro spuštěnou frontu a může být použit aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ProcessName** fronty identifikované polem *QName* ; podrobnosti o tomto atributu viz“Atributy pro fronty” na stránce 827.

Názvy, které jsou kratší než definovaná délka pole, jsou vždy doplněny vpravo s mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána hodnotou MQ_PROCESS_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

TriggerData (MQCHAR64)

Jedná se o volný formát dat pro použití aplikací monitoru spouštěčů, který přijme zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **TriggerData** fronty identifikované polem *QName* ; podrobnosti o tomto atributu viz “Atributy pro fronty” na stránce 827 . Obsah těchto dat nemá význam pro správce front.

V produktu z/OS, pro aplikaci CICS spuštěnou pomocí transakce CKTI, tyto informace nejsou použity.

Délka tohoto pole je dána hodnotou MQ_TRIGGER_DATA_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 64 prázdných znaků v jiných programovacích jazycích.

ApplType (MQLONG)

Identifikuje charakter programu, který má být spuštěn, a je použit aplikací monitor spouštěčů, která přijme zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ApplType** objektu procesu určeného polem *ProcessName* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu” na stránce 862 . Obsah těchto dat nemá význam pro správce front.

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
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_BATCH
aplikace pro dávkové úlohy

MQAT_BROKER
Aplikace zprostředkovatele

MQAT_CICS
CICS .

MOST MQAT_CICS_BRIDGE
CICS bridge .

MQAT_CICS_VSE
CICS/VSE .

MQAT_DOS
IBM MQ MQI client na PC DOS.

MQAT_IMS
IMS .

MOST 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 Agent.

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
Aplikace neznámého typu.

UŽIVATEL MQAT_USER
Uživatelsky definovaný typ aplikace.

MQAT_VOS
Aplikace Stratus VOS.

MQAT_WINDOWS
16bitová aplikace Windows .

POČ MQAT_WINDOWS_NT
32bitovou aplikaci Windows .

MQAT_WLM
Aplikace správce pracovní zátěže produktu z/OS .

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)

Jedná se o znakový řetězec identifikující aplikaci, která má být spuštěna, a kterou používá aplikace pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ApplId** objektu procesu určeného polem *ProcessName* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu” na stránce 862 . Obsah těchto dat nemá význam pro správce front.

Význam *ApplId* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný serverem IBM MQ vyžaduje, aby byl *ApplId* název spustitelného programu. Níže uvedené poznámky se vztahují na uvedená prostředí:

- V systému z/OSje *ApplId* :
  - Identifikátor transakce CICS , pro aplikace spuštěné pomocí transakce monitoru spouštěčů CICS CKTI
  - Identifikátor transakce IMS , pro aplikace spuštěné pomocí monitoru spouštěčů IMS CSQQTRMN
- Na systémech Windows může mít název programu předponu jednotky a cesty k adresáři.
- V systému IBM imůže být název programu uvozeno názvem knihovny a znakem/.
- V systému AIX and Linuxmůže být název programu uveden jako předpona cesty k adresáři.

Délka tohoto pole je dána hodnotou MQ_PROCESS_APPL_ID_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 256 prázdných znaků v jiných programovacích jazycích.

EnvData (MQCHAR128)

Jedná se o znakový řetězec, který obsahuje informace související s prostředím týkající se aplikace, která má být spuštěna, a kterou používá aplikace pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **EnvData** objektu procesu určeného polem *ProcessName* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu” na stránce 862 . Obsah těchto dat nemá význam pro správce front.

V systému z/OS, pro aplikaci CICS spuštěnou pomocí transakce CKTI nebo aplikaci IMS , která má být spuštěna pomocí transakce CSQQTRMN, se tyto informace nepoužijí.

Délka tohoto pole je dána hodnotou MQ_PROCESS_ENV_DATA_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 128 prázdných znaků v jiných programovacích jazycích.

UserData (MQCHAR128)

Jedná se o znakový řetězec, který obsahuje informace o uživateli související s aplikací ke spuštění, a používá se aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **UserData** objektu procesu určeného polem *ProcessName* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu” na stránce 862 . Obsah těchto dat nemá význam pro správce front.

Pro produkt Microsoft Windowsnesmí znakový řetězec obsahovat uvozovky, pokud má být definice procesu předána produktu **runmqtrm**.

Délka tohoto pole je dána hodnotou MQ_PROCESS_USER_DATA_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v 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 536. 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

Tabulka 536. Pole v tabulce MQTMC2 (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.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)

Identifikátor struktury.

Hodnota musí být:

ID_STRUKTURY MQTC_STRUCT
Identifikátor struktury zprávy spouštěče (znakový formát).

Pro programovací jazyk C je také definována konstanta MQTMC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQTMC_STRUC_ID, ale je to pole znaků místo řetězce.

Verze (MQCHAR4)

Číslo verze struktury.

Hodnota musí být:

MQTMC_VERSION_2
Struktura zpráv spouštěcího impulsu verze 2 (znaková formát).

Pro programovací jazyk C je také definována konstanta MQTMC_VERSION_2_ARRAY ; má stejnou hodnotu jako MQTMC_VERSION_2, ale je to pole znaků místo řetězce.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQTMC_VERSION
Aktuální verze struktury zprávy spouštěče (ve znakovém formátu).

QName (MQCHAR48)
Název spuštěné fronty.

Viz pole *QName* ve struktuře MQTM.

ProcessName (MQCHAR48)
Název objektu procesu.

Viz pole *ProcessName* ve struktuře MQTM.

TriggerData (MQCHAR64)
Data spouątěče.

Viz pole *TriggerData* ve struktuře MQTM.

ApplType (MQCHAR4)
Typ aplikace.

Toto pole vždy obsahuje mezery, bez ohledu na hodnotu v poli *ApplType* ve struktuře MQTM původní zprávy spouštěče.

ApplId (MQCHAR256)

Identifikátor aplikace.

Viz pole *ApplId* ve struktuře MQTM.

EnvData (MQCHAR128)
Data prostředí.

Viz pole *EnvData* ve struktuře MQTM.

UserData (MQCHAR128)
Uživatelská data.

Viz pole *UserData* ve struktuře MQTM.

QMgrName (MQCHAR48)
Název správce front.

Jedná se o název správce front, v němž 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 537. 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)

Jedná se o identifikátor struktury. Hodnota musí být:

ID_STRUKTURY MQWIH_
Identifikátor pro strukturu záhlaví informací o práci.

Pro programovací jazyk C je také definován konstantní MQWIH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQWIH_STRUC_ID, ale je to pole znaků namísto řetězce.

Počáteční hodnota tohoto pole je MQWIH_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQWIH_VERSION_1
Struktura záhlaví pracovních informací Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQWIH_CURRENT_VERSION
Aktuální verze struktury záhlaví pracovních informací.

Počáteční hodnota tohoto pole je MQWIH_VERSION_1.

StrucLength (MQLONG)

Jedná se o délku struktury MQWIH. Hodnota musí být:

MQWIH_LENGTH_1
Délka struktury záhlaví pracovních informací version-1 .

Následující konstanta uvádí délku aktuální verze:

AKTUÁLNÍ_DÉLKA MQWIH_CURRENT_LENGTH
Délka aktuální verze struktury záhlaví pracovních informací.

Počáteční hodnota tohoto pole je MQWIH_LENGTH_1.

Kódování (MQLONG)

Určuje číselné kódování dat, která se řídí strukturou MQWIH. Nevztahuje se na číselná data v samotné struktuře MQWIH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

CodedCharSetId (MQLONG)

Určuje identifikátor znakové sady pro data, která následují strukturu MQWIH. Nevztahuje se na znaková data v samotné struktuře MQWIH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Můžete použít následující speciální hodnotu:

MQCSI_INHERIT
Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota MQCCSI_INHERIT není vrácena voláním MQGET.

Hodnotu MQCCSI_INHERIT nelze použít, je-li hodnota pole *PutApplType* v deskriptoru MQMD MQAT_BROKER.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

Formát (MQCHAR8)

Určuje název formátu dat, která následují za strukturou MQWIH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro pole *Format* v produktu MQMD.

Délka tohoto pole je dána hodnotou MQ_FORMAT_LENGTH. Počáteční hodnota tohoto pole je MQFMT_NONE.

Příznaky (MQLONG)

Hodnota musí být:

MQWIH_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQWIH_NONE.

ServiceName (MQCHAR32)

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)

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)

Jedná se o token zprávy, který jednoznačně identifikuje zprávu.

V případě 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.

Rezervováno (MQCHAR32)

Jedná se o 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 538. 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)

Jedná se o identifikátor struktury. Hodnota musí být:

ID_STRUKTURY MQXP_STRUCTION_ID
Identifikátor struktury výstupního parametru.

Pro programovací jazyk C je také definována konstanta MQXP_STRUC_ID_ARRAY; hodnota má stejnou hodnotu jako MQXP_STRUC_ID, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro ukončení.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQXP_VERSION_1
Číslo verze pro výstupní parametr-blok struktury.

Poznámka: Když je představena nová verze této struktury, rozvržení existující součásti se nezmění. Uživatelská procedura musí proto zkontrolovat, zda je číslo verze rovné nebo větší než nejnižší verze, která obsahuje pole, která má uživatelská procedura použít.

Toto je vstupní pole pro ukončení.

ExitId (MQLONG)

Toto je nastaveno na vstupu do uživatelské procedury a označuje typ uživatelské procedury:

UŽIVATELSKÁ PROCEDURA MQXT_API_CROSSING_EXIT
Uživatelská procedura přeletu rozhraní API pro produkt CICS.

Toto je vstupní pole pro ukončení.

ExitReason (MQLONG)

Toto je nastaveno na vstupu do uživatelské procedury. Pro uživatelskou proceduru přejezdu rozhraní API označuje, zda je rutina volána před nebo po provedení volání rozhraní API:

MQXR_PŘED
Před spuštěním rozhraní API.

MQXR_PO
Po provedení rozhraní API.

Toto je vstupní pole pro ukončení.

ExitResponse (MQLONG)

Hodnota je nastavena uživatelskou procedurou pro komunikaci s volajícím. Jsou definovány tyto hodnoty:

MQXCC_OK
Ukončení bylo úspěšně dokončeno.

FUNKCE MQXCC_SUPPRESS_FUNCTION
Potlačit funkci.

Je-li tato hodnota nastavena pomocí uživatelské procedury překřížení rozhraní API s názvem *před* voláním rozhraní API, volání API se neprovede. Volání *CompCode* pro volání je nastaveno na hodnotu MQCC_FAILED, *Reason* je nastaven na hodnotu MQRC_SUPPRESDAT_BY_EXIT a všechny ostatní parametry zůstanou zachovány jako jejich ukončení.

Je-li tato hodnota nastavena pomocí uživatelské procedury překřížení rozhraní API s názvem *po* volání rozhraní API, je správce front ignorován.

FUNKCE MQXCC_SKIP_FUNCTION
Vynechat funkci.

Je-li tato hodnota nastavena prostřednictvím uživatelské procedury pro přechod přes rozhraní API s názvem *před* voláním rozhraní API, volání API se neprovede; zbývající parametry *CompCode* a *Reason* a všechny ostatní parametry zůstanou zachovány jako konec.

Je-li tato hodnota nastavena pomocí uživatelské procedury překřížení rozhraní API s názvem *po* volání rozhraní API, je správce front ignorován.

Jedná se o výstupní pole z uživatelské procedury.

ExitCommand (MQLONG)

Toto pole je nastaveno na vstupu do uživatelské procedury. Identifikuje volání rozhraní API, které způsobilo vyvolání procedury ukončení:

ZPĚTNÉ VOLÁNÍ MQXC_
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.

Počet ExitParm(MQLONG)

Toto pole je nastaveno na vstupu do uživatelské procedury. Obsahuje počet parametrů, které volání MQ přijímá.

Tabulka 539. Počet parametrů pro jednotlivé 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.

Rezervováno (MQLONG)

Jedná se o vyhrazené pole. Jeho hodnota není významná pro ukončení.

Oblast ExitUser(MQBYTE16)

Jedná se o pole, které je k dispozici pro uživatelskou proceduru. Inicializuje se na binární nulu pro délku pole před prvním vyvoláním uživatelské procedury pro úlohu a poté jsou všechny změny provedené v tomto poli provedené uživatelskou procedurou zachovány v rámci vyvolání uživatelské procedury. Je definována následující hodnota:

MQXA_NONE
Žádné informace o uživateli.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQXUA_NONE_ARRAY; má stejnou hodnotu jako MQXUA_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána proměnnou MQ_EXIT_USER_AREA_LENGTH. Jedná se o 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 540. 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

Tabulka 540. Pole v MQXQH pro MQXQH (pokračování)

Název a popis pole	Název konstanty	Počáteční hodnota konstanty (pokud existuje)
MsgDesc (původní deskriptor zprávy)	Stejné názvy a hodnoty jako MQMD; viz Tabulka 501 na stránce 420	-

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 541. 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.
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

Tabulka 541. Hodnoty použité pro pole v odděleném deskriptoru MQMD (pokračování)

Pole v odděleném deskriptoru MQMD	Použitá hodnota
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)

Jedná se o identifikátor struktury. Hodnota musí být:

ID STRUKTURY MQXQ_STRUCTION_ID
Identifikátor pro strukturu záhlaví přenosové fronty.

Pro programovací jazyk C je také definována konstanta MQXQH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQXQH_STRUC_ID, ale je to pole znaků místo řetězce.

Počáteční hodnota tohoto pole je MQXQH_STRUC_ID.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQXQH_VERSION_1
Číslo verze pro strukturu záhlaví přenosové fronty.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQXQHL_AKTUÁLNÍ_VERZE
Aktuální verze struktury záhlaví přenosové fronty.

Počáteční hodnota tohoto pole je MQXQH_VERSION_1.

RemoteQName (MQCHAR48)

Jedná se o název fronty zpráv, která je zdánlivým konečným místem určení zprávy (může se ukázat, že se nejedná o konečné místo určení, pokud je například tato fronta definována v *RemoteQMgrName* jako lokální definice jiné vzdálené fronty).

Pokud se jedná o zprávu distribučního seznamu (to znamená, že pole *Format* v deskriptoru vložené zprávy je MQFMT_DIST_HEADER), je *RemoteQName* prázdný.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Název RemoteQMgr(MQCHAR48)

Jedná se o název správce front nebo skupiny sdílení front, která vlastní frontu, která je zdánlivě konečným cílem zprávy.

Je-li zpráva zprávou rozdělovníku, *RemoteQMgrName* je prázdné.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

MsgDesc (MQMD1)

Jedná se o vložený deskriptor zpráv a je to kopie deskriptoru MQMD deskriptoru zpráv, která byla zadána jako parametr **MsgDesc** v rámci volání MQPUT nebo MQPUT1 , když byla zpráva původně vložena do vzdálené fronty.

Poznámka: Jedná se o version-1 MQMD.

Počáteční hodnoty polí v této struktuře jsou stejné jako počáteční hodnoty v rámci struktury MQMD.

Volání funkcí

Tato sekce poskytuje informace o všech možných voláních MQI. Popisy, syntaxe, informace parametrů, poznámky k použití a vyvolání jazyků pro každý možný jazyk jsou uvedeny pro každý z různých volání.

Související odkazy

z/OS Příklady výstupu CEDF z volání MQI

Popisy volání

Tento oddíl popisuje volání MQI.

- “MQOPEN-Otevřít objekt” na stránce 726
- “MQPUT-Vložit zprávu” na stránce 743
- “MQPUT1 -Vložení jedné zprávy” na stránce 756
- “MQSET-Nastavit atributy objektu” na stránce 766
- “MQSETMP-nastavení vlastnosti zprávy” na stránce 773
- “MQSTAT-Načíst informace o stavu” na stránce 777
- “MQMHBUF-Převést popisovač zprávy do vyrovnávací paměti” na stránce 722
- “MQSUB-Registrace odběru” na stránce 781
- “MQSUBRQ-Požadavek na odběr” na stránce 788

Online nápověda na platformách UNIX ve formě stránek *man* je k dispozici pro tato volání.

Poznámka: Volání přidružená k převodu dat, MQXCNVC a MQ_DATA_CONV_EXIT, jsou v “Uživatelská procedura konverze dat” na stránce 897.

Konvence použité v popisech volání

U každého volání tato kolekce témat obsahuje popis parametrů a použití volání ve formátu, který je nezávislý na programovacím jazyku. Následuje typická vyvolání volání a typická deklarace parametrů, v každém z podporovaných programovacích jazyků.

Důležité: Při kódování volání rozhraní API produktu IBM MQ je třeba zajistit, aby byly poskytnuty všechny relevantní parametry (jak je popsáno v následujících sekcích). Pokud tak neučiníte, může dojít k nepředvídatelným výsledkům.

Popis každého volání obsahuje následující sekce:

Název volání
Název volání, za nímž následuje stručný popis účelu volání.

Parametry
Pro každý parametr je za názvem následován jeho datový typ v závorkách (). a jeden z následujících:

Vstup
Když zavoláte, dodáte informace do parametru.

výstup
Správce front vrátí informace v rámci parametru po dokončení nebo selhání volání.

Vstup a výstup
Když zavoláte, dodáte informace do parametru a správce front změní informace, když se volání dokončí nebo selže.

Příklad:

Compcode (MQLONG)-výstup

V některých případech je datový typ strukturou. Ve všech případech je v produktu “Elementární datové typy” na stránce 235více informací o datovém typu nebo struktuře.

Poslední dva parametry v každém volání jsou kód dokončení a kód příčiny. Kód dokončení označuje, zda bylo volání dokončeno úspěšně, částečně nebo vůbec. 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 866.

Poznámky k použití
Další informace o volání popisují, jak ji použít a jaká omezení jejího použití používají.

Vyvolání jazyka asembleru
Typické vyvolání volání a deklarace jeho parametrů v jazyku assembler.

Vyvolání jazyka C
Typické vyvolání volání a prohlášení o jeho parametrech v C.

Vyvolání COBOL
Typické vyvolání volání a deklarace jeho parametrů v jazyce COBOL.

Vyvolání PL/I
Typické vyvolání volání a deklarace jeho parametrů v PL/I.

Všechny parametry jsou předávány odkazem.

Vyvolání Visual Basic
Typické vyvolání volání a deklarace jeho parametrů ve Visual Basic.

Ostatní konvence notace jsou:

Konstanty
Názvy konstant se zobrazují velkými písmeny, např. MQOO_OUTPUT. Sada konstant, které mají stejnou předponu, se zobrazí takto: MQIA_ *. Informace o hodnotě konstanty viz “Konstanty” na stránce 60 .

Pole
V některých voláních jsou parametry pole znakových řetězců, které nemají pevné velikosti. V popisech těchto parametrů představuje malá písmena n číselnou konstantu. Když kódíte deklaraci pro tento parametr, nahraďte hodnotu n numerickou hodnotou, kterou požadujete.

Použití volání v jazyku C

Parametry, které jsou *pouze vstup* a typu MQHCONN, MQHOBJ, MQHMSG nebo MQLONG, jsou předávány hodnotou. Pro všechny ostatní parametry je hodnota parametru předávána hodnotou parametru *adresa* .

Nemusíte uvádět všechny parametry, které jsou předávány zadáním adresy pokaždé, když vyvoláte funkci. Tam, kde nepotřebujete konkrétní parametr, zadejte jako parametr při vyvolání funkce ukazatel null místo adresy dat parametru. Parametry, pro které je to možné, jsou identifikovány v popisech volání.

Není vrácen žádný parametr jako hodnota volání; v terminologii C to znamená, že všechna volání vrátí `void`.

Deklarace parametru Vyrovnávací paměť

Parametry **MQGET**, **MQPUT**a **MQPUT1** mají jeden parametr, který má nedefinovaný datový typ: parametr *Buffer* . Tento parametr použijte k odeslání a přijetí dat zprávy aplikace.

Parametry tohoto řazení jsou zobrazeny v příkladech C jako pole MQBYTE. Parametry můžete deklarovat tímto způsobem, ale obvykle je vhodnější deklarovat je jako konkrétní strukturu, která popisuje rozvržení dat ve zprávě. Prototyp funkce deklaruje parametr jako ukazatel na neobsazený, takže můžete zadat adresu libovolného druhu dat jako parametru při vyvolání volání.

Pointer-to-void je ukazatel na data nedefinovaného formátu. Je definován jako:

```
typedef void *PMQVOID;
```

MQBACK-Vrátit změny

Volání MQBACK označuje správci front, že všechny zprávy typu get a put, které se vyskytly od posledního bodu synchronizace, jsou vráceny.

Zprávy, které byly vloženy jako součást pracovní jednotky, se odstraní; zprávy načtené jako součást pracovní jednotky jsou obnoveny ve frontě.

- V systému z/OSse toto volání používá pouze pro dávkové programy (včetně dávkových DL/I programů IMS).

Syntaxe

MQBACK (*Hconn*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

NEVYŘÍZENÉ MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace back-out je nevyřízený.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není platné v prostředí.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

MQRC_OUTCOME_MIXED
(2123, X'84B') Výsledek operace commit nebo back-out je smíšený.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin .

Poznámky k použití

1. Toto volání můžete použít pouze v případě, že správce front sám koordinuje pracovní jednotku. To může být:
   - Lokální jednotka práce, kde změny ovlivní pouze prostředky MQ .
   - Globální jednotka práce, kde mohou změny ovlivnit prostředky patřící jiným správcům prostředků a které ovlivňují prostředky MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz "MQBEGIN-Begin unit of work" na stránce 626.
2. V prostředích, kde správce front nekoordinuje jednotku práce, použijte místo MQBACK odpovídající zpětné volání. Prostředí může také podporovat implicitní vrácení zpět v důsledku abnormálního ukončení aplikace.
   - V systému z/OSpoužijte následující volání:
     - Dávkové programy (včetně dávkových DL/I programů produktu IMS) mohou použít volání MQBACK, pokud má jednotka práce vliv pouze na prostředky MQ . However, if the unit of work affects both MQ resources and resources belonging to other resource managers (for example, Db2), use the SRRBACK call provided by the z/OS Recoverable Resource Service (RRS). Volání SRRBACK vrací změny prostředků náležejících ke správcům prostředků, kteří byli povoleni pro koordinaci RRS.
     - Aplikace produktu CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k zálohování jednotky práce. Nepoužívejte volání MQBACK pro aplikace produktu CICS .
     - Aplikace produktu IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. produkt `ROLB` , aby odvrátila jednotku práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové DL/I programy).
   - V systému IBM ipoužijte toto volání pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **`CMTSCOPE(*JOB)`** nesmí být vydán pro úlohu.
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce "MQDISC-Odpojení správce front" na stránce 678 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
   - Hodnoty polí *`GroupId`*, *`MsgSeqNumber`*, *`Offset`*a *`MsgFlags`* v MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

   Správce front uchovává *tři* sady informací o skupinách a segmentech, jednu sadu pro každou z následujících možností:
   - Poslední úspěšné volání MQPUT (může být součástí jednotky práce).
   - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí jednotky práce).

- Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (to nemůže být součástí pracovní jednotky).

5. Informace přidružené k volání MQGET se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQGET pro daný popisovač fronty v aktuální pracovní jednotce.

   Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnovenou skupinovou a segmentovou informaci, pokud je jednotka práce zálohována.

   Obnova informace o skupině a segmentu na její předchozí hodnotu, když je zálohována jednotka práce, umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů přes několik jednotek práce a restartovat ve správném bodu ve skupině zpráv nebo v logické zprávě, pokud se jedna z jednotek práce nezdaří.

   Použití několika jednotek práce může být výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv ve správném okamžiku, pokud dojde k selhání systému.

   Podrobnosti o restartování ve správném bodu po selhání systému naleznete v části MQPMO_LOGICAL_ORDER popsané v části “MQPMO-Volby vložení zprávy” na stránce 496a v části MQGMO_LOGICAL_ORDER popsané v části “MQGMO-Volby získání zprávy” na stránce 366.

   Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce.
6. Jednotka práce má stejný rozsah jako manipulátor připojení. Všechny volání MQ , které ovlivňují konkrétní jednotku práce, musí být provedeny pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz parametr **Hconn** popsaný v tématu “MQCONN-Připojení správce front” na stránce 656 .
7. Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
8. Dlouhá-spuštěná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydává výzvu k potvrzení nebo vrácení, může vyplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Pro ochranu proti této možnosti musí administrátor nastavit atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily 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 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-Begin unit of work

Volání MQBEGIN zahajuje transakci, která je koordinována správcem front a která může zahrnovat externí správce prostředků.

Syntaxe

MQBEGIN (*Hconn*, *BeginOptions*, *Kód_dokončení*, *Důvod*)

Parametry

Hconn

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

Hconn musí být nesdílený popisovač připojení. Je-li zadán popisovač sdíleného připojení, volání selže s kódem příčiny MQRC_HCONN_ERROR. Další informace o sdílených a nesdílených manipulátorů najdete v popisu voleb MQCNO_HANDLE_SHARE_ * v produktu “MQCNO-Volby připojení” na stránce 315 .

BeginOptions
Typ: MQBO-input/output

Jedná se o volby, které řídí akci MQBEGIN, jak je popsáno v tématu “MQBO-volby zahájení” na stránce 278.

Nejsou-li vyžadovány žádné volby, programy napsané v assembleru C nebo S/390 mohou uvádět adresu parametru null místo určení adresy struktury MQBO.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_NO_EXTERNAL_PARTICIPANTS
(2121, X'849 ') Nejsou registrovány žádné zúčastněné správce prostředků.

MQRC_PARTICIPANT_NOT_AVAILABLE
(2122, X'84A') Zúčastněné správce prostředků není k dispozici.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_BO_ERROR
(2134, X'856 ') Struktura začátku-volby není platná.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není platné v prostředí.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

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 najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Pomocí volání MQBEGIN spusťte jednotku práce, kterou koordinuje správce front, a která může zahrnovat změny prostředků vlastněných jinými správci prostředků. Správce front podporuje tři typy jednotek práce:
   - **Správa front-koordinovaná lokální jednotka práce**: Pracovní jednotka, v níž je správce front jediným účastníkem správce prostředků, a správce front tak vystupuje jako koordinátor jednotek práce.
     - Chcete-li spustit tento typ pracovní jednotky, určete volbu MQPMO_SYNCPOINT nebo MQGMO_SYNCPOINT na první volání MQPUT, MQPUT1nebo MQGET v pracovní jednotce.
     - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, použijte volání MQCMIT nebo MQBACK.
   - **Správce front-koordinovaná globální transakce práce**: A unit of work in which the queue manager acts as the unit-of-work coordinator, both for MQ resources *a* for resources belonging to other resource managers. Tito správci prostředků spolupracují se správcem front, aby zajistili, že všechny změny prostředků v pracovní jednotce budou potvrzeny nebo vráceny společně.
     - Chcete-li spustit tento typ jednotky práce, použijte volání MQBEGIN.
     - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, použijte volání MQCMIT a MQBACK.
   - **Externě koordinovaná globální transakce**: Pracovní jednotka, v níž je správce front účastníkem, ale správce front nepracuje jako koordinátor jednotky práce. Místo toho je k dispozici externí koordinátor pracovní jednotky, se kterým spolupracuje správce front.
     - Chcete-li spustit tento typ pracovní jednotky, použijte příslušné volání poskytnuté koordinátorem externí jednotky práce.

       Je-li volání MQBEGIN použito pro pokus o spuštění pracovní jednotky, volání se nezdaří s kódem příčiny MQRC_ENVIRONMENT_ERROR.
     - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, použijte volání operace commit a back-out poskytované koordinátorem externí jednotky práce.

       Pokud použijete volání MQCMIT nebo MQBACK k potvrzení nebo zpětné provedení pracovní jednotky, volání selže s kódem příčiny MQRC_ENVIRONMENT_ERROR.
2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce “MQDISC-Odpojení správce front” na stránce 678 .
3. Aplikace se může účastnit pouze jedné transakce v daném okamžiku. Volání MQBEGIN selže s kódem příčiny MQRC_UOW_IN_PROGRESS, pokud již existuje jednotka práce pro danou aplikaci, bez ohledu na typ jednotky práce, kterou má být.
4. Volání MQBEGIN není platné v prostředí klienta MQ MQI. Pokus o použití volání selhává s kódem příčiny MQRC_ENVIRONMENT_ERROR.
5. Je-li správce front koordinátorem jednotek práce pro globální jednotky práce, jsou správci prostředků, kteří se mohou podílet na pracovní jednotce, definovány v konfiguračním souboru správce front.
6. V systému IBM ijsou podporovány tyto tři typy pracovní jednotky:
   - **Koordinovaná lokální jednotka práce správce front** lze použít pouze tehdy, když definice vázaného zpracování neexistuje na úrovni úlohy, to znamená, že příkaz STRCMTCTL s argumentem **CMTSCOPE(*JOB)** nesmí být pro danou úlohu vydán.
   - **Koordinovaná globální jednotka práce správce front** není podporována.

- **Externě koordinované globální pracovní jednotky** lze použít pouze tehdy, když definice vázaného zpracování existuje na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** musí být vydán pro úlohu. Pokud byla tato akce provedena, operace IBM i COMMIT a ROLLBACK se vztahují na prostředky MQ a na prostředky patřící do jiných zúčastněných správců 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 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-Převedení vyrovnávací paměti na popisovač zprávy

Volání funkce MQBUFMH převede vyrovnávací paměť na popisovač zprávy a je inverzní k volání MQMHBUF.

Toto volání přebírá deskriptor zprávy a vlastnosti MQRFH2 ve vyrovnávací paměti a zpřístupňuje je prostřednictvím popisovače zprávy. Vlastnosti MQRFH2 v datech zprávy jsou volitelně odebrány. Pole *Encoding*, *CodedCharSetId*a *Format* deskriptoru zpráv se aktualizují, je-li to nutné, aby správně popisovaly obsah vyrovnávací paměti po odebrání vlastností.

Syntaxe

MQBUFMH (*Hconn*, *Hmsg*, *BufMsgHOpts*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota **Hconn** se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být ustanoveno platné připojení na podprocesu, který převádí vyrovnávací paměť na popisovač zprávy. Není-li ustanoveno platné připojení, volání selže při volání MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMQSG-vstup

Jedná se o popisovač zprávy, pro který je vyžadována vyrovnávací paměť. Hodnota byla vrácena předchozím voláním MQCRTMH.

BufMsgvolby HOpts
Typ: MQBMHO-vstup

Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou zpracovávány manipulátory zpráv z vyrovnávacích pamětí.

Podrobnosti viz "MQBMHO-Volby zpracování vyrovnávací paměti pro zprávu" na stránce 273.

MsgDesc
Typ: MQMD-I/O

Struktura *MsgDesc* obsahuje vlastnosti deskriptoru zpráv a popisuje obsah oblasti vyrovnávací paměti.

Ve 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 v kódování aplikace.

BufferLength
Typ: MQLONG-vstup

BufferLength je délka oblasti vyrovnávací paměti, v bajtech.

BufferLength z nulového počtu bajtů je platný a indikuje, že oblast vyrovnávací paměti neobsahuje žádná data.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup/výstup

Jedná se o volby, které řídí akci MQBEGIN, jak je popsáno v tématu "MQBEGIN-Begin unit of work" na stránce 626.

Buffer definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat byste měli zarovnat vyrovnávací paměť na 4bajtové hranici.

Pokud **Buffer** obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** na hodnoty odpovídající datům; to umožní převod dat, je-li to nutné.

Jsou-li vlastnosti nalezeny ve vyrovnávací paměti zpráv, mohou být odebrány později. Později budou k dispozici od obslužné rutiny zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel-to-void, což znamená, že adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BufferLength** nulový, **Buffer** není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assembler s hodnotou null.

DataLength
Typ: MQLONG-výstup

Délka vyrovnávací paměti, která může mít odebrané vlastnosti, v bajtech.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

CHYBA MQRC_BMHO_ERROR
(2489, X'09B9') Struktura obslužného programu vyrovnávací paměti pro zpracování zprávy není platná.

CHYBA MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

CHYBA MQRC_HMSG_ERROR
(2460, X'099C') Popisovač zprávy není platný.

CHYBA MQRC_MD_ERROR
(2026, X'07EA') Deskriptor zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Popisovač zprávy je již používán.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_RFH_ERROR
(2334, X'091E') Struktura MQRFH2 není platná.

CHYBA MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nebylo možné analyzovat.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

Volání MQBUFMH nelze zachytit pomocí uživatelských procedur rozhraní API-vyrovnávací paměť je převedena na popisovač zprávy v prostoru aplikace; volání není k dispozici pro 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-Správa zpětného volání

Volání MQCB registruje zpětné volání pro zadaný popisovač objektu a řídí aktivaci a změny pro zpětné volání.

Zpětné volání je část kódu (zadaná buď jako název funkce, kterou lze dynamicky propojit, nebo jako ukazatel funkce) volanou produktem IBM MQ , když dojde k určitým událostem.

Chcete-li použít MQCB a MQCTL na klientu, musíte být připojeni k serveru, kde vyjednaný parametr **SHARECNV** kanálu souhlasil s nenulovým hodnotou.

Typy zpětného volání, které lze definovat, jsou:

Spotřebitel zpráv.
Funkce zpětného volání spotřebitele zpráv se volá tehdy, je-li na manipulátoru objektu dostupná zpráva splňující zadaná kritéria výběru.

Na každém popisovači objektu může být registrována pouze jedna funkce zpětného volání. Má-li být jedna fronta čtena s více kritérii výběru, musí být fronta otevřena vícekrát a musí být registrována funkce spotřebitele na každém popisovači.

obslužná rutina událostí
Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí zpětného volání.

Funkce je volána, když se vyskytne podmínka události, například správce front nebo zastavení připojení nebo uvedení do klidového stavu.

Funkce není volána pro podmínky, které jsou specifické pro jednotlivého spotřebitele zpráv, například MQRC_GET_INHIBITED; je volán, avšak pokud funkce zpětného volání neskončí normálně.

Syntaxe

MQCB (*Hconn*, *Operation*, *CallbackDesc*, *Hobj*, *MsgDesc*, *GetMsgOpts*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS for CICS můžete pro produkt *MQHC_DEF_HCONN* určit následující speciální hodnotu pro použití manipulátoru připojení přidruženého k této prováděcí jednotce.

Operace
Typ: MQLONG-vstup

Operace se zpracovává na zpětné volání definované pro zadaný popisovač objektu. Je třeba určit jednu z následujících voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát 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 má být volána, a kritéria výběru, která se mají použít.

Je-li již definována funkce zpětného volání pro popisovač objektu, definice je nahrazena. Je-li při nahrazování zpětného volání zjištěna chyba, bude zrušena registrace funkce.

Je-li zpětné volání zaregistrováno v rámci stejné funkce zpětného volání, ve které byla zrušena registrace, je toto volání považováno za operaci nahrazení; počáteční nebo poslední volání se nevyvolá.

Příkaz MQOP_REGISTER lze použít s parametrem MQOP_SUSPEND nebo MQOP_RESUME.

MQOP_DEREGISTRACI
Zastavte spotřebovávání zpráv pro popisovač objektu a odeberte popisovač z těch vhodných pro zpětné volání.

Zpětné volání se automaticky zruší, je-li přidružený popisovač uzavřen.

Je-li MQOP_DEREGISTER volán ze zákaznického serveru a zpětné volání má definované ukončení volání, je vyvoláno po návratu ze strany spotřebitele.

Je-li tato operace vydána pro objekt *Hobj* bez registrovaného odběratele, volání se vrátí s hodnotou MQRC_CALLBACK_NOT_REGISTERED.

MQOP_SUSPEND
Pozastaví příjem zpráv pro popisovač objektu.

Je-li tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí při pozastavení události nepřijímá události a všechny události, které jste minuli v pozastaveném stavu, nejsou při pokračování operace poskytnuty.

Během pozastavení funkce odběratele pokračuje v získávání zpětných volání typu ovládacího prvku.

MQOP_RESUME
Obnovte příjem zpráv pro popisovač objektu.

Je-li tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí při pozastavení události nepřijímá události a všechny události, které jste minuli v pozastaveném stavu, nejsou při pokračování 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 požadován pouze pro volbu MQOP_REGISTER; pokud deskriptor není povinný, adresa parametru předaná může mít hodnotu null.

HOBJ
Typ: MQHOTBJ-vstup

Tento manipulátor představuje přístup, který byl vytvořen objektu, ze kterého má být zpráva spotřebována. Jedná se o popisovač, který byl vrácen z předchozího volání MQOPEN nebo MQSUB (v parametru **Hobj**).

Hobj není vyžadována při definování rutiny obslužné rutiny událostí (MQCBT_EVENT_HANDLER) a měla by být zadána jako MQHO_NONE.

Pokud byl produkt *Hobj* vrácen z volání MQOPEN, musí být fronta otevřena s jednou nebo více z následujících voleb:

- MQO_INPUT_SHARED
- MQO_INPUT_EXCLUSIVE
- MQO_INPUT_AS_Q_DEF
- MQOOK_BROWSE

MsgDesc
Typ: MQMD-vstup

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy.

Parametr **MsgDesc** definuje atributy zpráv požadovaných odběratelem a verze MQMD, která má být předána spotřebiteli zpráv.

Parametry *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*a *Offset* v deskriptoru MQMD se používají pro výběr zpráv v závislosti na tom, které volby jsou určeny parametrem **GetMsgOpts** .

Volby *Encoding* a *CodedCharSetId* se používají ke konverzi zpráv, pokud zadáte volbu MQGMO_CONVERT.

Podrobnosti viz MQMD .

Produkt *MsgDesc* se používá pro MQOP_REGISTER a v případě, že požadujete jiné hodnoty než výchozí hodnoty pro jakákoli pole. *MsgDesc* se nepoužívá pro obslužnou rutinu událostí.

Pokud deskriptor není požadován, poslaná adresa parametru může mít hodnotu null.

Všimněte si, že pokud je více spotřebitelů registrováno ve stejné frontě s překrývajícími se selektory, zvolený spotřebitel pro každou zprávu není definován.

GetMsgOpts
Typ: MQGMO-vstup

Parametr **GetMsgOpts** určuje, jak bude spotřebitel zpráv přijímat zprávy. Všechny volby tohoto parametru mají význam, jak je popsáno v části “MQGMO-Volby získání zprávy” na stránce 366, je-li použito na volání MQGET, s výjimkou:

SIGNÁL MQGMO_SET_DATA
Tato volba není povolena.

MQGMO_BROWSE_FIRST, MQGMO_BROWSE_NEXT, MQGMO_MARK_ *
Pořadí zpráv doručených uživateli prohlížení je určeno kombinací těchto voleb. Mezi důležité kombinace patří:

NEJPRVE MQGMO_BROWSE_FIRST
První zpráva ve frontě se doručí opakovaně spotřebiteli. To je užitečné, když spotřebitel destruktivně spotřebovává zprávu ve zpětném volání. Použijte tuto volbu s opatrností.

PŘÍŠTĚ MQGMO_BROWSE_NEXT
Spotřebiteli je dána každá zpráva ve frontě, od aktuální pozice kurzoru, dokud není dosaženo konce fronty.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT
Kurzor se resetuje na začátek fronty. Spotřebitel pak dostane každou zprávu, dokud se kurzor nedostane na konec fronty.

MQGMO_BROWSE_FIRST + MQGMO_MARK_ *
Od začátku fronty je spotřebiteli dána první neoznačená zpráva ve frontě, která je poté označena pro tohoto spotřebitele. Tato kombinace zajistí, aby spotřebitel mohl přijímat nové zprávy za aktuální bod kurzoru za aktuální.

MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
Počínaje pozicí kurzoru je spotřebitel přidělen další neoznačenou zprávu ve frontě, která je poté označena pro tohoto spotřebitele. Tuto kombinaci používejte s pečlivostí, protože zprávy lze přidávat do fronty za aktuální pozicí kurzoru.

MQGMO_BROWSE_FIRST + MQGMO_BROWSE_NEXT + MQGMO_MARK_ *
Tato kombinace není povolena. Při použití volání je vrácen parametr MQRC_OPTIONS_ERROR.

MQGMO_NO_WAIT, MQGMO_WAIT a WaitInterval
Tyto volby řídí způsob vyvolání odběratele.

MQGMO_NO_WAIT
Spotřebitel není nikdy volán s MQRC_NO_MSG_AVAILABLE. Spotřebitel je volán pouze pro zprávy a události.

MQGMO_WAIT s hodnotou nula WaitInterval ,
Kód MQRC_NO_MSG_AVAILABLE je předán spotřebiteli, pokud nejsou k dispozici žádné zprávy a buď byl spotřebitel spuštěn, nebo byl spotřebitel dodán alespoň jednu zprávu od posledního kódu příčiny "no messages".

Tím zabráníte tomu, aby spotřebitel byl ve smyčce v zaneprázdněném cyklu, je-li zadán nulový interval čekání.

MQGMO_WAIT a kladná hodnota WaitInterval
Spotřebitel je volán po uvedeném intervalu čekání s kódem příčiny MQRC_NO_MSG_AVAILABLE. Toto volání se provádí bez ohledu na to, zda byly odběrateli doručovány nějaké zprávy. To umožní uživateli provést zpracování prezenčního signálu nebo zpracování dávkového zpracování.

MQGMO_WAIT a WaitInterval z MQWI_UNLIMITED
Tento parametr určuje nekonečné čekání před vrácením MQRC_NO_MSG_AVAILABLE. Spotřebitel není nikdy volán s MQRC_NO_MSG_AVAILABLE.

Produkt *GetMsgOpts* se používá pouze pro MQOP_REGISTER a v případě, že požadujete jiné hodnoty než výchozí hodnoty pro jakákoli pole. *GetMsgOpts* se nepoužívá pro obslužnou rutinu událostí.

Pokud se *GetMsgOpts* nepožaduje, předaný adresa parametru může mít hodnotu null. Použití tohoto parametru je stejné jako uvedení MQGMO_DEFAULT spolu s MQGMO_FAIL_IF_QUIESCING.

Je-li v rámci struktury MQGMO zadán popisovač vlastností zprávy, je v rámci struktury MQGMO, která je předána do zpětného volání spotřebitele, předána kopie. Při návratu z volání MQCB může aplikace odstranit popisovač vlastností zprávy.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kódy příčiny v následujícím seznamu jsou ty, které může správce front vrátit pro parametr **Reason** .

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb pro převod dat.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno 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 činnost, protože neexistuje žádné registrované zpětné volání.

CHYBA MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') Musí být zadán buď *CallbackFunction* , nebo *CallbackName* , ale ne obojí.

MQRC_CALLBACK_TYPE_ERROR
(2483, X'9B3') Nesprávné pole typu zpětného volání.

CHYBA MQRC_CBD_OPTIONS_ERROR
(2484, X'9B4') Nesprávné pole voleb MQCBD.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CORRELA_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není platné v prostředí.

PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá informace o zablokování fronty.

KONFLIKT MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální jednotky konfliktu práce.

CHYBA MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb získání zprávy není platná.

FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Nekonzistentní specifikace procházení.

NEKONZISTENCE 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.

KONFLIKT MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s místní jednotkou práce.

CHYBA MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

CHYBA MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B4') Nesprávné pole *MaxMsgLength* .

CHYBA MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zprávy není platný.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Uvedený vstupní bod funkce nebyl nalezen v modulu.

MQRC_MODULE_INVALID
(2496, X'9C0') Modul byl nalezen, avšak je nesprávného typu; není 32bitový, 64bitový, nebo platnou dynamickou knihovnou odkazů.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Modul nebyl nalezen v cestě pro vyhledávání, nebo neměl oprávnění k načtení.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

CHYBA MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Nejsou k dispozici žádné zprávy.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Procházení kurzoru není umístěno na zprávě.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřená pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu byla od otevření změněna.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

CHYBA OPERACE MQRC_OPERATION_ERROR
(2206, X'89E') Nesprávný kód operace na volání rozhraní API.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA OBJEKTU 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_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má špatný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815 ') Signál nevyřízený pro tento popisovač.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora bodu synchronizace není k dispozici.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

CHYBA MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotek práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unit of work not available for the queue manager to use.

CHYBA MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Čekací interval v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Chybná verze dodávaného MQGMO.

VERZE MQRC_WRONG_MD_VERSION
(2257, X'8D1') Chybná verze dodaných MQMD.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

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 použije k získání zprávy.
2. MQCB lze použít k definování rutin zpětného volání před spuštěním spotřeby s rozhraním MQCTL nebo je lze použít v rámci rutiny zpětného volání.
3. Chcete-li použít funkci MQCB mimo rutinu zpětného volání, je třeba nejprve pozastavit spotřebu zpráv pomocí funkce MQCTL a pokračovat ve spotřebě po jejím použití.
4. MQCB není v rámci adaptéru IMS podporován.

Posloupnost zpětného volání odběratele zpráv

V průběhu životního cyklu spotřebitele můžete nakonfigurovat odběratele k vyvolání zpětného volání v klíčových bodech. Příklad:

- když je spotřebitel poprvé registrován,
- při spuštění připojení,
- když je připojení zastaveno a
- je-li odběratel deregistrován, ať už explicitně, nebo implicitně MQCLOSE.

Tabulka 542. Definice příkazu MQCTL

Sloveso	**Význam**
MQCTL (START)	Volání MQCTL pomocí operace MQOP_START
MQCTL (ZASTAVIT)	Volání MQCTL pomocí operace MQOP_STOP
MQCTL (ČEKÁNÍ)	Volání MQCTL pomocí operace MQOP_START_WAIT

To umožňuje spotřebiteli udržovat stav přidružený k odběrateli. Je-li aplikace požádána o zpětné volání, jsou pravidla pro vyvolání spotřebitele následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je volána vždy ve stejném podprocesu, jako volání MQCB (REGISTER).

SPUSTIT
Je vždy volán synchronně s použitím příkazu MQCTL (START).

- Všechna zpětná volání START jsou dokončeny před návratem příkazu MQCTL (START).

Je ve stejném vláknu jako doručení zprávy, je-li požadováno THREAD_AFFINITY.

Volání se spuštěním není garantováno, pokud například předchozí zpětné volání vyvolá MQCTL (STOP) během MQCTL (START).

ZASTAVIT
Po tomto volání nebudou po tomto volání doručeny žádné další zprávy nebo události, dokud není připojení znovu spuštěno.

Hodnota STOP je garantována, pokud byla aplikace dříve volána pro START, nebo zprávu nebo událost.

ZRUŠIT REGISTRACI
Je vždy posledním typem vyvolání zpětného volání.

Ujistěte se, že aplikace provádí inicializaci a vyčištění na základě podprocesů ve zpětných voláních START a STOP. Inicializaci a vyčištění založené na nevláknech můžete provést pomocí zpětných volání REGISTER a DEREGISTER.

Neuvádějte žádné hypotézy o životnosti a dostupnosti jiného podprocesu než toho, co je uvedeno. Nespoléhejte se například na podproces, který zůstává naživu nad posledním voláním funkce DECREGISTER. Podobně, pokud jste se rozhodli nepoužívat THREAD_AFFINITY, nepředpokládejte, že podproces existuje vždy, když je připojení spuštěno.

Pokud má vaše aplikace určité požadavky na charakteristiky vlákna, může to vždy vytvořit odpovídajícím způsobem podproces, pak použít MQCTL (WAIT). Tento efekt má efekt 'donarování' podprocesu na IBM MQ pro asynchronní doručování zpráv.

Použití připojení spotřebitele zpráv

V průběhu životního cyklu spotřebitele můžete nakonfigurovat odběratele k vyvolání zpětného volání v klíčových bodech. Příklad:

- když je spotřebitel poprvé registrován,
- při spuštění připojení,
- když je připojení zastaveno a
- je-li odběratel deregistrován, ať už explicitně, nebo implicitně MQCLOSE.

Tabulka 543. Definice příkazu MQCTL

Sloveso	**Význam**
MQCTL (START)	Volání MQCTL pomocí operace MQOP_START
MQCTL (ZASTAVIT)	Volání MQCTL pomocí operace MQOP_STOP
MQCTL (ČEKÁNÍ)	Volání MQCTL pomocí operace MQOP_START_WAIT

To umožňuje spotřebiteli udržovat stav přidružený k odběrateli. Je-li aplikace požádána o zpětné volání, jsou pravidla pro vyvolání spotřebitele následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je volána vždy ve stejném podprocesu, jako volání MQCB (REGISTER).

SPUSTIT
Je vždy volán synchronně s použitím příkazu MQCTL (START).

- Všechna zpětná volání START jsou dokončeny před návratem příkazu MQCTL (START).

Je ve stejném vláknu jako doručení zprávy, je-li požadováno THREAD_AFFINITY.

Volání se spuštěním není garantováno, pokud například předchozí zpětné volání vyvolá MQCTL (STOP) během MQCTL (START).

ZASTAVIT
Po tomto volání nebudou po tomto volání doručeny žádné další zprávy nebo události, dokud není připojení znovu spuštěno.

Hodnota STOP je garantována, pokud byla aplikace dříve volána pro START, nebo zprávu nebo událost.

ZRUŠIT REGISTRACI
Je vždy posledním typem vyvolání zpětného volání.

Ujistěte se, že aplikace provádí inicializaci a vyčištění na základě podprocesů ve zpětných voláních START a STOP. Inicializaci a vyčištění založené na nevláknech můžete provést pomocí zpětných volání REGISTER a DEREGISTER.

Neuvádějte žádné hypotézy o životnosti a dostupnosti jiného podprocesu než toho, co je uvedeno. Nespoléhejte se například na podproces, který zůstává naživu nad posledním voláním funkce DECREGISTER. Podobně, pokud jste se rozhodli nepoužívat THREAD_AFFINITY, nepředpokládejte, že podproces existuje vždy, když je připojení spuštěno.

Pokud má vaše aplikace určité požadavky na charakteristiky vlákna, může to vždy vytvořit odpovídajícím způsobem podproces, pak použít MQCTL (WAIT). Tento efekt má efekt 'donarování' podprocesu na 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 funkce zpětného volání pro obsluhu událostí a pro asynchronní spotřebu zpráv.

Definice volání MQCB_FUNKCE je k dispozici pouze pro popis parametrů předávaných funkci zpětného volání. Správcem front není poskytnut žádný vstupní bod s názvem MQCB_FUNCTION.

Specifikace aktuální 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
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX. V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro Hconn je určena následující hodnota:

Objekt MQHC_DEF_CONN

Výchozí popisovač připojení.

MsgDesc
Typ: MQMD-vstup

Tato struktura popisuje atributy načtené zprávy.

Podrobnosti viz “MQMD-Deskriptor zpráv” na stránce 419.

Verze předaný MQMD je stejná verze jako předaná volání MQCB, která definuje funkci odběratele.

Adresa MQMD je předávána jako null, pokud byl použit MQGMO verze 4 k požadavku, aby byl vrácen popisovač zprávy místo MQMD.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

GetMsgOpts
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 podporovanou verzí.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Jedná se o oblast obsahující data zprávy.

Pokud není k dispozici žádná zpráva pro toto volání nebo pokud zpráva neobsahuje žádná data zprávy, je adresa *Buffer* poslána jako nulová.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

Kontext
Typ: MQCBC-input/output

Tato struktura poskytuje kontextové informace pro funkce zpětného volání. Podrobnosti viz "MQCBC-Kontext zpětného volání" na stránce 280.

Poznámky k použití

1. Mějte na paměti, že pokud rutiny zpětného volání používají služby, které by mohly prodlevu nebo blokovat podproces, například příkaz MQGET s čekáním, může zpozdit odbavení jiných zpětných volání.
2. Samostatná jednotka práce není automaticky zřízena pro každé vyvolání rutiny zpětného volání, takže rutiny mohou vydávat volání s potvrzením nebo odložit potvrzení, dokud nebude zpracována logická dávka práce. Je-li dávka práce potvrzena, potvrzuje zprávy pro všechny funkce zpětného volání, které byly vyvolány od posledního bodu synchronizace.
3. Programy vyvolané CICS LINK nebo CICS START načítají parametry pomocí služeb CICS prostřednictvím pojmenovaných objektů známých jako kanálové kontejnery. Názvy kontejnerů jsou stejné jako názvy parametrů. Další informace naleznete v dokumentaci produktu CICS .
4. Rutiny zpětného volání mohou vydávat volání MQDISC, ale ne pro vlastní připojení. Pokud například rutina zpětného volání vytvořila připojení, může k odpojení připojení také připojení.
5. Rutina zpětného volání by neměla obecně spoléhat na vyvolání ze stejného podprocesu vždy po každém. Je-li to nutné, použijte při spuštění připojení MQCTLO_THREAD_AFFINITY.
6. Když rutina zpětného volání přijme nenulový kód příčiny, musí provést příslušnou akci.
7. Funkce MQCB_FUNKCE není v rámci adaptéru IMS podporována.

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.

Při 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 části "Atributy pro fronty" na stránce 827). Volby zavření jsou shrnuty v tomto tématu.

Trvalé odběry lze buď zachovat, nebo odebrat. Tyto odběry jsou vytvořeny pomocí volání MQSUB s volbou MQSO_TRVALÝ.

Při zavírání manipulátoru do spravovaného místa určení (tj. parametru **Hobj** vráceného voláním 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 nezůstane žádná.

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 649 .

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 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 byl příkaz *Hobj* vrácen při volání MQSUB, volání selže s kódem příčiny MQRC_OPTION_NOT_VALID_FOR_TYPE a objekt nebude odstraněn.

Tabulka 544. Volby zavření pro různé typy objektů

Typ objektu nebo fronty	MQCO_NONE	MQCO_DELETE	MQCO_DELETE_PURGE
Objekt jiný než fronta	Zachováno	Neplatné	Neplatné
Předdefinovaná fronta	Zachováno	Neplatné	Neplatné

Tabulka 544. Volby zavření pro různé typy objektů (pokračování)			
Typ objektu nebo fronty	**MQCO_NONE**	**MQCO_DELETE**	**MQCO_DELETE_PURGE**
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.

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 byla akce 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 545. 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 zavřením popisovače trvalého odběru a jeho zrušením, nebo zavřením popisovače netrvalého odběru, použijte následující volby uzavření odběru:

Tabulka 546. 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 vyžádání aplikací, a mohou být buď vyřazeny, nebo spotřebovány z fronty před dokončením operace MQCLOSE.

MQCO_IMMEDIATE
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.

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 již nejsou žádné další zprávy ve vyrovnávací paměti klienta pro dopředné čtení, protože mezi posledním voláním MQGET a následujícím voláním MQCLOSE může být doručena zpráva, která by byla zahozena, kdyby bylo použito MQCO_IMMEDIATE.

Je-li 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 649 .

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') Při 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*:
   - Pokud jsou 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 je fronta znovu otevřena s volbou MQOO_BROWSE, vytvoří se nový kurzor procházení (viz MQOO_BROWSE).
- Je-li pro tento manipulátor v době volání MQCLOSE aktuálně uzamknuta zpráva, bude zámek uvolněn (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* (buď 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 manipulátor *Hobj* zadaný ve volání MQCLOSE není ten, který byl 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 pracovní jednotky, které ovlivňují frontu, fronta a její zprávy jsou stále odstraněny, ale pracovní jednotky neselhávají. 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*:
   - 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.

     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 **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 signalizuje správci front, že aplikace dosáhla synchronizačního bodu, a že všechny zprávy a operace get, které se vyskytly od posledního bodu synchronizace, jsou trvalé.

Zprávy, které jsou vloženy jako součást pracovní jednotky, jsou zpřístupněny ostatním aplikacím; zprávy načtené jako součást pracovní jednotky jsou odstraněny.

- z/OS V systému z/OSse volání používá pouze pro dávkové programy (včetně dávkových DL/I programů IMS).

Syntaxe

MQCMIT (*Hconn*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Vypsané kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** .

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_BACKED_OUT
(2003, X'7D3') Unit of work backed out.

NEVYŘÍZENÉ MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace vázaného zpracování je nevyřízený.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CALL_INTERRUPTED
(2549, X'9F5') MQPUT nebo MQCMIT bylo přerušeno a zpracování opětovného připojení nemůže znovu vytvořit definitivní výsledek.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není platné v prostředí.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

MQRC_OUTCOME_MIXED
(2123, X'84B') Výsledek operace commit nebo back-out je smíšený.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

SELHÁNÍ OPERACE MQRC_RECONNECT_FAILED
(2548, X'9F4') Po opětovném připojení došlo k chybě při obnovení manipulátorů pro opětovné připojení připojení k tabulce.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Toto volání používejte pouze v případě, že správce front sám koordinuje pracovní jednotku. To může být:
   - Lokální jednotka práce, kde se změny týkají pouze IBM MQ prostředků.
   - Globální jednotka práce, kde mohou změny ovlivnit prostředky patřící jiným správcům prostředků a které ovlivňují prostředky produktu IBM MQ .

Další podrobnosti o lokálních a globálních jednotkách práce viz "MQBEGIN-Begin unit of work" na stránce 626.

2. V prostředích, ve kterých správce front nekoordinuje pracovní jednotku, je třeba namísto funkce MQCMIT použít příslušné volání potvrzení. Prostředí může také podporovat implicitní potvrzení způsobené normálně ukončováním aplikace.
   - V systému z/OSpoužijte následující volání:
     - Dávkové programy (včetně dávkových DL/I programů produktu IMS) mohou použít volání MQCMIT, pokud jednotka práce ovlivňuje pouze prostředky produktu IBM MQ . However, if the unit of work affects both IBM MQ resources and resources belonging to other resource managers (for example, Db2), use the SRRCMIT call provided by the z/OS Recoverable Resource Service (RRS). Volání SRRCMIT potvrzuje změny prostředků náležejících ke správcům prostředků, kteří byli povoleni pro koordinaci RRS.
     - Aplikace CICS musí použít příkaz `EXEC CICS SYNCPOINT` k výslovnému potvrzení jednotky práce. Eventuálně je ukončení transakce výsledkem implicitního potvrzení transakce. Volání MQCMIT nelze použít pro aplikace produktu CICS .
     - Aplikace produktu IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. `GU` a `CHKP` , k potvrzení jednotky práce. Volání MQCMIT nelze použít pro aplikace IMS (jiné než dávkové DL/I programy).
   - V systému IBM ipoužijte toto volání pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz `STRCMTCTL` s parametrem **`CMTSCOPE(*JOB)`** nesmí být vydán pro úlohu.
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v části Poznámky k použití MQDISC .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
   - Hodnoty polí *`GroupId`*, *`MsgSeqNumber`*, *`Offset`*a *`MsgFlags`* v MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

   Když je jednotka práce potvrzena, správce front zachová informace o skupině a segmentu a aplikace může pokračovat ve vkládání nebo získávání zpráv do aktuální skupiny zpráv nebo logické zprávy.

   Zachování informací o skupině a segmentech při potvrzení transakce umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů v rámci několika pracovních jednotek. Použití několika jednotek práce je výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv ve správném okamžiku, pokud dojde k selhání systému. Podrobnosti o restartování ve správném bodu po selhání systému najdete v tématu MQPMO_LOGICAL_ORDER a MQGMO_LOGICAL_ORDER.

   Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce:
5. Pracovní jednotka má stejný rozsah jako popisovač připojení; všechna volání IBM MQ , která ovlivňují konkrétní transakci, musí být provedena pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení naleznete v popisu parametru **Hconn** popsaného v MQCONN.
6. Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
7. Dlouhá-spuštěná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydá potvrzení nebo zpětné volání, může plnit fronty se zprávami, které nejsou k dispozici pro jiné aplikace. Chcete-li se proti tomu bránit, musí administrátor nastavit

atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily správné fungování očekávaných aplikací systému zpráv.

8. ALW Pokud je v systémech AIX, Linux, and Windows parametr **Reason** MQRC_CONNECTION_BROKEN (s *CompCode* MQCC_FAILED) nebo MQRC_UNEXPECTED_ERROR, je možné, že byla jednotka práce 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 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 547 na stránce 658); 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

Tabulka 547. Rozsah nesdílených manipulátorů na různých platformách (pokračování)	
Platforma	**Rozsah nesdíleného popisovače**
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 336 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-Připojit správce front (rozšířený)

Volání MQCONNX připojuje aplikační program ke správci front. Poskytuje manipulátor připojení ke správci front, který je používán aplikací při následných voláních IBM MQ .

Volání MQCONNX se podobá volání MQCONN, až na to, že MQCONNX umožňuje určit volby pro řízení způsobu, jakým volání funguje.

- Toto volání je podporováno ve všech systémech IBM MQ a v klientech IBM MQ připojených k těmto systémům.

Připojení klienta nelze provést na instalaci pouze serveru a lokální připojení nelze provést pouze u instalace klienta.

Syntaxe

MQCONNX (*QMgrName*, *ConnectOpts*, *Hconn*, *CompCode*, *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -Vstup

Podrobné informace naleznete v popisu parametru **QMgrName** popsaného v příručce “MQCONN-Připojení správce front” na stránce 656 .

ConnectOpts
Typ: MQCNO-input/output

Podrobnosti viz “MQCNO-Volby připojení” na stránce 315.

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 548 na stránce 664); 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 548. 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

Podrobné informace naleznete v popisu parametru **CompCode** popsaného v příručce “MQCONN-Připojení správce front” na stránce 656 .

Příčina
Typ: MQLONG-výstup

Volání MQCONN a MQCONNX mohou vracet následující kódy. Seznam dalších kódů, které mohou být vráceny voláním MQCONNX, najdete 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 336 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:

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_AIR_ERROR
(2385, X' 951 ') Záznam ověřovacích informací není platný.

CHYBA 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 nejsou platná.

CHYBA MQRC_AUTH_INFO_TYPE_ERROR
(2386, X' 952 ') Typ ověřovacích informací není platný.

CHYBA MQRC_CD_ERROR
(2277, X'8E5') Definice kanálu není platná.

CHYBA MQRC_CLIENT_CONN_ERROR
(2278, X'8E6') Pole připojení klienta nejsou platná.

CHYBA MQRC_CNO_ERROR
(2139, X'85B') Struktura volby Connect-options není platná.

MQRC_CONN_TAG_IN_USE
(2271, X'8DF') Značka připojení se používá.

MQRC_CONN_TAG_NOT_USABLE
(2350, X'92E') Značka připojení není použitelná.

CHYBA MQRC_LDAP_PASSWORD_ERROR
(2390, X' 956 ') Heslo LDAP není platné.

CHYBA 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.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_SCO_ERROR
Struktura konfigurace SSL (2380, X'94C') není platná struktura voleb konfigurace SSL.

CHYBA MQRC_SSL_CONFIG_ERROR
(2392, X' 958 ') Chyba konfigurace SSL.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

Pro programovací jazyk Visual Basic se používá následující bod:

- Parametr **ConnectOpts** je deklarován jako typ MQCNO. Je-li aplikace spuštěna jako IBM MQ MQI clienta chcete určit parametry kanálu připojení klienta, deklarujte parametr **ConnectOpts** jako typ Any, aby aplikace mohla určovat strukturu MQCNOCD při volání na místě 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 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-Řízení zpětných volání

Volání MQCTL provádí řízení akcí zpětných volání a manipulátorů objektů otevřených pro připojení.

Syntaxe

MQCTL (*Hconn*, *Operation*, *ControlOpts*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* můžete zadat následující speciální hodnotu:

MQC_DEF_HCONN
Výchozí popisovač připojení.

Operace
Typ: MQLONG-vstup

Operace se zpracovává na zpětné volání definované pro zadaný popisovač objektu. Musíte uvést jednu a jednu jedinou z následujících možností:

MQOP_START

Spustit přijímání zpráv pro všechny definované funkce odběratele zpráv pro uvedený popisovač připojení.

Zpětná volání se spouští na podprocesu spuštěnému systémem, který se liší od všech podprocesů aplikace.

Tato operace poskytuje řízení poskytovaného manipulátoru připojení k systému. Jediné volání MQI, které může být vydáno jiným vláknem, než je odběratelský podproces, je:

- MQCTL s operací MQOP_STOP
- MQCTL s operací MQOP_SUSPEND
- MQDISC-Provede operaci MQCTL s operací MQOP_STOP před odpojením modulu HConn.

Funkce MQRC_HCONN_ASYNC_ACTIVE je vrácena v případě, že je při spuštění manipulátoru připojení zadáno volání rozhraní API produktu IBM MQ a volání nepochází z funkce spotřebitele zpráv.

Pokud spotřebitel zpráv zastaví připojení během volání MQCBCT_START_CALL, pak se volání MQCTL vrátí s kódem příčiny selhání MQRC_CONNECTION_STOPPED.

To může být vydáno ve funkci odběratele. Pro stejné připojení jako rutina zpětného volání je jeho jediným účelem zrušení dříve vydané operace MQOP_STOP.

Tato volba není podporována v následujících prostředích: CICS v systému z/OS , nebo pokud je aplikace svázána s knihovnou IBM MQ bez podprocesů.

MQOP_START_WAIT
Spustit přijímání zpráv pro všechny definované funkce odběratele zpráv pro uvedený popisovač připojení.

Spotřebitelé zpráv se spouštějí na stejném podprocesu a řízení se nevrací volajícímu objektu MQCTL, dokud:

- Uvolněno v použití operací MQOP_STOP nebo MQOP_SUSPEND produktu MQCTL nebo
- Všechny rutiny odběratele byly deregistrovány nebo pozastaveny.

Pokud jsou všichni spotřebitelé odregistrováni nebo pozastaveni, je vydána implicitní operace MQOP_STOP.

Tuto volbu nelze použít v rámci rutiny zpětného volání, a to ani pro aktuální popisovač připojení, ani pro žádný jiný manipulátor připojení. Je-li volání vyzkoušeno, vrátí se s hodnotou MQRC_ENVIRONMENT_ERROR.

Pokud během operace MQOP_START_WAIT nejsou žádné registrované, nepozastavené spotřebitele, volání selže s kódem příčiny MQRC_NO_CALLBACKS_ACTIVE.

Je-li během operace MQOP_START-WAIT připojení pozastaveno, volání MQCTL vrátí kód příčiny varování MQRC_CONNECTION_SUSPENDED; připojení zůstane 'spuštěno'.

Aplikace se může rozhodnout pro zadání příkazu MQOP_STOP nebo MQOP_RESUME. V této instanci jsou bloky operací MQOP_RESUME.

Tato volba není podporována v jednom vláknovém klientovi.

MQOP_STOP
Zastavte příjem zpráv a počkejte, až všichni spotřebitelé dokončí své operace před dokončením této volby. Tato operace uvolní manipulátor připojení.

Je-li tato volba vydána v rámci rutiny zpětného volání, nebude tato volba účinná, dokud rutina nebude ukončena. Žádné další rutiny pro spotřebitele zpráv se nezavolají po dokončení zpracování rutin pro zprávy, které již byly přečteny, a po zastavení volání (je-li požadována) pro rutiny zpětného volání.

Je-li vydáno mimo rutinu zpětného volání, řízení se nevrátí k volajícímu, dokud nebudou dokončeny rutiny odběratele pro zprávy, které již byly načteny, a po ukončení volání (je-li požadována) na zpětné volání. Samotné zpětné volání však zůstává registrováno.

Tato funkce nemá žádný vliv na zprávy dopředného čtení. Musíte zajistit, aby spotřebitelé spouštěli MQCLOSE (MQCO_QUIESCE) z funkce zpětného volání, abyste určili, zda jsou k dispozici nějaké další zprávy, které mají být dodány.

MQOP_SUSPEND
Pozastavit příjem zpráv. Tato operace uvolní manipulátor připojení.

To nemá žádný vliv na čtení napřed zpráv pro aplikaci. Hodláte-li dlouhodobě zastavit spotřebovávání zpráv, zvažte uzavření fronty a opětovné otevření, až spotřeba pokračuje.

Je-li vydáno v rámci rutiny zpětného volání, neprojeví se, dokud rutina nebude ukončena. Po ukončení aktuální rutiny nebudou volány žádné další rutiny pro spotřebitele zpráv.

Je-li vydáno mimo zpětné volání, řízení se nevrátí k volajícímu, dokud nebude dokončena aktuální zákaznický rutina a nebudou zavolány žádné další.

MQOP_RESUME
Pokračujte ve spotřebování zpráv.

Tato volba je obvykle vydána z hlavního podprocesu aplikace, ale lze ji také použít v rámci rutiny zpětného volání ke zrušení dřívější žádosti o pozastavení vydané ve stejné rutině.

Je-li příkaz MQOP_RESUME použit k obnovení operace MQOP_START_WAIT, pak budou bloky operací.

ControlOpts
Typ: MQCTLO-vstup

Volby, které řídí akci MQCTL

Podrobnosti o struktuře viz MQCTLO .

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb pro převod dat.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

CHYBA_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_ REGISTERED
(2448, X' 990 ') Nelze zrušit registraci, pozastavení nebo obnovení, protože neexistuje žádné registrované zpětné volání

CHYBA MQRC_CALLBACK_ROUTINE_ERROR
(2486, X'9B6') Either, both CallbackFunction a CallbackName byly zadány v volání MQOP_REGISTER.

Nebo byly zadány buď CallbackFunction , nebo CallbackName , ale neodpovídají momentálně registrované funkci zpětného volání.

MQRC_CALLBACK_TYPE_ERROR
(2483, X'9B3') Nesprávné pole typu CallBackType.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

CHYBA MQRC_CBD_ERROR
(2444, X'98C') Blok volby je chybný.

CHYBA MQRC_CBD_OPTIONS_ERROR
(2484, X'9B4') Nesprávné pole voleb MQCBD.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CORRELA_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
(2012, X'7DC') Volání není platné v prostředí.

PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá informace o zablokování fronty.

KONFLIKT MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální jednotky konfliktu práce.

CHYBA MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb získání zprávy není platná.

FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Nekonzistentní specifikace procházení.

NEKONZISTENCE 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.

KONFLIKT MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s místní jednotkou práce.

CHYBA MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

CHYBA MQRC_MAX_MSG_LENGTH_ERROR
(2485, X'9B5') Nesprávná hodnota pole MaxMsgLength

CHYBA MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zprávy není platný.

MQRC_MODULE_ENTRY_NOT_FOUND
(2497, X'9C1') Uvedený vstupní bod funkce nebyl nalezen v modulu.

MQRC_MODULE_INVALID
(2496, X'9C0') Modul je nalezen, ale je nesprávného typu (32 bit/64 bitů) nebo není platnou knihovnou DLL.

MQRC_MODULE_NOT_FOUND
(2495, X'9BF') Modul nebyl nalezen v cestě pro vyhledávání, nebo neměl oprávnění k načtení.

CHYBA MQRC_MSG_ID_
(2206, X'89E') Chyba identifikátoru zprávy.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

CHYBA MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřená pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu byla od otevření změněna.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

CHYBA OPERACE MQRC_OPERATION_ERROR
(2488, X'9B8') Nesprávný kód operace na volání rozhraní API

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA OBJEKTU 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_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má špatný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815 ') Signál nevyřízený pro tento popisovač.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

CHYBA MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotek práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unit of work not available for the queue manager to use.

CHYBA MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Čekací interval v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Chybná verze dodávaného MQGMO.

VERZE MQRC_WRONG_MD_VERSION
(2257, X'8D1') Chybná verze dodaných MQMD.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Rutiny zpětného volání musí zkontrolovat odezvy ze všech služeb, které vyvolávají, a pokud rutina zjistí podmínku, kterou nelze vyřešit, musí vydat příkaz MQCB MQOP_DEREGISTER, který zabrání opakovaným voláním rutiny zpětného volání.
2. Pokud používáte asynchronní spotřebu v aplikaci, kde správce transakcí XA spravuje globální transakce, včetně aktualizací produktu IBM MQ, je třeba zvážit následující dodatečné body:
   a. Po volání produktu **xa_open**není platné volání MQCTL (MQOP_START) pro objekt **HConn**po jeho vytvoření.

      Důvodem je skutečnost, že se produkt **HConn** připojil ke kontextu XA, a proto nelze k němu přistupovat v rámci samostatného podprocesu nebo podprocesů, které používá asynchronní mechanismus spotřeby.
   b. Při volání funkce MQCTL (MQOP_START) v tomto scénáři dojde k selhání volání s kódem příčiny MQRC_ASYNC_XA_CONFLICT (2350).
   c. Po volání produktu **xa_open**je platné volání funkce MQCTL (MQOP_START_WAIT) pro objekt **HConn**po jeho vytvoření.

      Důvodem je to, že tato metoda spuštění asynchronního mechanismu spotřeby způsobí, že se všechna další zpětná volání pro spuštění produktu **HConn** spustí na podprocesu, ve kterém došlo k volání MQCTL. Odkaz mezi **HConn** a vláknem proto není ztracen.
3. z/OS Při operaci z/OS, je-li operace MQOP_START:
   - Programy, které používají asynchronní rutiny zpětného volání, musí mít oprávnění k použití produktu z/OS UNIX System Services (z/OS UNIX).
   - Programy jazyka LE (Language Environment), které používají asynchronní rutiny zpětného volání, musí používat běhovou volbu LE POSIX(ON).
   - Programy typu Non-LE, které používají asynchronní rutiny zpětného volání, nesmí používat rozhraní z/OS UNIX pthread_create (callable service BPX1PTC).
4. z/OS MQCTL není podporováno v rámci adaptéru IMS .

Poznámka: V produktu CICSnení operace MQOP_START podporována. 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-Odpojení správce front

Volání MQDISC přeruší spojení mezi správcem front a aplikačním programem a je inverzní k volání MQCONN nebo MQCONNX.

- V systému z/OSvšechny aplikace, které používají asynchronní spotřebu zpráv, zpracování událostí nebo zpětné volání, hlavní řídicí podproces musí před ukončením vydat volání MQDISC. Další podrobnosti najdete v tématu Asynchronní spotřeba zpráv produktu IBM MQ .
- V produktu z/OSnemusí aplikace produktu CICS vydat toto volání k odpojení od správce front.

  Pokud aplikace CICS toto volání provádí, nemá žádný účinek, pokud nebylo provedeno dřívější volání MQCONNX a uvádí jednu 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 manipulátory objektů zavřeny.

Syntaxe

MQDISC (*Hconn*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-input/output

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace produktu CICS můžete vynechat volání MQCONN a pro produkt *Hconn* určit následující hodnotu:

MQC_DEF_HCONN
Výchozí popisovač připojení.

Při úspěšném dokončení volání nastaví správce front *Hconn* na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

MQC_UNUSABLE_HCONN
Nepoužitelná obsluha připojení.

V systému z/OSje parametr *Hconn* nastaven na hodnotu, která není definována.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících kódů:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_BACKED_OUT
(2003, X'7D3') Unit of work backed out.

MQRC_CONN_TAG_NOT_RELEASED
(2344, X' 928 ') Značka připojení není uvolněná.

NEVYŘÍZENÉ MQRC_OUTCOME_PENDING
(2124, X'84C') Výsledek operace vázaného zpracování je nevyřízený.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_DIC_LOAD_ERROR
(2138, X'85A') Nelze načíst modul odpojení adaptéru.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

MQRC_API_EXIT_INIT_ERROR
Inicializace uživatelské procedury rozhraní API (2375, X' 947 ') API se nezdařila.

CHYBA MQRC_API_EXIT_TERM_ERROR
(2376, X' 948 ') Ukončení uživatelské procedury rozhraní API se nezdařilo.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

MQRC_OUTCOME_MIXED
(2123, X'84B') Výsledek operace commit nebo back-out je smíšený.

CHYBA OBJEKTU 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 není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Je-li volání MQDISC vydáno, má-li připojení stále otevřené objekty pod tímto připojením, správce front tyto objekty zavře a volby zavření nastavené na hodnotu MQCO_NONE.
2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, jak aplikace končí:

   a. Pokud aplikace vydá volání MQDISC před ukončením:

   - Pro koordinovanou pracovní jednotku správce front vydá správce front volání MQCMIT jménem aplikace. Jednotka práce je potvrzena, pokud je to možné, a vrácena, pokud ne.
   - Pro externě koordinovanou jednotku práce není žádná změna stavu pracovní jednotky; správce front však obvykle informuje o tom, že pracovní jednotka musí být potvrzena, když ji požádá koordinátor jednotky práce.

     V systémech z/OS, CICS, IMS (jiných než dávkových programů DL/1) a aplikací RRS jsou tyto aplikace podobné.

   b. Pokud aplikace skončí normálně, ale bez zadání volání MQDISC, závisí akce na daném prostředí:

   - V produktu z/OS, s výjimkou aplikací MQ Java nebo MQ JMS , se vyskytují akce popsané v poznámce 2a .
   - Ve všech ostatních případech se vyskytnou akce popsané v poznámce 2c .

Kvůli rozdílům mezi prostředími se ujistěte, že aplikace, které chcete použít k portu, buď potvrdí, nebo zazálohují jednotku práce, než skončí.

c. Pokud aplikace skončí *nestandardně* bez volání volání MQDISC, bude jednotka práce vrácena zpět.

3. V systému z/OSplatí následující body:

- Aplikace produktu CICS nemusí zadávat volání MQDISC k odpojení od správce front, protože se samotný systém CICS připojuje ke správci front a volání MQDISC nemá žádný vliv na toto připojení.
- CICS, IMS (jiné než dávkové programy DL/1) a aplikace RRS používají jednotky práce koordinovanou externím koordinátorem jednotky práce. Výsledkem je, že volání MQDISC nemá vliv na stav pracovní jednotky (pokud existuje), která existuje při vydání volání.

  Volání MQDISC však *znamená* konec použití značky připojení *ConnTag* , které bylo přidruženo k připojení dřívějším voláním MQCONNX vydaným aplikací. Pokud existuje aktivní pracovní jednotka, která odkazuje na značku připojení při volání MQDISC, volání bude dokončeno s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_CONN_TAG_NOT_RELEASED. Značka připojení nebude k dispozici pro opětovné použití, dokud nebude externí koordinátor jednotek práce vyřešen pracovní jednotku.

Poznámka: V produktu CICSnení operace MQOP_START podporována. 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 */
```

Vyvolání asembleru System/390

```
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 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*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

HOBJ
Typ: MQHOTBJ-vstup

Tento popisovač představuje frontu, ze které se má načíst 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 stránce 726):

- MQO_INPUT_SHARED
- MQO_INPUT_EXCLUSIVE
- MQO_INPUT_AS_Q_DEF
- MQOOK_BROWSE

MsgDesc
Typ: MQMD-I/O

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy. Podrobnosti viz “MQMD-Deskriptor zpráv” na stránce 419.

Je-li *BufferLength* menší než délka zprávy, *MsgDesc* je zaplněn správcem front, zda je MQGMO_ACCEPT_TRUNCATED_MSG zadán v parametru **GetMsgOpts** (viz MQGMO-Options field).

Pokud aplikace poskytuje version-1 MQMD, vrácená zpráva má před daty zprávy aplikace předponu MQMDE, ale pouze v případě, že jedno nebo více polí v prostředí MQMDE má nevýchozí

hodnotu. Pokud mají všechna pole v MQMDE výchozí hodnoty, MQMDE se vynechá. Název formátu MQFMT_MD_EXTENSION v poli *Formát* v MQMD označuje, že je přítomen objekt MQMDE.

Aplikace nemusí poskytovat strukturu MQMD, je-li v poli *MsgHandle* zadána platná obsluha zprávy. Není-li v tomto poli uvedeno nic, bude deskriptor zprávy odebrán z deskriptoru asociovaného s manipulátory zpráv.

Pokud aplikace poskytuje popisovač zprávy, nikoli strukturu MQMD, a určuje MQGMO_PROPERTIES_FORCE_MQRFH2, volání selže s kódem příčiny MQRC_MD_ERROR. Volání také selhává, 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.

Jsou-li zadány volby shody a je použit deskriptor zprávy přidružený k popisovači zprávy, vstupní pole použitá pro srovnávání se zobrazí od popisovače zprávy.

GetMsgOpts
Typ: MQGMO-input/output

Podrobnosti viz "MQGMO-Volby získání zprávy" na stránce 366.

BufferLength
Typ: MQLONG-vstup

Toto je délka v bajtech oblasti *Buffer* . Uveďte nulu pro zprávy, které nemají žádná data, nebo pokud má být zpráva odebrána z fronty a vyřazena data (musíte uvést MQGMO_ACCEPT_TRUNCATED_MSG v tomto případě).

Poznámka: Délka nejdelší zprávy, kterou je možné číst z fronty, je dána atributem fronty **MaxMsgLength** ; viz "Atributy pro fronty" na stránce 827.

Vyrovnávací paměť
Typ: MQBYTExBufferLength-output

Jedná se o 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 záhlaví záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Pokud je *BufferLength* menší než délka zprávy, do **Buffer**se přesune co největší část zprávy. K tomu dojde v případě, že je v parametru **GetMsgOpts** zadán parametr MQGMO_ACCEPT_TRUNCATED_MSG (další informace naleznete v části MQGMO-Options field).

Znaková sada a kódování dat v **Buffer** jsou dána poli *CodedCharSetId* a *Encoding* vrácenými v argumentu **MsgDesc** . Jsou-li tyto hodnoty odlišné od hodnot požadovaných příjemcem, příjemce musí data zprávy aplikace převést na znakovou sadu a požadované kódování. Je možné použít volbu MQGMO_CONVERT (v případě potřeby s uživatelem napsanou uživatelskou procedurou) k převodu dat zprávy; podrobnosti o této volbě naleznete v části "MQGMO-Volby získání zprávy" na stránce 366 .

Poznámka: Všechny ostatní parametry volání MQGET se nacházejí ve znakové sadě a kódování lokálního správce front (daný atributem správce front **CodedCharSetId** a MQENC_NATIVE).

Pokud se volání nezdaří, mohl by se obsah vyrovnávací paměti stále měnit.

V programovacím jazyku C je tento parametr deklarován jako ukazatel-to-void: adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BufferLength** nulový, *Buffer* není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assembler s hodnotou null.

DataLength
Typ: MQLONG-výstup

Toto je délka dat aplikace *ve zprávě*v bajtech. Je-li tato hodnota větší než *BufferLength*, vrátí se v parametru **Buffer** pouze *BufferLength* bytů (to znamená, že zpráva je zkrácena). Je-li hodnota nula, zpráva neobsahuje žádná data aplikace.

Je-li *BufferLength* menší než délka zprávy, *DataLength* je správce front stále dokončen, je-li v parametru **GetMsgOpts** zadán parametr MQGMO_ACCEPT_TRUNCATED_MSG (další informace naleznete v části MQGMO-Options field). To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k umístění dat zprávy a pak znovu vydat volání s vyrovnávací pamětí odpovídající velikosti.

Je-li však zadána volba MQGMO_CONVERT a převedená data zprávy jsou příliš dlouhá na to, aby se vešly do *Buffer*, hodnota vrácená pro *DataLength* je:

- Délka *nepřevedených* dat, pro formáty definované správcem front.

  V tomto případě, pokud by charakter dat způsobil rozšíření během konverze, musí aplikace alokovat vyrovnávací paměť větší než hodnotu vrácenou správcem front pro *DataLength*.
- Hodnota vrácená uživatelskou procedurou pro převod dat pro formáty definované aplikací.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Vypsané kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** . Pokud aplikace určuje volbu MQGMO_CONVERT a uživatelská procedura je vyvolána pro převod některých nebo všech dat zprávy, uživatelská procedura určí, jaká hodnota se vrátí pro parametr **Reason** . V důsledku toho jsou možné hodnoty jiné než zdokumentované hodnoty.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

MQRC_CONVERTED_STRING_TOO_BIG
(2190, X'88E') Konvertovaný řetězec je příliš velký pro pole.

CHYBA MQRC_DBCS_ERROR
(2150, X'866 ') DBCS řetězec není platný.

CHYBA MQRC_FORMAT_ERROR
(2110, X'83E') Formát zprávy není platný.

SKUPINA MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není úplná.

ZPRÁVA MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není úplná.

MQRC_INCONSISTENT_CCCSIDS
(2243, X'8C3') Segmenty zprávy mají odlišné CCSID.

KÓDOVÁNÍ MQRC_INCONSISTENT_ENCODINGS
(2244, X'8C4') Segmenty zprávy mají odlišné kódování.

NEKONZISTENCE MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

CHYBA MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Neplatné použití tokenu zprávy.

MQRC_NO_MSG_LOCKED
(2209, X'8A1') Žádná zpráva nebyla zamknuta.

MQRC_NOT_CONVERTED
(2119, X'847 ') Data zprávy nejsou převedena.

MQRC_OPTIONS_CHANGED
(nnnn, X'xxx ') Volby, které měly být konzistentní, byly změněny.

MQRC_PARTIALLY_CONVERTED
(2272, X'8E0') Data zprávy jsou částečně převedena.

MQRC_SIGNAL_REQUEST_ACCEPTED
(2070, X'816 ') Nebyla vrácena žádná zpráva (ale přijat požadavek na signál).

CHYBA MQRC_SOURCE_BUFFER_ERROR
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

CHYBA 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.

CHYBA MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842 ') Kódování čísel s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') Kódování celého čísla zdroje nebylo rozpoznáno.

CHYBA MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Parametr délky zdroje není platný.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862 ') Cílový parametr vyrovnávací paměti není platný.

CHYBA MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845 ') Packed-decimal encoding specified by receiver not recognized.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846 ') Kódování čísel s pohyblivou řádovou čárkou určené příjemcem není rozpoznáno.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') Cílové celé číslo kódování nebylo rozpoznáno.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Byla vrácena oříznutá zpráva (zpracování dokončeno).

OPERACE MQRC_TRUNCATED_MSG_FAILED
(2080, X'820 ') Byla vrácena zkrácená zpráva (zpracování není dokončeno).

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_CONV_LOAD_ERROR
(2133, X'855 ') Nelze načíst moduly služeb pro převod dat.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Unit of work backed out.

CHYBA MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Prostředek Coupling Facility není k dispozici.

MQRC_CF_STRU_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility selhala.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

MQRC_CF_STRU_LIST_HDR_IN_USE
(2347, X'92B') Hlavička prostředku Coupling-facility-záhlaví se používá.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CORRELA_ID_ERROR
(2207, X'89F') Chyba identifikátoru korelace.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Subsystém Db2 není k dispozici.

MQRC_GET_INHIBITED
(2016, X'7E0') Získá informace o zablokování fronty.

KONFLIKT MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální jednotky konfliktu práce.

CHYBA MQRC_GMO_ERROR
(2186, X'88A') Struktura voleb získání zprávy není platná.

FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

MQRC_INCONSISTENT_BROWSE
(2259, X'8D3') Nekonzistentní specifikace procházení.

NEKONZISTENCE 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.

KONFLIKT MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s místní jednotkou práce.

CHYBA MQRC_MATCH_OPTIONS_ERROR
(2247, X'8C7') Volby shody nejsou platné.

CHYBA MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zprávy není platný.

CHYBA MQRC_MSG_ID_
(2206, X'89E') Chyba identifikátoru zprávy.

MQRC_MSG_SEQ_NUMBER_ERROR
(2250, X'8CA') Pořadové číslo zprávy není platné.

CHYBA MQRC_MSG_TOKEN_ERROR
(2331, X'91B') Použití tokenu zprávy není platné.

MQRC_NO_MSG_AVAILABLE
(2033, X'7F1') Nejsou k dispozici žádné zprávy.

MQRC_NO_MSG_UNDER_CURSOR
(2034, X'7F2') Procházení kurzoru není umístěno na zprávě.

MQRC_NOT_OPEN_FOR_BROWSE
(2036, X'7F4') Fronta není otevřená pro procházení.

MQRC_NOT_OPEN_FOR_INPUT
(2037, X'7F5') Fronta není otevřena pro vstup.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu byla od otevření změněna.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA OBJEKTU 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_INDEX_TYPE_ERROR
(2394, X'95A') Fronta má špatný typ indexu.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_SECOND_MARK_NOT_ALLOWED
(2062, X'80E') Zpráva je již označena.

MQRC_SIGNAL_OUTSTANDING
(2069, X'815 ') Signál nevyřízený pro tento popisovač.

MQRC_SIGNAL1_ERROR
(2099, X'833 ') Signální pole není platné.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizačního bodu není k dispozici.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

CHYBA MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotek práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unit of work not available for the queue manager to use.

CHYBA MQRC_WAIT_INTERVAL_ERROR
(2090, X'82A') Čekací interval v MQGMO není platný.

MQRC_WRONG_GMO_VERSION
(2256, X'8D0') Chybná verze dodávaného MQGMO.

VERZE MQRC_WRONG_MD_VERSION
(2257, X'8D1') Chybná verze dodaných MQMD.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Načtená zpráva je obvykle vymazána z fronty. Toto odstranění se může vyskytnout jako součást samotného volání MQGET nebo jako součást synchronizačního bodu.

   Volby procházení jsou: MQGMO_BROTS_FIRST, MQGMO_BROE_NEXT a MQGMOROWS_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, je odemknuta dříve zamčená zpráva. V tomto případě není načtena žádná zpráva a parametry **MsgDesc**, **BufferLength**, **Buffer**a **DataLength** se nekontrolují ani nemění.

3. U aplikací, které vydávají volání MQGET, může být zpráva načtena, pokud dojde k nestandardnímu ukončení nebo přerušení spojení při zpracování volání. K tomuto problému dochází proto, že náhradní místo spuštěné na stejné platformě jako správce front, které vydává volání MQGET v zastoupení aplikace, nemůže zjistit ztrátu aplikace, dokud se nevrátila zpráva k aplikaci, poté, co byla zpráva odebrána z fronty. K tomuto problému může dojít jak pro trvalé zprávy, tak pro přechodné zprávy.

   Chcete-li vyloučit riziko ztráty zpráv tímto způsobem, vždy načítat zprávy v rámci jednotek práce. To znamená zadáním volby MQGMO_SYNTCPOINT na volání MQGET a pomocí volání MQCMIT nebo MQBACK k potvrzení nebo vrácení pracovní jednotky při dokončení zpracování zpráv. Je-li zadán parametr MQGMO_SYNCPOINT a klient byl ukončen nestandardním způsobem nebo došlo k přerušení spojení, vrátí náhradní jednotka práci na správci front a zpráva je znovu zařazena do fronty. Další informace o bodech synchronizace naleznete v tématu Aspekty synchronizačních bodů v aplikacích produktu IBM MQ.

   Tato situace může nastat s klienty produktu IBM MQ a s aplikacemi, které jsou spuštěny na stejné platformě jako správce front.

4. Pokud aplikace vkládá posloupnost zpráv do konkrétníhofronta v rámci jedné jednotky práce a poté potvrdí, že jednotka práce byla úspěšně dokončena, zprávy jsou k dispozici pro načtení následujícím způsobem:
   - Je-li fronta *nesdílená fronta* (tedy lokální fronta), budou všechny zprávy v rámci jednotky práce k dispozici ve stejnou dobu.
   - Je-li fronta *sdílená fronta*, budou zprávy v rámci jednotky práce dostupné v pořadí, ve kterém byly vloženy, ale ne všechny najednou. Když je systém výrazně zatížen, je možné, aby první zpráva v jednotce práce byla úspěšně načtena, ale pro volání MQGET pro druhou nebo následující zprávu v jednotce práce, která má selhat s parametrem MQRC_NO_MSG_AVAILABLE. Pokud k tomuto problému dojde, aplikace musí čekat krátkou dobu a poté se pokusit o provedení operace znovu.
5. Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny určité podmínky. Podrobnosti naleznete v tématu Poznámky k použití MQPUT . Jsou-li podmínky splněny, jsou zprávy předloženy přijímající aplikaci v pořadí, v jakém byly odeslány, pokud:
   - Z fronty získává zprávy pouze jeden příjemce.

     Pokud existují dvě nebo více aplikací, které dostávají zprávy z fronty, musí souhlasit s odesilatelem mechanizmu, který má být použit k identifikaci zpráv, které patří do posloupnosti. Odesilatel může například nastavit všechna pole `CorrelId` ve zprávách v posloupnosti na hodnotu, která byla jedinečná pro danou posloupnost zpráv.
   - Příjemce neprovede záměrné změny pořadí načítání, například zadáním konkrétního `MsgId` nebo `CorrelId`.

   Pokud odesílající aplikace vloží zprávy jako skupinu zpráv, jsou zprávy předkládány přijímající aplikaci ve správném pořadí, pokud přijímající aplikace určuje volbu MQGMO_LOGICAL_ORDER na volání MQGET. Další informace o skupinách zpráv viz:
   - pole MQMD- MsgFlags
   - MQPMO_LOGICAL_ORDER
   - MQGMO_LOGICAL_ORDER

   Pokud uživatel získává zprávy ve skupině pod bodem synchronizace, musí před pokusem o dokončení transakce zajistit, aby byla zpracována úplná skupina.
6. Aplikace musí testovat pro kód zpětné vazby MQFB_QUIT v poli `Feedback` parametru **`MsgDesc`** , a pokud naleznou tuto hodnotu, musí končit. Další informace viz Pole MQMD-Feedback .
7. Pokud byla fronta označená `Hobj` otevřena s volbou MQOO_SAVE_ALL_CONTEXT a kód dokončení z volání MQGET je MQCC_OK nebo MQCC_WARNING, kontext přidružený k manipulátoru fronty `Hobj` je nastaven na kontext zprávy, která byla načtena (pokud není nastavena volba MQGMO_BALed FIRST, MQGMO_BROE_NEXT nebo MQGMOROWS_MSG_UNDER_CURSOR), v takovém případě je kontext označen jako nedostupný).

   Uložený kontext můžete použít na následné volání MQPUT nebo MQPUT1 uvedením voleb MQPMO_PASS_IDENTITY_CONTEXT nebo MQPMO_PASS_ALL_CONTEXT. To umožňuje přenést kontext přijaté zprávy jako celek nebo jeho část do jiné zprávy (například, když je zpráva předána do jiné fronty). Další informace o kontextu zprávy viz téma Kontext zprávy.
8. Pokud zahrnete volbu MQGMO_CONVERT do parametru **`GetMsgOpts`** , data zprávy aplikace jsou převedena na reprezentaci požadovanou přijímající aplikací před tím, než jsou data umístěna do parametru **`Buffer`** :
   - Pole `Format` v informacích o ovládacím prvku ve zprávě identifikuje strukturu dat aplikace a pole `CodedCharSetId` a `Encoding` v řídicích informacích ve zprávě uvádí identifikátor a kódování znakové sady.
   - Aplikace, která volá volání MQGET, uvádí v polích `CodedCharSetId` a `Encoding` v parametru **`MsgDesc`** identifikátor a kódování znakové sady, do kterého se mají převést data zprávy aplikace.

   Je-li konverze dat zprávy nezbytná, provede převod buď samotným správcem front, nebo uživatelem zapsaným výstupem, v závislosti na hodnotě pole `Format` v informacích o ovládacím prvku ve zprávě:

- Následující formáty názvů 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
  - HLAVIČKA MQFMT_DEAD_LETTER_HEADER
  - ZÁHLAVÍ MQFMT_DICT_HEADER
  - MQFMT_EVENT verze 1
  - MQFMT_EVENT verze 2 (pouze z/OS)
  - MQFMT_IMS
  - MQFMT_IMS_VAR_STRING
  - ROZŠÍŘENÍ MQFMT_MD_EXTENSION
  - MQFMT_PCF
  - MQFMT_REF_MSG_HEADER
  - ZÁHLAVÍ MQFMT_RF_HEADER
  - MQFMT_RF_HEADER_2
  - ŘETĚZEC MQFMT_STRING
  - SPOUŠTĚČ MQFMT_TRIGGER
  - MQFMT_WORK_INFO_HEADER (pouze z/OS)
  - ZÁHLAVÍ 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 při načítání zprávy z fronty nepokusí o převod.

  Poznámka: Je-li parametr MQGMO_CONVERT zadán v rámci volání MQGET pro zprávu s názvem formátu MQFMT_NONE a znaková sada nebo kódování zprávy se liší od hodnoty zadané argumentem **MsgDesc** , vrátí se zpráva v parametru **Buffer** (za předpokladu, že neobsahuje žádné další chyby), ale volání bude dokončeno s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

  MQFMT_NONE můžete použít buď, když charakter dat zprávy znamená, že nevyžaduje převod, nebo když se odesílající a přijímající aplikace mezi sebou dohodly na formuláři, ve kterém mají být odeslána data zprávy.
- Všechny ostatní názvy formátů předají zprávu uživateli, který je zapsán pro převod. Ukončení má stejný název jako formát, kromě dodatků specifických pro prostředí. Jména formátů zadaných uživatelem nesmí začínat písmeny IBM MQ.

  Podrobné informace o ukončení konverze dat naleznete v části “Uživatelská procedura konverze dat” na stránce 897 .

Uživatelská data ve zprávě lze převést mezi libovolnými podporovanými znakovými sadami a kódováními. Uvědomte si však, že pokud zpráva obsahuje jednu nebo více struktur záhlaví IBM MQ , zprávu nelze převést ze znakové sady, která má znaky s dvoubajtovou nebo vícebajtovou nebo vícebajtovou znakovou sadou pro některý ze znaků platných v názvech front. Kód příčiny MQRC_SOURCE_CCSID_ERROR nebo MQRC_TARGET_CCSID_ERROR má za následek pokus o provedení tohoto pokusu a zpráva byla vrácena nekonverzovanou. Příkladem takové znakové sady je znaková sada Unicode UTF-16 .

Při návratu z MQGET označuje následující kód příčiny, že zpráva byla úspěšně převedena:

- MQRC_NONE

Následující kód příčiny informuje o tom, 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 indikují, že zpráva nebyla převedena.

Poznámka: Interpretace tohoto kódu příčiny je pravdivá pro převody provedené uživatelem napsaným výstupem pouze v případě, že uživatelská procedura odpovídá pokynům pro zpracování popsaným v příručce “Uživatelská procedura konverze dat” na stránce 897.

9. Když používáte objektově orientované rozhraní k získání zpráv, můžete se rozhodnout nezadat vyrovnávací paměť, která má obsahovat data zprávy pro volání MQGET. Ve verzích produktu IBM MQstarších než IBM WebSphere MQ 7.0však bylo možné, aby příkaz MQGET selhal s kódem příčiny MQRC_CONVERTERED_MSG_TO_BIG, a to ani v případě, že vyrovnávací paměť nebyla zadána. Od IBM WebSphere MQ 7.0, když obdržíte zprávu pomocí objektově orientované aplikace bez omezení velikosti vyrovnávací paměti pro příjem zpráv, aplikace selže při MQRC_CONVERTED_MSG_TOO_BIG a přijme převedenou zprávu. To platí pro následující prostředí:

   - .NET, včetně plně spravovaných aplikací
   - JAZYK C++
   - Java (IBM MQ classes for Java)

   Poznámka: Pro všechny klienty platí, že pokud je hodnota proměnné `sharingConversations` nula, kanál bude fungovat tak, jak se stalo před produktem IBM WebSphere MQ 7.0, a zpracování zpráv se vrátí k chování produktu IBM WebSphere MQ 6 . V této situaci, pokud je vyrovnávací paměť příliš malá pro přijetí převedené zprávy, je vrácena nekonvertovaná zpráva s kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG. Další informace o produktu `sharingConversations`naleznete v tématu Použití sdílení konverzací v aplikaci klienta.

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 použít výchozí znakovou sadu určenou pro instalaci, která se blíží ke skutečné znakové sadě při převodu řetězcových dat. Výsledkem je, že volání MQGET může být úspěšné s kódem dokončení MQCC_OK, namísto dokončení 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 to, 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 platí jak pro data zprávy aplikace, tak pro znaková pole v strukturách MQMD a MQMDE:

    - Výchozí konverze dat zprávy aplikace se vyskytne pouze, když jsou všechny následující příkazy pravdivé:
      - Aplikace určuje MQGMO_CONVERT.
      - Zpráva obsahuje data, která musí být převedena buď ze znakové sady nebo do znakové sady, která není podporována.
      - Výchozí převod byl povolen, když byl správce front nainstalován nebo restartován.
    - 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 volba MQGMO_CONVERT není určena aplikací na volání MQGET.

11. V případě programovacího jazyka Visual Basic platí tyto body:

    - Je-li velikost parametru **`Buffer`** menší než délka zadaná 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, není typu `String`, použijte příkazVolání MQGETAny na místo MQGET.

      Volání MQGETAny má stejné parametry jako volání MQGET, kromě toho, že parametr **`Buffer`** je deklarován jako typ `Any`, což umožňuje načíst jakýkoli typ dat. To však znamená, že produkt `Buffer` nelze zkontrolovat, aby se zajistilo, že velikost bude mít velikost alespoň `BufferLength` bajtů.

12. Ne všechny volby MQGET jsou podporovány, je-li povoleno dopředné čtení. V následující tabulce je uvedeno, které volby jsou povoleny a zda je lze změnit mezi voláními MQGET.

Tabulka 549. Volby MQGET povolené, je-li povoleno čtení napřed

	Povoleno, je-li dopředné čtení povoleno a lze je měnit mezi voláními MQGET	**Povoleno, je-li dopředné čtení povoleno, ale nelze je měnit mezi voláními MQGET [a].**	**Volby MQGET, které nejsou povoleny, je-li povoleno čtení napřed, [b]**
Hodnoty MQGET MD	MsgId [c] CorrelId [c]	Kódování CodedCharSetId	
Volby MQGMO MQGET	MQGMO_WAIT MQGMO_NO_WAIT FUNKCE MQGMO_FAIL_IF_QUIESCING MQGMOPRE_FIRST [d] MQGMO_BE_NEXT [d] ZPRÁVA MQGMO_BROWSE_MESSAGE _UNDER_CURSOR [d]	MQGMO_SYNCPOINT_IF_PERSISTENT MQGMO_NO_SYNCPOINT SOUBOR MQGMO_ACCEPT_TRUNCATED_MSG MQGMO_CONVERT MQGMO_LOGICAL_ORDER ZPRÁVA MQGMO_COMPLETE_MSG MQGMO_ALL_MSGS_AVAILABLE DOSTUPNÉ MQGMO_ALL_SEGMENTS_AVAILABLE POPISOVAČ MQGMO_MARK_ BROWSE_HANDLE MQGMO_MARKER_BROWSE_CO_OP MQGMO_UNMARK_ BROWSE_CO_OP POPISOVAČ MQGMO_UNMARK_ BROWSE_HANDLE MQGMO_UNMARKED_BROWSE_MSG, MQGMO_PROPERTIES_FORCE_MQRFH2 MQGMO_NO_PROPERTIES MQGMO_PROPERTIES_IN_HANDLE MQGMO_PROPERTIES_COMPATIBILITY	SIGNÁL MQGMO_SET_DATA MQGMO_SYNCPOINT PŘESKOČENO MQGMO_MARK_ _VYDÁNÍ MQGMO_MSG_UNDER _CURSOR [d] MQGMOVÝ_ZÁMEK MQGMO_ODEMKNOUT
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 získat nepotvrzené zprávy pouze v případě, že tyto zprávy jsou vloženy do stejné lokální jednotky práce jako získání. Aplikace nemohou přijímat nepotvrzené zprávy nedestruktivně.
14. Zprávy pod kurzorem procházení lze načíst v jednotce práce. V tomto ohledu není možné 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 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-Dotaz na atributy objektu

Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

Platné jsou tyto typy objektů:

- Správce front
- Fronta
- Seznam názvů
- Definice procesu

Syntaxe

MQINQ (*Hconn*, *Hobj*, *SelectorCount*, *Selektory*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN -vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX .

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je zadána následující hodnota:

MQHC_DEF_HCONN
Výchozí popisovač připojení.

HOBJ
Typ: MQHOBJ -vstup

Tento manipulátor představuje objekt (typu libovolného typu) s požadovanými atributy. Popisovač musí být vrácen předchozím voláním MQOPEN , které určuje 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

Jedná se o pole selektorů atributů produktu **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 se volání nezdaří s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_SELECTOR_ERROR.

Ve zvláštním případě front:

- Není-li selektor platný pro fronty libovolného typu, volání selže s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_SELECTOR_ERROR.
- Pokud se selektor vztahuje pouze na fronty typů jiných typů, než je typ objektu, volání se zdaří s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_SELECTOR_NOT_FOR_TYPE.
- Je-li dotazovaná fronta fronta klastru, selektory, které jsou platné, závisí na tom, jak byla fronta vyřešena; viz "Poznámky k použití" na stránce 713 , kde jsou další podrobnosti.

Selektory můžete určit v libovolném pořadí. Hodnoty atributu odpovídající celočíselným selektorům atributů (selektory MQIA_*) jsou vraceny v produktu *IntAttrs* ve stejném pořadí, ve kterém se tyto selektory vyskytují v produktu *Selectors*. Hodnoty atributu, které odpovídají selektorům znakových atributů (selektory MQCA_*), jsou vraceny v produktu *CharAttrs* ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory MQIA_* mohou být prokládané se selektory MQCA_* ; je důležité pouze relativní pořadí v rámci každého typu.

Poznámka:

1. Selektory atributů celého čísla a znaku 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 definují konstanty MQIA_LAST_USED a MQCA_LAST_USED nejvyšší hodnotu, kterou správce front přijme.
2. Pokud se všechny selektory MQIA_* vyskytnou jako první, lze použít stejná čísla prvků k adresování příslušných prvků v polích *Selectors* a *IntAttrs* .
3. Pokud je argument **SelectorCount** nastaven na nulu, *Selectors* se na něj neodkazuje. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo S/390 assembler s hodnotou null.

Atributy, které mohou být dotazovány, jsou vypsány v následujících tabulkách. Pro selektory produktu MQCA_* je konstanta, která definuje délku výsledného řetězce v řetězci *CharAttrs* , uvedena v závorkách.

Tabulky, které následují za seznamem selektorů, podle objektů, v abecedním pořadí, takto:

- Selektory atributů produktu Tabulka 550 na stránce 700 MQINQ pro fronty
- Selektory atributů Tabulka 551 na stránce 703 MQINQ pro seznamy názvů
- Selektory atributů produktu Tabulka 552 na stránce 703 MQINQ pro definice procesu
- Selektory atributů produktu Tabulka 553 na stránce 704 MQINQ pro správce front

Všechny selektory jsou podporovány na všech platformách produktu IBM MQ kromě těch, které jsou uvedeny ve sloupci **Poznámka** takto:

NEz/OS
Podporováno na všech platformách **s výjimkou** z/OS

z/OS
Podporováno **pouze** v systému z/OS

Tabulka 550. Selektory atributů MQINQ 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	

Tabulka 550. Selektory atributů MQINQ pro fronty (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQCA_BACKOUT_REQ_Q_NAME	MQ_Q_NAME_LENGTH	Nadměrný název fronty vrácených zpráv	
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	
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 známo 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 účtovacích dat 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	

Tabulka 550. Selektory atributů MQINQ pro fronty (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	Výchozí volba open-for-input	
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 zjistit počet vrácení	
MQIA_INDEX_TYPE	MQLONG	Typ indexu udržovaný pro frontu	z/OS
MQIA_INHIBIT_GET	MQLONG	Zda jsou povoleny operace get.	
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 zpráv důležitá	
MQIA_NPM_CLASS	MQLONG	Úroveň spolehlivosti pro přechodné zprávy	
MQIA_OPEN_INPUT_COUNT	MQLONG	Počet volání příkazu MQOPEN , která mají otevřenou frontu pro vstup	
MQIA_OPEN_OUTPUT_COUNT	MQLONG	Počet volání příkazu MQOPEN , která mají otevřenou frontu pro výstup	
MQIA_PROPERTY_CONTROL	MQLONG	Atribut řízení vlastnosti	
MQIA_Q_DEPTH_HIGH_EVENT	MQLONG	Řídicí atribut pro vysoké události hloubky fronty	NEz/OS
MQIA_Q_DEPTH_HIGH_LIMIT	MQLONG	Horní mez hloubky fronty	NEz/OS
MQIA_Q_DEPTH_LOW_EVENT	MQLONG	Řídicí atribut pro nízké události hloubky fronty	NEz/OS
MQIA_Q_DEPTH_LOW_LIMIT	MQLONG	Dolní mez hloubky fronty	NEz/OS
MQIA_Q_DEPTH_MAX_EVENT	MQLONG	Řídicí atribut pro maximální události hloubky 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

Tabulka 550. Selektory atributů MQINQ pro fronty (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_RETENTION_INTERVAL	MQLONG	Interval uchování fronty	
MQIA_SCOPE	MQLONG	Obor 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 551. Selektory atributů MQINQ pro seznamy názvů

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum většino-nedávných změn	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas nejposlednějších změn	
MQCA_NAMELIST_DESC	MQ_NAMELIST_DESC_LENGTH	Popis seznamu názvů	
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*	Názvy v seznamu názvů	
MQIA_NAME_COUNT	MQLONG	Počet názvů v seznamu názvů	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS

Tabulka 552. Selektory atributů MQINQ pro definice procesu

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum většino-nedávných změn	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas nejposlednějších změn	
MQCA_APPL_ID	MQ_PROCESS_APPL_ID_LENGTH	Identifikátor aplikace	

Tabulka 552. Selektory atributů MQINQ pro definice procesu (pokračování)

Selektor	Délka pole	Popis	Poznámka
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 553. Selektory atributů produktu MQINQ pro správce front

Selektor	Délka pole	Popis	Poznámka
MQCA_ALTERATION_DATE	MQ_DATE_LENGTH	Datum většino-nedávných změn	
MQCA_ALTERATION_TIME	MQ_TIME_LENGTH	Čas nejposlednějších změn	
MQCA_CHANNEL_AUTO_DEF_EXIT	MQ_EXIT_NAME_LENGTH	Název uživatelské procedury automatické definice kanálu	
MQCA_CHINIT_SERVICE_PARM		Rezervováno pro použití produktem IBM	
MQCA_CLUSTER_WORKLOAD_DATA	MQ_EXIT_DATA_LENGTH	Data předávaná 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 příkazu systému	
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 za účelem spojení. Název se použije při použití služeb správy dynamické domény správce pracovní zátěže.	z/OS
MQCA_IGQ_USER_ID	MQ_USER_ID_LENGTH	Identifikátor uživatele fronty v rámci skupiny	z/OS

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQCA_INSTALLATION_DESC	MQ_INSTALLATION_DESC_LENGTH	Popis přidružené instalace	Ne z/OS. NEIBM i
MQCA_INSTALLATION_NAME	MQ_INSTALLATION_NAME_LENGTH	Název instalace přidružené ke správci front	Ne z/OS. NEIBM i
MQCA_INSTALLATION_PATH	MQ_INSTALLATION_PATH_LENGTH	Cesta, kde je instalován přidružený produkt IBM MQ	Ne z/OS. NEIBM i
MQCA_LU_GROUP_NAME	MQ_LU_NAME_LENGTH	Generický název jednotky LU pro modul listener LU 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front, jež má být použita	z/OS
MQCA_LU_NAME	MQ_LU_NAME_LENGTH	Název jednotky 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 člena SYS1.PARMLIB APPCPM *xx*, který jmenuje LUADD pro tento inicializátor kanálu.	z/OS
MQCA_PARENT	MQ_Q_MGR_NAME_LENGTH	Název hierarchicky připojeného správce front, který je nominován jako nadřízený prvek 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	
MQCA_TCP_NAME	MQ_TCP_NAME_LENGTH	Název systému TCP/IP, který používáte	z/OS

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_ACCOUNTING_CONN_OVERRIDE	MQLONG	Přepsat nastavení evidence	NEz/OS
MQIA_ACCOUNTING_INTERVAL	MQLONG	Jak často zapisovat intermediační evidenční záznamy	NEz/OS
MQIA_ACCOUNTING_MQI	MQLONG	Řídí shromažďování účtovacích informací pro data MQI	NEz/OS
MQIA_ACCOUNTING_Q	MQLONG	Řídí shromažďování účtovacích informací pro fronty	NEz/OS
MQIA_ACTIVE_CHANNELS	MQLONG	Maximální počet kanálů, které mohou být aktivní kdykoli	z/OS
MQIA_ADOPTNEWMCA_CHECK	MQLONG	Prvky, které jsou zkontrolovány a určují, zda má být adoptována agent MCA. Kontrola se provede, pokud je zjištěn nový příchozí kanál, který má stejný název jako agent MCA, který je již aktivní.	z/OS
MQIA_ADOPTNEWMCA_INTERVAL	MQLONG	Doba v sekundách, po kterou bude nový kanál čekat na ukončení osiřelého kanálu	NEz/OS
MQIA_ADOPTNEWMCA_TYPE	MQLONG	Určuje, zda restartovat osiřelou instanci agenta MCA určitého typu kanálu automaticky, když je zjištěn nový příchozí požadavek na kanál odpovídající parametrům AdoptNewMCACheck	z/OS
MQIA_AUTHORITY_EVENT	MQLONG	Řídicí atribut 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	Řídicí atribut pro události kanálu	
MQIA_CHINIT_ADAPTERS	MQLONG	Počet podúloh adaptéru, které mají být použity pro zpracování volání IBM MQ	z/OS
MQIA_CHINIT_DISPATCHERS	MQLONG	Počet dispečerů, který má být použit pro inicializátor kanálu	z/OS
MQIA_CHINIT_TRACE_AUTO_START	MQLONG	Zda se má spustit trasování inicializátoru kanálu automaticky	z/OS
MQIA_CHINIT_TRACE_TABLE_SIZE	MQLONG	Velikost datového prostoru trasování (v MB) inicializátoru kanálu	z/OS

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
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	Řídicí atribut 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
MQIA_DNS_WLM	MQLONG	Údaj o tom, zda modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registruje správce pracovní zátěže pro služby dynamického názvu domény	z/OS
MQIA_EXPIRY_INTERVAL	MQLONG	Interval mezi skenováními pro vypršelé	z/OS
MQIA_GROUP_UR	MQLONG	Řídicí atribut určuje, zda jsou pro tohoto správce front povoleny skupiny zotavení GROUP. Dispozice jednotky zotavení GROUP je k dispozici pouze v případě, že je správce front členem skupiny sdílení front.	z/OS
MQIA_IGQ_PUT_AUTHORITY	MQLONG	Řazení do front 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 IBM MQ pokusy o restartování modulu listener v případě selhání APPC nebo TCP/IP.	z/OS
MQIA_LOCAL_EVENT	MQLONG	Řídicí atribut pro lokální události	NEz/OS
MQIA_LOGGER_EVENT	MQLONG	Řídicí atribut pro blokování událostí	NEz/OS

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
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 jehož uplynutí může správce front automaticky odebrat značku z procházení zpráv. **Upozornění:** Tuto hodnotu byste neměli nastavit 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	
MQIA_MAX_PRIORITY	MQLONG	Maximální priorita	
MQIA_MAX_UNCOMMITTED_MSGS	MQLONG	Maximální počet nepotvrzených zpráv v rámci jednotky práce	
MQIA_OUTBOUND_PORT_MAX	MQLONG	S MQIA_OUTBOUND_PORT_MINdefinuje rozsah čísel portů, které se mají použít při vázání odchozích kanálů	z/OS
MQIA_OUTBOUND_PORT_MIN	MQLONG	S MQIA_OUTBOUND_PORT_MAXdefinuje rozsah čísel portů, které se mají použít při vázání odchozích kanálů	z/OS
MQIA_PERFORMANCE_EVENT	MQLONG	Řídicí atribut pro události výkonu	NEz/OS
MQIA_PLATFORM	MQLONG	Platforma, na které je správce front umístěn	
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Označuje, zda jsou funkce zabezpečení produktu Advanced Message Security dostupné pro správce front.	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	Počet pokusů o opětovné zpracování nezdařené zprávy příkazu pod bodem synchronizace	

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQIA_PUBSUB_MODE	MQLONG	Určuje, zda je spuštěn stroj publikování/odběru a rozhraní publikování/odběru ve frontě. Aplikace pro publikování nebo přihlášení k odběru pomocí rozhraní API vyžadují stroj publikování/odběru. Fronty, které jsou monitorovány rozhraním publikování/odběru ve frontě, vyžadují, aby bylo spuštěno rozhraní publikování/odběru ve frontě.	
MQIA_PUBSUB_NP_MSG	MQLONG	Zda se má vyřadit (nebo uchovat) nedoručenou vstupní zprávu	
MQIA_PUBSUB_NP_RESP	MQLONG	Ovládá chování nedoručených odpovědí zpráv.	
MQIA_PUBSUB_SYNC_PT	MQLONG	Zda se v synchronizačním bodu zpracovávají 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 k jakýmkoli strukturám prostředku CF s parametrem CFCONLOS nastaveným na hodnotu ASQMGR .	z/OS
MQIA_RECEIVE_TIMEOUT	MQLONG	Přibližně, jak dlouho kanál TCP/IP čeká na příjem dat, včetně synchronizačních signálů od svého partnera, než se vrátí do neaktivního stavu. Hodnota je numerická, kvalifikovaná MQIA_RECEIVE_TIMEOUT_TYPE.	z/OS
MQIA_RECEIVE_TIMEOUT_MIN	MQLONG	Minimální doba, po kterou kanál protokolu TCP/IP čeká na příjem dat, včetně synchronizačních 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 synchronizačních 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	Řídicí atribut pro vzdálené události	NEz/OS
MQIA_SECURITY_CASE	MQLONG	Případ profilů zabezpečení	z/OS

Tabulka 553. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_SSL_EVENT	MQLONG	Řídicí atribut pro události kanálu	
MQIA_SSL_FIPS_REQUIRED	MQLONG	Použít pro šifrování pouze certifikované algoritmy FIPS	
MQIA_SSL_RESET_COUNT	MQLONG	Počet resetování klíčů TLS	
MQIA_START_STOP_EVENT	MQLONG	Řídicí atribut pro události zahájení zastavení	NEz/OS
MQIA_STATISTICS_AUTO_CLUSSDR	MQLONG	Ovládá shromažďování statistických monitorovacích informací pro odesílací kanály klastru	
MQIA_STATISTICS_CHANNEL	MQLONG	Ovládá shromažďování statistických dat pro kanály	
MQIA_STATISTICS_INTERVAL	MQLONG	Jak často zapisovat data monitorování statistiky	NEz/OS
MQIA_STATISTICS_MQI	MQLONG	Ovládá shromažďování informací o monitorování statistiky pro správce front	NEz/OS
MQIA_STATISTICS_Q	MQLONG	Řídí shromažďování statistických dat pro fronty	NEz/OS
MQIA_SYNCPOINT	MQLONG	dostupnost bodu synchronizace	
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 se má použít funkce TCP KEEPALIVE ke kontrole, zda je druhý konec připojení stále dostupný	z/OS
MQIA_TCP_STACK_TYPE	MQLONG	Zda může inicializátor kanálu použít pouze adresní prostor TCP/IP zadaný v TCPNAME, nebo se může volitelně připojit k jakékoli vybrané adrese TCP/IP	z/OS
MQIA_TRACE_ROUTE_RECORDING	MQLONG	Ovládá záznam informací o přenosové cestě trasování	z/OS
MQIA_TREE_LIFE_TIME	MQLONG	Životnost nepoužitých neadministrativních témat	
MQIA_TRIGGER_INTERVAL	MQLONG	Interval spouštěče	

IntAttrCount

Typ: MQLONG -vstup

Toto je počet prvků v poli *IntAttrs* . Nula je platná hodnota.

Je-li IntAttrCount alespoň počtem selektorů MQIA_* v parametru **Selectors** , jsou vráceny všechny požadované celočíselné atributy.

IntAttrs
Typ: MQLONG x *IntAttrCount* -výstup

Toto je pole celočíselných hodnot atributů *IntAttrCount* .

Hodnoty celočíselných atributů se vrací ve stejném pořadí jako selektory MQIA_* v parametru **Selectors** . Pokud pole obsahuje více prvků než počet selektorů MQIA_* , přebytečné prvky se nezmění.

Pokud *Hobj* představuje frontu, ale selektor atributu se nevztahuje na tento typ fronty, je vrácena specifická hodnota MQIAV_NOT_APPLICABLE . Je vrácen pro odpovídající prvek v poli *IntAttrs* .

Pokud je argument **IntAttrCount** nebo **SelectorCount** nula, *IntAttrs* se na ně neodkazuje. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo S/390 assembler s hodnotou null.

Délka CharAttr
Typ: MQLONG -vstup

Toto je délka v bajtech parametru **CharAttrs** .

Hodnota CharAttrLength 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 zřetězeny znakové atributy, zřetězené. Délka vyrovnávací paměti je dána parametrem **CharAttrLength** .

Atributy znaků se vrací ve stejném pořadí jako selektory MQCA_* v parametru **Selectors** . Délka každého řetězce atributu je pevná pro každý atribut (viz Selektory) a hodnota v ní je zprava vyplněna mezerami, je-li to nutné. Můžete vytvořit vyrovnávací paměť větší, než je třeba, aby obsahovala všechny požadované atributy znaků a vyplňující znaky. Bajty přesahující poslední vrácenou hodnotu atributu jsou nezměněny.

Pokud *Hobj* představuje frontu, ale selektor atributu se nevztahuje na tento typ fronty, je vrácen znakový řetězec skládající se pouze z hvězdiček (*). Hvězdička je vrácena jako hodnota tohoto atributu v produktu *CharAttrs*.

Pokud je argument *CharAttrLength* nebo **SelectorCount** nula, *CharAttrs* se na ně neodkazuje. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo S/390 assembler s hodnotou null.

CompCode
Typ: MQLONG -výstup

Kód dokončení:

MQCC_OK
Úspěšné dokončení.

MQCC_WARNING
Varování (částečné dokončení).

MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Pokud je *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000') Chybí důvod k ohlášení.

Pokud je *CompCode* MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
(2008, X'7D8') Nedostatek prostoru povolený pro atributy znaků.

MQRC_INT_ATTR_COUNT_TOO_SMALL
(2022, X'7E6') Není dovolena dostatek prostoru pro celočíselné atributy.

MQRC_SELECTOR_NOT_FOR_TYPE
(2068, X'814') Selektor není použitelný 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') Ukončení API se nezdařilo.

MQRC_API_EXIT_LOAD_ERROR
(2183, X'887') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_ASID_MISMATCH
(2157, X'86D') Primární a domovská ID ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_STRUC_FAILED
(2373, X'945') Struktura prostředku Coupling Facility selhala.

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 atributů znaků není platný.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

MQRC_CONNECTION_BROKEN
(2009, X'7D9') Spojení se správcem front bylo ztraceno.

MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_STOPPING
(2203, X'89B') Probíhá ukončování činnosti 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') Pole celočíselné atributy není platné.

MQRC_NOT_OPEN_FOR_INQUIRE
(2038, X'7F6') Fronta není otevřena pro dotaz.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu se od otevření změnila.

MQRC_OBJECT_DAMAGED
(2101, X'835') Objekt je 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') Jméno správce front není platné nebo je neznámé.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_STOPPING
(2162, X'872') Správce front se vypíná.

MQRC_RESOURCE_PROBLEM
(2102, X'836') Není k dispozici dostatek systémových prostředků.

MQRC_SELECTOR_COUNT_ERROR
(2065, X'811') Počet selektorů není platný.

MQRC_SELECTOR_ERROR
(2067, X'813') Selektor atributu není platný.

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812') Počet selektorů je příliš velký.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_UNEXPECTED_ERROR
(2195, X'893') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin .

Poznámky k použití

1. Vrácené hodnoty jsou snímky vybraných atributů. Neexistuje žádná záruka, že atributy zůstanou stejné, než bude aplikace moci reagovat na vrácené hodnoty.
2. Otevřete-li modelovou frontu, vytvoří se dynamická lokální fronta. Dynamická lokální fronta se vytvoří, i když otevřete modelovou frontu a dotázat se na její atributy.

   Atributy dynamické fronty jsou ve velké míře stejné jako atributy modelové fronty v době, kdy je vytvořena dynamická fronta. Pokud pak použijete volání MQINQ v této frontě, správce front vrátí atributy dynamické fronty, nikoli atributy modelové fronty. Podrobnosti o tom, které atributy modelové fronty jsou zděděny dynamickou frontou, viz Tabulka 562 na stránce 830 .
3. Pokud je dotazovaný objekt alias frontu, jsou hodnoty atributů vrácené voláním MQINQ atributy fronty alias. Nejsou to atributy základní fronty nebo tématu, na které se rozlišuje alias.
4. Pokud je dotazovaný objekt fronta klastru, atributy, které mohou být dotazovány, závisí na tom, jak je fronta otevřena:
   - Můžete otevřít frontu klastru pro dotaz plus jeden nebo více operací vstupu, procházení nebo nastavení. Chcete-li tak učinit, musí existovat lokální instance fronty klastru, aby byla otevřená úspěšná. V tomto případě jsou atributy, které lze provádět dotazy, atributy, které jsou platné pro lokální fronty.

     Je-li fronta klastru otevřena pro dotaz bez zadání vstupu, procházení nebo nastavení, volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068), pokud se pokoušíte dotázat atributy, které jsou platné pouze pro lokální fronty, a nikoli fronty klastru.
   - Frontu klastru můžete otevřít pro dotazování při předávání názvu správce základní fronty připojeného správce front.

Chcete-li tak učinit, musí existovat lokální instance fronty klastru, aby byla otevřená úspěšná. Není-li základní správce front předán, volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068), pokud se pokoušíte dotázat se na atributy, které jsou platné pouze pro lokální fronty, a nikoli fronty klastru.

- Je-li fronta klastru otevřena pouze pro zjišťování, nebo dotaz a výstup, je možné se dotazovat pouze na uvedené atributy. Atribut **QType** má hodnotu MQQT_CLUSTER v tomto případě:
  - 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 jej otevřít s argumentem MQOO_BIND_NOT_FIXED zadaným ve výzvě MQOPEN . Případně zadejte MQOO_BIND_AS_Q_DEFa nastavte atribut **DefBind** fronty na MQBND_BIND_NOT_FIXED. Pokud otevřete frontu klastru bez pevné vazby, může po sobě následujících volání MQINQ pro frontu zjistit různé instance fronty klastru. Avšak, je to typické pro všechny instance mají stejné hodnoty atributu.
- Objekt alias fronty může být definován pro klastr. Protože TARGTYPE a TARGET nejsou klastrové atributy, proces, který provádí proces MQOPEN ve frontě aliasů, si není vědom objektu, na který je alias interpretováno.

  Během počátečního příkazu MQOPENse fronta aliasů interpretuje jako správce front a fronta v klastru. Rozpoznání názvu probíhá znovu na vzdáleném správci front a je zde, že je interpretována hodnota TARGTPYE fronty aliasů.

  Pokud se alias fronty interpretuje jako alias tématu, dojde k publikování zpráv vložených do fronty aliasů v tomto vzdáleném správci front.

Viz Fronty klastru

5. Možná budete chtít zjistit více atributů a pak nastavit některé z nich pomocí volání MQSET . Chcete-li program dotázat a nastavit efektivně, umístěte atributy, které mají být nastaveny na začátku polí selektoru. Pokud tak učiníte, lze pro MQSETpoužít stejná pole se sníženými počty.
6. Pokud se objeví více než jedna z varovných situací (viz parametr **CompCode**), vrácený kód příčiny je první v následujícím seznamu, který se používá:
   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 827
   - "Atributy pro seznamy názvů" na stránce 860
   - "Atributy pro definice procesu" na stránce 862
   - "Atributy správce front" na stránce 791

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 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-Dotaz na vlastnost zprávy

Volání MQINQMP vrací hodnotu vlastnosti zprávy.

Syntaxe

MQINQMP (*Hconn*, *Hmsg*, *InqPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *DataLength*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být na podprocesu zjišťující platné připojení pro vlastnost popisovače zprávy navázáno platné připojení, jinak volání selže s MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Toto je popisovač zprávy, který má být dotazován. Hodnota byla vrácena předchozím voláním příkazu **MQCRTMH** .

InqPropOpts
Typ: MQIMPO-input/output

Podrobnosti naleznete v datovém typu MQIMPO .

Název
Typ: MQCHARV-input/output

Název vlastnosti, která se má dotázat.

Pokud nelze nalézt žádnou vlastnost s tímto názvem, volání selže s příčinou MQRC_PROPERTY_NOT_AVAILABLE.

Na konci názvu vlastnosti můžete použít znak procento znaků zástupného znaku (%). Zástupný znak odpovídá žádnému znaku nebo více znakům, včetně znaku tečky (.). To umožňuje aplikaci dotazovat se na hodnotu mnoha vlastností. Volejte funkci 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 selže s hodnotou MQRC_PROPERTY_NOT_AVAILABLE. Pokud je pole *ReturnedName* ve struktuře Opts InqPropinicializováno s adresou nebo offsetem pro vrácený název vlastnosti, je tento proces dokončen při návratu z MQINQMP s názvem vlastnosti, která se shoduje. Je-li pole *VSBufSize* prvku *ReturnedName* ve struktuře InqPropOpts menší než délka vráceného názvu vlastnosti, kód dokončení je nastaven jako MQCC_FAILED s příčinou MQRC_PROPERTONE_TOO_BIG.

Vlastnosti, které mají známá synonyma, se vrátí takto:

1. Vlastnosti s předponou "mqps." jsou vráceny jako název vlastnosti produktu IBM MQ . Například "MQTopicString" je spíše vrácený název než "mqps.Top"
2. Vlastnosti s předponou "jms." nebo "mcd." se vrací jako název pole záhlaví JMS , například "JMSExpiration" je spíše vrácený název než "jms.Exp".
3. Vlastnosti s předponou "usr." jsou vráceny bez této předpony, např. "Color" je vrácen spíše než "usr.Color".

Vlastnosti se synonymy jsou vráceny pouze jednou.

V programovacím jazyku C jsou definovány následující makro proměnné pro dotazy na všechny vlastnosti a všechny vlastnosti, které začínají řetězcem "usr.":

MQPROP_INQUIRE_ALL
Dotaz na všechny vlastnosti zprávy.

MQPROP_INQUIRE_ALL lze použít následujícím způsobem:

```
MQCHARV Name = {MQPROP_INQUIRE_ALL};
```

MQPROP_INQUIRE_ALL_USR
Zjišťovat všechny vlastnosti zprávy, které spouští "usr.". Vrácený název je vrácen bez parametru "usr." .

Je-li zadán parametr MQIMP_INQ_NEXT, ale název se od předchozího volání změnil, nebo je to první volání, předpokládá se MQIMPO_INQ_FIRST.

Další informace o použití názvů vlastností naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

PropDesc
Typ: MQPD-výstup

Tato struktura se používá k definování atributů vlastnosti, včetně toho, co se stane, pokud tato vlastnost není podporována, jaký kontext zprávy vlastnost patří a do jakých zpráv má být vlastnost zkopírována. Podrobnosti o této struktuře viz MQPD .

Typ
Typ: MQLONG-input/output

Při návratu z volání MQINQMP je tento parametr nastaven na datový typ *Hodnota*. Datový typ může být libovolný z následujících:

LOGICKÁ HODNOTA MQTYPE_BOOLEAN
Booleovský.

ŘETĚZEC MQTYPE_BYTE_STRING
bajtový řetězec.

MQTYPE_INT8
8bitové podepsané celé číslo.

MQTYPE_INT16
16bitové podepsané celé číslo.

MQTYPE_INT32
32bitové celé číslo se znaménkem.

MQTYPE_INT64
64bitové podepsané celé číslo.

MQTYPE_FLOAT32
32-bitové číslo s pohyblivou řádovou čárkou.

MQTYPE_FLOAT64
64-bitové číslo s pohyblivou řádovou čárkou.

ŘETĚZEC MQTYPE_STRING
Znakový řetězec.

MQTYPE_NULL
Vlastnost existuje, ale má hodnotu null.

Není-li datový typ hodnoty vlastnosti rozpoznán, je vrácen parametr MQTYPE_STRING a do oblasti *Hodnota* bude vložena řetězcová reprezentace hodnoty. Řetězcovou reprezentaci datového typu lze nalézt v poli *TypeString* v parametru *InqPropOpts* . Kód dokončení varování je vrácen s příčinou MQRC_PROTO_TYPE_NOT_SUPPORTED.

Navíc, je-li zadána volba MQIMPO_CONVERT_TYPE, je požadována konverze hodnoty vlastnosti. Použijte *Typ* jako vstup pro uvedení datového typu, který má vlastnost vracet jako. Podrobné informace o převodu datového typu naleznete v popisu volby MQIMPO_CONVERT_TYPE struktury MQIMPO .

Pokud nevyžadujete převod typu, můžete na vstupu použít následující hodnotu:

MQTYPE_AS_SET
Hodnota vlastnosti je vrácena bez převodu jeho datového typu.

ValueLength
Typ: MQLONG-vstup

Délka v bajtech oblasti Hodnota. Uveďte nulu pro vlastnosti, pro které není požadována vrácená hodnota. Mohou to být vlastnosti, které jsou navrženy aplikací, aby měly hodnotu null nebo prázdný řetězec. Také uveďte nulu, pokud byla zadána volba MQIMPO_QUERY_LENGTH ; v tomto případě se nevrátí žádná hodnota.

Hodnota
Typ: MQBYTEx *ValueLength* -výstup

Toto je oblast, která má obsahovat dotazovanou hodnotu vlastnosti. Vyrovnávací paměť by měla být zarovnána na hranici vhodnou pro vracenou hodnotu. Pokud tak neučiníte, může to vést k chybě při pozdějším přístupu k této hodnotě.

Je-li hodnota *ValueLength* menší než délka hodnoty vlastnosti, hodnota vlastnosti je přesunuta do *Value* 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 tento parametr deklarován jako ukazatel-to-void; adresa libovolného typu dat může být zadána jako parametr.

Je-li parametr *ValueLength* nula, hodnota *Value* se neoznačuje a její hodnota předávaná programy napsanými v C nebo System/390 assembler může mít hodnotu null.

DataLength
Typ: MQLONG-výstup

Jedná se o délku skutečné hodnoty vlastnosti v bajtech, jak je vráceno v oblasti *Hodnota* .

Je-li hodnota *DataLength* menší než délka hodnoty vlastnosti, *DataLength* je stále zaplněna při návratu z volání MQINQMP. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k umístění hodnoty vlastnosti, a pak znovu zadejte volání s vyrovnávací pamětí příslušné velikosti.

Mohou být vráceny také následující hodnoty.

Je-li parametr *Type* nastaven na typ MQTYPE_STRING nebo MQTYPE_BYTE_STRING, postupujte takto:

MQVL_EMPTY_STRING
Vlastnost existuje, ale neobsahuje žádné znaky ani bajty.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_PROP_NAME_NOT_CONVERTED
(2492, X'09BC') Vrácený název vlastnosti není 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.

CHYBA MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nebylo možné analyzovat.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

NESROVNALOST MQRC_ASID_
(2157, X'086D') Primární a domovské ASID se liší.

CHYBA MQRC_BUFFER_ERROR
(2004, X'07D4') Hodnota parametru hodnoty není platná.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Hodnota parametru délky hodnoty není platná.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Parametr délky dat není platný.

CHYBA MQRC_IMPO_ERROR
(2464, X'09A0') Dotaz na strukturu voleb vlastností zprávy není platný.

CHYBA MQRC_HMSG_ERROR
(2460, X'099C') Popisovač zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Popisovač zprávy je již používán.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'07F8') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_PD_ERROR
(2482, X'09B2') Struktura deskriptoru vlastností není platná.

MQRC_PROP_CONV_NOT_SUPPORTED
(2470, X'09A6') Převod ze skutečného na požadovaný datový typ není podporován.

CHYBA MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

MQRC_PROPERTY_NAME_TOO_BIG
(2465, X'09A1') Název vlastnosti je příliš velký pro vracenou vyrovnávací paměť názvu.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7) Vlastnost není k dispozici.

HODNOTA MQRC_PROPERTY_VALUE_TOO_BIG
(2469, X'09A5') Hodnota vlastnosti je příliš velká pro oblast Hodnota.

CHYBA MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Chyba formátu čísla zjištěna v datech hodnoty.

CHYBA MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Neplatný požadovaný typ vlastnosti.

CHYBA MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'0871 ') Není k dispozici dostatek paměti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'0893 ') Vyskytla se neočekávaná chyba.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

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 popisovač zprávy do vyrovnávací paměti a je inverzní k volání MQBUFMH.

Syntaxe

MQMHBUF (*Hconn*, *Hmsg*, *MsgHBufOpts*, *Name*, *MsgDesc*, *BufferLength*, *Buffer*, *DataLength*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být ustanoveno platné připojení na podprocesu, který odstraňuje popisovač zprávy. Není-li ustanoveno platné připojení, volání selže při volání MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy, pro který je vyžadována vyrovnávací paměť. Hodnota byla vrácena předchozím voláním MQCRTMH.

MsgHBufOpts
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 474.

Název
Typ: MQCHARV-vstup

Název vlastnosti nebo vlastností, které mají být vloženy do vyrovnávací paměti.

Není-li nalezena žádná vlastnost odpovídající názvu, volání selže s MQRC_PROPERTY_NOT_AVAILABLE.

Můžete použít zástupný znak pro vložení více než jedné vlastnosti do vyrovnávací paměti. Chcete-li tak učinit, použijte zástupný znak '%' na konci názvu vlastnosti. Tento zástupný znak odpovídá nule nebo více znakům, včetně znaku '.' Znak.

V programovacím jazyku C jsou definovány následující makro proměnné pro dotazy na všechny vlastnosti a všechny vlastnosti, které začínají řetězcem '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 naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

MsgDesc
Typ: MQMD-I/O

Struktura *MsgDesc* popisuje obsah oblasti vyrovnávací paměti.

Na výstupu jsou pole *Encoding*, *CodedCharSetId* a *Format* nastavena tak, aby správně popisovala kódování, identifikátor znakové sady a formát dat v oblasti vyrovnávací paměti tak, jak je zapsaly volání.

Data v této struktuře se nacházejí ve znakové sadě a kódování aplikace.

BufferLength
Typ: MQLONG-vstup

BufferLength je délka oblasti vyrovnávací paměti, v bajtech.

Vyrovnávací paměť
Typ: MQBYTExBufferLength-output

Buffer definuje oblast, která má obsahovat vlastnosti zprávy. Musíte zarovnat vyrovnávací paměť na 4bajtové hranici.

Pokud je *BufferLength* menší než délka požadovaná pro uložení vlastností v *Buffer*, MQMHBUF selže s MQRC_PROPERTY_VALUE_TOO_BIG.

Obsah vyrovnávací paměti se může měnit 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 se neshodují s hodnotou uvedenou v *Name* a volání selže s kódem příčiny MQRC_PROPERTY_NOT_AVAILABLE.

Je-li *BufferLength* menší než délka požadovaná pro uložení vlastností ve vyrovnávací paměti, volání MQMHUF selže s MQRC_PROPERTY_VALUE_TOO_BIG, ale hodnota je stále zadána do *DataLength*. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné pro přizpůsobení vlastností a pak znovu zadejte volání s požadovanou *BufferLength*.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

CHYBA MQRC_MHBO_ERROR
(2501, X'095C') Popisovač zprávy pro strukturu vyrovnávací paměti není platný.

CHYBA MQRC_BUFFER_ERROR
(2004, X'07D4') Parametr vyrovnávací paměti není platný.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'07DA') Parametr délky dat není platný.

CHYBA MQRC_HMSG_ERROR
(2460, X'099C') Popisovač zprávy není platný.

CHYBA MQRC_MD_ERROR
(2026, X'07EA') Deskriptor zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Popisovač zprávy je již používán.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Název vlastnosti je neplatný.

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Vlastnost není k dispozici.

HODNOTA MQRC_PROPERTY_VALUE_TOO_BIG
(2469, X'09A5') hodnota BufferLength je příliš malá, aby mohla obsahovat zadané vlastnosti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

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 popisovač zprávy do vyrovnávací paměti.

Můžete ji použít s uživatelskou procedurou rozhraní API MQGET k přístupu k určitým vlastnostem, pomocí rozhraní API vlastností zpráv, a poté je předat zpět do vyrovnávací paměti aplikace určené k použití záhlaví MQRFH2 namísto obslužných rutin zpráv.

Toto volání je inverzní k volání MQBUFMH, které lze použít k analýze vlastností zpráv z vyrovnávací paměti do manipulátorů zpráv.

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-Otevřít objekt

Volání MQOPEN vytváří přístup k objektu.

Platné jsou tyto typy objektů:

- Fronta (včetně distribučních seznamů)
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Syntaxe

MQOPEN (*Hconn*, *ObjDesc*, *Options*, *Hobj*, *CompCode*, *Reason*)

Parametry

Hconn

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota Hconn byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

ObjDesc

Typ: MQOD-input/output

Jedná se o strukturu, která identifikuje objekt, který má být otevřen; podrobnosti viz "MQOD-Popisovač objektu" na stránce 476 .

Je-li pole ObjectName v parametru **ObjDesc** název modelové fronty, je dynamická lokální fronta je vytvořen s atributy modelové fronty; to se stává bez ohledu na to, které volby zadáte v parametru **Options** . Následné operace používající příkaz Hobj vrácené voláním MQOPEN jsou prováděny v nové dynamické frontě a nikoli ve frontě modelu. To platí i pro volání MQINQ a MQSET. Název modelové fronty v parametru **ObjDesc** se nahradí názvem vytvořené dynamické fronty. Typ dynamické fronty je určen hodnotou atributu **DefinitionType** v modelové frontě (viz "Atributy pro fronty" na stránce 827). Informace o možnostech zavření použitelných pro dynamické fronty naleznete v popisu volání MQCLOSE.

Volby

Typ: MQLONG-vstup

Je třeba určit alespoň jednu z následujících voleb:

- MQOOK_BROWSE

- MQOO_INPUT_ * (pouze jeden z nich)
- MQO_ DOTÁZAT SE
- MQOOK_VÝSTUP
- MQOOK_SADA
- MQOO_BIND_ * (pouze jeden z nich)

Podrobné informace o těchto volbách naleznete v následující tabulce. Další možnosti lze zadat podle potřeby. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné. Povoleny jsou pouze volby, které jsou použitelné pro typ objektu určeného parametrem `ObjDesc` .

Tabulka 554. 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
MQOOK_VÝSTUP	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_BROWSE	Ano	Ano	Ne	Ne	Ne	Ne
MQOO_CO_OP	Ano	Ano	Ne	Ne	Ne	Ne
MQO_ DOTÁZAT SE	Ano	Ano	[2]	Ano	Ne	Ne
MQOOK_SADA	Ano	Ano	[2]	Ne	Ne	Ne
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
Funkce MQOO_ALTERNATE_USER_AUTHORITY	Ano	Ano	Ano	Ano	Ano	Ano
Funkce MQOO_FAIL_IF_QUIESCING	Ano	Ano	Ano	Ano	Ano	Ano
MQOO_RESOLE_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, na kterou je určen alias.
2. Tato volba je platná pouze pro lokální definici vzdálené fronty.

3. Tato volba může být uvedena pro jakýkoli typ fronty, ale je ignorována, pokud fronta není fronta klastru. Atribut fronty **DefBind** však přepíše základní frontu i v případě, že fronta aliasů se nenachází v klastru.
4. Tyto atributy lze použít spolu s tématem, ale ovlivní pouze kontext nastavený pro zachovanou zprávu, nikoli pole kontextu odesílaná na libovolného odběratele.

Volby přístupu: Následující volby řídí typ operací, které lze na objektu provést:

MQO_INPUT_AS_Q_DEF
Chcete-li získat zprávy pomocí výchozího nastavení fronty, otevřete frontu.

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 827 .

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

MQO_INPUT_SHARED
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání se může zdařit, pokud je fronta aktuálně otevřena touto nebo jinou aplikací s MQOO_INPUT_SHARED, ale selže s kódem příčiny MQRC_OBJECT_IN_USE, je-li fronta aktuálně otevřena s MQOO_INPUT_EXCLUSIVE.

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

MQO_INPUT_EXCLUSIVE
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání se nezdaří s kódem příčiny MQRC_OBJECT_IN_USE, je-li fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (MQOO_INPUT_SHARED nebo MQOO_INPUT_EXCLUSIVE).

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

MQOOK_VÝSTUP

Otevřete frontu pro vložení zpráv, nebo téma nebo 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 následné volání MQPUT selžou při nastavení atributu na tuto hodnotu).

Tato volba je platná pro všechny typy front, včetně distribučních seznamů a témat.

Pro tyto volby platí následující poznámky:

- Může být uvedena pouze jedna 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 následující volání MQGET selžou, zatímco je atribut nastaven na tuto hodnotu).
- Je-li fronta definovaná jako nesdílitelná (tedy atribut fronty **Shareability** má hodnotu MQQA_NOT_SHAREABLE), pokusí se otevřít frontu pro sdílený přístup jako pokusy o otevření fronty s výlučným přístupem.
- Je-li alias fronta otevřena s jednou z těchto voleb, test pro výhradní použití (nebo pro to, zda má výlučnému použití jiná aplikace) je proti základní frontě, na kterou je alias interpretováno.
- Tyto volby nejsou platné, pokud **ObjectQMgrName** je název alias správce front; to je pravda i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je název lokálního správce front.

MQOOK_BROWSE
Chcete-li procházet zprávy, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET s jednou z následujících voleb:

- NEJPRVE MQGMO_BROWSE_FIRST
- PŘÍŠTĚ 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 založí kurzor procházení a umístí jej logicky před první zprávou ve frontě; další informace naleznete v tématu Pole MQGMO-Volby-volby .

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty. Je také neplatný, je-li `ObjectQMgrName` název alias správce front; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

MQOO_CO_OP
Otevřeno jako spolupracující člen sady obslužných rutin.

Tato volba je platná pouze s volbou MQOO_BROWSE. Je-li zadán bez MQOOL_BROWSE, funkce MQOPEN se vrátí s parametrem MQRC_OPTIONS_ERROR.

Vrácený popisovač je považován za člena spolupracující sady obslužných rutin pro následné volání MQGET s jednou z následujících voleb:

- MQGMO_MARKER_BROWSE_CO_OP
- MQGMO_UNMARKED_BROWSE_MSG,
- MQGMO_UNMARK_ BROWSE_CO_OP

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

MQO_ DOTÁZAT SE
Otevřít objekt k dotazu na atributy.

Fronta, seznam názvů, definice procesu nebo správce front je otevřen pro použití s dalšími voláními MQINQ.

Tato volba je platná pro všechny typy objektů jiných než distribuční seznamy. Není platná, pokud `ObjectQMgrName` je název alias správce front; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

MQOOK_SADA
Otevřete frontu pro nastavení atributů.

Fronta je otevřena pro použití s následnými voláními MQSET.

Tato volba je platná pro všechny typy front jiných než distribučních seznamů. Není platný, je-li `ObjectQMgrName` název lokální definice vzdálené fronty. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

Volby vázání: Při otevírání objektu z fronty klastru se používají následující volby: tyto volby řídí vázání manipulátoru fronty k instanci fronty klastru:

MQO_BIND_ON_OPEN
Lokální správce front sváže manipulátor fronty s instancí cílové fronty při otevření fronty.
V důsledku toho jsou všechny zprávy používající tento popisovač odeslány do stejné instance cílové fronty a stejnou přenosovou cestou.

Tato volba je platná pouze pro fronty a má vliv pouze na fronty klastru. Je-li tato volba zadána pro frontu, která není frontou klastru, je tato volba ignorována.

MQOO_BIND_NOT_FIXED
Tím se zastaví lokální správce front s vazbou manipulátoru fronty na instanci cílové fronty. Výsledkem je, že po sobě jdoucí volání MQPUT používající tento popisovač odesílá zprávy do různých instancí cílové fronty nebo do stejné instance, ale na různé přenosové cesty. Umožňuje také, aby byla instance vybrána později lokálním správcem front, vzdáleným správcem front nebo agentem MCA (Message Channel Agent) v souladu se podmínkami sítě.

Poznámka: Aplikace klienta a serveru, které potřebují výměnu posloupnosti zpráv k 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 do jiných instancí serverové aplikace.

Je-li pro frontu klastru zadán parametr MQOO_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 zadán parametr MQOO_BIND_NOT_FIXED.

Je-li hodnota MQOO_INQUIRE uvedena s MQOO_BIND_NOT_FIXED, pak po sobě jdoucích volání MQINQ pomocí tohoto manipulátoru může zjistit různé instance fronty klastru, ačkoli všechny instance mají stejné hodnoty atributu.

Hodnota MQOO_BIND_NOT_FIXED je platná pouze pro fronty a má vliv pouze na fronty klastru. Je-li tato volba zadána pro frontu, která není frontou klastru, je tato volba ignorována.

SKUPINA MQO_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby skupina zpráv byla alokována do stejné cílové instance.

Tato volba je platná pouze pro fronty a má vliv pouze na fronty klastru. Je-li tato volba zadána pro frontu, která není frontou klastru, je tato volba ignorována.

MQOO_BIND_AS_Q_DEF
Lokální správce front váže manipulátor fronty tak, jak je definován atributem fronty produktu **`DefBind`** . Hodnota tohoto atributu je buď MQBND_BIND_ON_OPEN, MQBND_BIND_NOT_FIXED, nebo MQBND_BIND_ON_GROUP.

Hodnota MQOO_BIND_AS_Q_DEF je výchozí hodnotou MQOO_BIND_ON_OPEN, MQOO_BIND_NOT_FIXED nebo MQOO_BIND_ON_GROUP není zadána.

Dokumentaci programu podpory MQOO_BIND_AS_Q_DEF. Není určeno, že tato volba se používá při použití jedné z dalších dvou voleb vázání, ale protože její hodnota je nula, nelze takové použití detekovat.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

ÁLNÍ_KONTEXT MQOO_SAVE_ALL_CONTEXT
Informace o kontextu jsou přidruženy k tomuto manipulátoru fronty. Tyto informace jsou nastaveny z kontextu jakékoli zprávy načtené pomocí tohoto popisovače. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tyto informace o kontextu mohou být předány 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 496.

Do doby, kdy byla zpráva úspěšně načtena, nelze předat kontext do fronty, která je vložena do fronty.

Zpráva načtená pomocí jedné z voleb procházení MQGMO_BROWSE_ * nemá uložené informace o kontextu (ačkoli jsou pole kontextu v parametru **`MsgDesc`** nastavena po procházení).

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty. Musí být zadána jedna z voleb MQOO_INPUT_ *.

KONTEXT MQOO_PASS_IDENTITY_CONTEXT
To umožňuje, aby byla volba MQPMO_PASS_IDENTITY_CONTEXT určena v parametru **`PutMsgOpts`** , když je zpráva vložena do fronty; to dává zprávě informace o kontextu identity

ze vstupní fronty, která byla otevřena pomocí volby MQOO_SAVE_ALL_CONTEXT. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

MQOO_PASS_ALL_CONTEXT, KONTEXT
To umožňuje uvedení volby MQPMO_PASS_ALL_CONTEXT do parametru **PutMsgOpts** , když je zpráva vložena do fronty; to dává zprávě informace o identitě a původu z vstupní fronty, která byla otevřena pomocí volby MQOO_SAVE_ALL_CONTEXT. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba implikuje MQOO_PASS_IDENTITY_CONTEXT, což nemusí být zadáno. Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

KONTEXT MQOO_SET_IDENTITY_CONTEXT
To umožní určení hodnoty MQPMO_SET_IDENTITY_CONTEXT v parametru **PutMsgOpts** při vložení zprávy do fronty; to dává zprávě informace o kontextu identity obsažené v parametru **MsgDesc** uvedeném na volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba implikuje MQOO_PASS_IDENTITY_CONTEXT, což nemusí být zadáno. Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

MQO_SET_ALL_CONTEXT,
To umožní určení hodnoty MQPMO_SET_ALL_CONTEXT v parametru **PutMsgOpts** při vložení zprávy do fronty; to dává zprávě informace o identitě a zdroji původu obsažené v parametru **MsgDesc** uvedeném v rámci volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba zahrnuje následující volby, které proto nemusí být zadány:

- KONTEXT MQOO_PASS_IDENTITY_CONTEXT
- MQOO_PASS_ALL_CONTEXT, KONTEXT
- KONTEXT MQOO_SET_IDENTITY_CONTEXT

Musí být zadána volba MQOO_OUTPUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

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:

- Jak klient, tak i vzdálený správce front musí být IBM WebSphere MQ 7.0 nebo novější.
- 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 určují, zda se přechodné zprávy odešlou klientovi před tím, než je aplikace požaduje. Pro volby čtení napřed se vztahují následující poznámky:

- Může být uvedena pouze jedna 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 jsou použitelné pouze v případě, že je zadán také jeden z položek MQOO_BROWSE, MQOO_INPUT_SHARED a MQOO_INPUT_EXCLUSIVE, i když se nejedná o chybu při určování těchto voleb pro MQOO_INQUIRE nebo MQOO_SET.
- Pokud aplikace není spuštěna jako klient produktu IBM MQ , jsou tyto volby ignorovány.

MQOO_NO_READ_AHEAD
Netrvalé zprávy nejsou odeslány klientovi před tím, než je aplikace požaduje.

MQOO_READ_AHEAD
Netrvalé zprávy jsou odeslány na klienta před tím, než je aplikace požaduje.

MQOO_READ_AHEAD_AS_Q_DEF
Chování dopředného čtení je určeno výchozím atributem dopředného čtení fronty, která se má otevřít. Toto je výchozí hodnota.

Další volby: Následující volby řídí kontrolu autorizace, co se stane, když je správce front uváděn do klidového stavu, zda má být rozlišeno jméno lokální fronty a výběrové vysílání:

MQO_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 *AlternateUserId* autorizován k otevření objektu s uvedenými volbami přístupu, bez ohledu na to, zda je identifikátor uživatele, pod kterým je aplikace spuštěna, oprávněn tak učinit. To však neplatí pro žádné zadané volby kontextu, které jsou však vždy zkontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.

Tato volba je platná pro všechny typy objektů.

UVÁDĚNÍ MQOO_FAIL_IF_QUIESCING
Volání MQOPEN selže, je-li správce front ve stavu uvedení do klidového stavu.

z/OS V produktu z/OSv případě aplikace 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 Přehled produktu IBM MQ MQI clients.

MQOO_RESOLVE_LOCAL_Q,
Vyplňte pole ResolvedQName ve struktuře MQOD s použitím názvu lokální fronty, která byla otevřena. Podobným způsobem je název ResolvedQMgrvyplněn názvem lokálního správce front, který je hostitelem lokální fronty. Je-li struktura MQOD menší než verze 3, je hodnota MQOO_RESOLVE_LOCAL_Q ignorována, protože nebyla vrácena žádná chyba.

Lokální fronta je vždy vrácena, je-li otevřena buď lokální alias, nebo modelová fronta, ale v tomto případě se nejedná o případ, kdy je například otevřena vzdálená fronta nebo jiná než lokální fronta klastru bez volby MQOO_RESOLVE_LOCAL_Q; název ResolvedQName a ResolvedQMgrse vyplní s názvem RemoteQName a RemoteQMgr, který se nachází v definici vzdálené fronty, nebo podobně jako vybraná vzdálená fronta klastru.

Určíte-li hodnotu MQOO_RESOLVE_LOCAL_Q při otevírání, například vzdálená fronta, ResolvedQName je přenosová fronta, do níž jsou zprávy vloženy. Název ResolvedQMgrje vyplněn názvem lokálního správce front, který je hostitelem přenosové fronty.

Máte-li oprávnění k procházení, vstupu nebo výstupu ve frontě, máte oprávnění k uvedení tohoto příznaku v rámci volání MQOPEN. Není třeba žádné zvláštní oprávnění.

Tato volba je platná pouze pro fronty a správce front.

MQOO_RESOLVE_LOCAL_TOPIC.
Vyplňte pole ResolvedQName ve struktuře MQOD s názvem otevřeného administrativního tématu.

MQOO_NO_MULTICAST
Zprávy publikování se neodesílají pomocí výběrového vysílání.

Tato volba je platná pouze s volbou MQOO_OUTPUT. Je-li zadán bez MQOO_OUTPUT, vrací se MQOPEN s MQRC_OPTIONS_ERROR.

Tato volba je platná pouze pro určité téma.

HOBJ
Typ: MQHOTBJ-výstup

Tento manipulátor představuje přístup, který byl vytvořen objektu. Musí být zadán při následných voláních IBM MQ , které pracují s objektem. Přestane být platný, když je vydáno volání MQCLOSE, nebo když se jednotka zpracování, která definuje rozsah popisovače, ukončí.

Rozsah vrácený manipulátorů objektu je stejný jako rozsah manipulátoru připojení zadaného při volání. Informace o rozsahu zpracování naleznete v tématu Parametr MQCONN-Hconn .

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') Vráceno více kódů příčiny.

Je-li *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Alias základní fronty není platný typ.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Prostředek Coupling Facility není k dispozici.

MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') Kontrola autorizace struktury prostředku Coupling Facility se nezdařila.

CHYBA MQRC_CF_STRUC_STRUCT
(2349, X'92D') Struktura prostředku Coupling Facility není platná.

MQRC_CF_STRU_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility selhala.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

MQRC_CF_STRU_LIST_HDR_IN_USE
(2347, X'92B') Hlavička prostředku Coupling-facility-záhlaví se používá.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

CHYBA MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Ukončení pracovní zátěže klastru se nezdařilo.

MQRC_CLUSTER_PUT_BLOKOVÁNO
(2268, X'8DC') Volání blokováno pro všechny fronty v klastru.

CHYBA MQRC_CLUSTER_RESOLVUTION_ERROR
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

CHYBA MQRC_CLUSTER_RESOURCE_
(2269, X'8DD') Chyba prostředku klastru.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Subsystém Db2 není k dispozici.

CHYBA MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198, X'896 ') Výchozí přenosová fronta není lokální.

CHYBA MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897 ') Chyba použití předvolené přenosové fronty.

CHYBA 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.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') Vráceno více kódů příčiny.

MQRC_NAME_IN_USE
(2201, X'899 ') Název se používá.

MQRC_NAME_NE_VALID_STAR_TYP
(2194, X'892 ') Název objektu není platný pro daný typ objektu.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

MQRC_OBJECT_ALREADY_EXISTS
(2100, X'834 ') Objekt existuje.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objekt je již otevřen s konfliktními volbami.

MQRC_OBJECT_LEVEL_INCOMPATIBLE
(2360, X' 938 ') Úroveň objektů není kompatibilní.

CHYBA MQRC_OBJECT_NAME_ERROR
(2152, X'868 ') Název objektu není platný.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927 ') Objekt není jedinečný.

CHYBA MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869 ') Název správce front objektu není platný.

CHYBA MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Záznamy objektů nejsou platné.

CHYBA MQRC_OBJECT_STRING_ERROR
(2441, X'0989 ') Pole objektového řetězce není platné

CHYBA MQRC_OBJECT_TYPE_ERROR
(2043, X'7FB') Typ objektu není platný.

CHYBA 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.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA OBJEKTU MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

ÚPLNÁ OPERACE MQRC_PAGESET_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_Q_DELETED
(2052, X'804 ') Fronta byla odstraněna.

CHYBA MQRC_Q_MGR_NAME_ERROR
(2058, X'80A') Název správce front není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

CHYBA MQRC_Q_TYPE_ERROR
(2057, X'809 ') Typ fronty není platný.

CHYBA MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet záznamů přítomných záznamů není platný.

CHYBA MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888 ') Název vzdálené fronty není platný.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

CHYBA MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') Bylo vydáno volání MQOPEN, MQPUT1 nebo MQSUB, ale byl zadán výběrový řetězec, který obsahoval chybu syntaxe.

UŽIVATELSKÁ PROCEDURA MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo zamítnuto uživatelskou procedurou pracovní zátěže klastru.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822 ') Neznámá 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 objektu.

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 nesprávnou úroveň.

CHYBA MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') Přenosová fronta není lokální.

CHYBA MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') Přenosová fronta s chybným použitím.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Obecné poznámky k použití

1. Otevíraný objekt je jeden z následujících:
   - Fronta pro:
     - Získat nebo procházet zprávy (pomocí volání MQGET)
     - Vložit zprávy (pomocí volání MQPUT)
     - Dotazovat se na atributy fronty (pomocí volání MQINQ)
     - Nastavení atributů fronty (pomocí volání MQSET)

     Je-li uvedená fronta modelová fronta, vytvoří se dynamická lokální fronta. Viz parametr **ObjDesc** popsaný v tématu “MQOPEN-Otevřít objekt” na stránce 726.

     Rozdělovník je speciální typ objektu fronty, který obsahuje seznam front. Lze ji otevřít pro vkládání zpráv, nikoli však k získání nebo procházení zpráv nebo k zjišťování či nastavení atributů. Další podrobnosti najdete v poznámce pod čarou 8.

     Fronta, která má `QSGDISP(GROUP)` , je speciální typ definice fronty, kterou nelze použít s voláními MQOPEN nebo MQPUT1 .
   - Seznam názvů k dotazu na názvy front v seznamu (pomocí volání MQINQ).

- Definice procesu k dotazům na atributy procesu (pomocí volání MQINQ).
- Správce front se dotáže na atributy lokálního správce front (pomocí volání MQINQ).
- Chcete-li publikovat zprávu (pomocí volání MQPUT), téma

2. Aplikace může otevřít stejný objekt více než jednou. Pro každé otevření je vrácen jiný popisovač objektu. Každý vrácený popisovač může být použit pro funkce, pro které bylo provedeno odpovídající otevření.
3. Je-li otevíraný objekt jiný než fronta klastru, všechna rozpoznání názvu v lokálním správci front se uskuteční v době volání MQOPEN. To může zahrnovat:
   - Vyřešení názvu lokální definice vzdálené fronty na název vzdáleného správce front a název, pod kterým je fronta známa ve vzdáleném správci front
   - Rozlišení názvu vzdáleného správce front na název lokální přenosové fronty
   - z/OS Pouze v produktu z/OS , rozpoznání názvu vzdáleného správce front s názvem sdílené přenosové fronty použité agentem IGQ (platí pouze v případě, že lokální a vzdálené správce front patří do stejné skupiny sdílení front)
   - Rozlišování aliasů k názvu základní fronty nebo objektu tématu.

   Mějte však na paměti, že následující volání MQINQ nebo MQSET pro manipulátor se týkají výhradně názvu, který byl otevřen, a nikoli objektu, který je výsledkem rozlišení názvu. Je-li například otevřený objekt alias, jsou atributy vrácené voláním MQINQ atributy aliasu, nikoli atributy základní fronty nebo objektu tématu, na které je alias interpretováno.

   Je-li otevíraný objekt fronta klastru, může v době volání MQOPEN dojít k rozpoznání názvu nebo může být odloženo na později. Bod, ve kterém je rozpoznání prováděno, je řízen volbami MQOO_BIND_ * uvedenými v rámci volání MQOPEN:

   - MQO_BIND_ON_OPEN
   - MQOO_BIND_NOT_FIXED
   - MQOO_BIND_AS_Q_DEF
   - SKUPINA MQO_BIND_ON_GROUP

   Další informace o rozlišování názvů pro fronty klastru najdete v tématu Rozlišování názvů .
4. Volání MQOPEN s volbou MQOO_BROWSE založí kurzor procházení pro použití s voláními MQGET, které určují manipulátor objektu a jednu z voleb procházení. To umožňuje skenování fronty, aniž by došlo ke změně jejího obsahu. Zprávu, která byla nalezena procházením, lze odebrat z fronty pomocí volby MQGMO_MSG_UNDER_CURSOR.

   Více kurzorů procházení může být aktivní pro jednu aplikaci vysláním několika požadavků MQOPEN pro stejnou frontu.
5. Aplikace spuštěné monitorem spouštěčů jsou předány názvu fronty přidružené k aplikaci při spuštění aplikace. Tento název fronty může být zadán v parametru **ObjDesc** pro otevření fronty. Další podrobnosti viz “MQTMC2 -zpráva spouštěče 2 (znakový formát)” na stránce 600 .

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:

- Jak klient, tak i vzdálený správce front musí být IBM WebSphere MQ 7.0 nebo novější.
- 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.

Pro použití voleb dopředného čtení se používají následující poznámky.

1. Volby dopředného čtení lze použít pouze v případě, že jsou určeny také volby MQOO_BROWSE, MQOO_INPUT_SHARED a MQOO_INPUT_EXCLUSIVE, pouze v případě, že jsou volby dopředného čtení použity. Pokud jsou volby dopředného čtení zadány s volbami MQOO_ INQUIRE nebo MQOO_SET, nebude vrácena chyba.
2. Čtení napřed není povoleno, je-li požadováno, pokud nejsou volby použité při prvním volání MQGET podporovány pro použití s dopředným čtením. Čtení napřed je také vypnuto, když se klient připojuje ke správci front, který nepodporuje dopředné čtení.
3. Pokud aplikace není spuštěna jako klient produktu IBM MQ , volby dopředného čtení jsou ignorovány.

Fronty klastru

Pro použití klastrových front se používají následující poznámky.

1. Je-li poprvé otevřena fronta klastru a lokální správce front není správce front úplného úložiště, obdrží lokální správce front informace o frontě klastru ze správce front úplného úložiště. Je-li síť zaneprázdněna, může správce lokální fronty přijmout několik sekund, aby mohl přijímat potřebné informace od správce front úložiště. V důsledku toho může aplikace, která vydala volání MQOPEN, čekat až 10 sekund, než se řízení vrátí z volání MQOPEN. Pokud lokální správce front v rámci této doby neobdrží potřebné informace o frontě klastru, volání selže s kódem příčiny MQRC_CLUSTER_RESOLUTION_ERROR.
2. Když se otevře fronta klastru a v klastru je více instancí fronty, instance se otevře v závislosti na volbách uvedených v volání MQOPEN:
   - Pokud uvedené volby zahrnují některou z následujících voleb:
     - MQOOK_BROWSE
     - MQO_INPUT_AS_Q_DEF
     - MQO_INPUT_EXCLUSIVE
     - MQO_INPUT_SHARED
     - MQOOK_SADA

     Instance otevřené fronty klastru musí být lokální instance. Pokud zde není žádná lokální instance fronty, volání MQOPEN selže.
   - Pokud uvedené volby nezahrnují žádnou z voleb popsaných dříve, ale zahrnují jednu nebo obě z následujících možností:
     - MQO_ DOTÁZAT SE
     - MQOOK_VÝSTUP

     instance otevřená je lokální instance, pokud existuje, a vzdálená instance jinak (pokud používáte předvolby CLWLUSEQ). Instance zvolená správcem front však může být změněna uživatelskou procedurou pracovní zátěže klastru (je-li k tomu nějaká).
3. Pokud existuje odběr pro danou frontu, ale není potvrzený úplným úložištěm, objekt není přítomen v klastru a volání se nezdaří s kódem příčiny MQRC_OBJECT_NAME.

Další informace o frontách klastru najdete v tématu Klastrové fronty.

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

Distribuční seznamy jsou podporovány v následujících prostředích:

- AIX
- IBM i
- Linux

- Windows Windows

a pro IBM MQ MQI clients připojené 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 řetězec nebo řetězec s hodnotou null.
   - `ObjectQMgrName` musí být prázdný řetězec nebo řetězec s hodnotou null.
   - `RecsPresent` musí být větší než nula.
   - Jeden z produktů `ObjectRecOffset` a `ObjectRecPtr` musí být nula a druhý nenulový.
   - Ne více než jeden z `ResponseRecOffset` a `ResponseRecPtr` může být nenulový.
   - Musí existovat `RecsPresent` záznamů objektů adresovaných buď `ObjectRecOffset` nebo `ObjectRecPtr`. Záznamy objektů musí být nastaveny na názvy cílových front, které se mají otevřít.
   - Je-li některý z produktů `ResponseRecOffset` a `ResponseRecPtr` nenulový, musí být přítomny záznamy odpovědí `RecsPresent` . Jsou nastavována správcem front, pokud je volání dokončeno s kódem příčiny MQRC_MULTIPLE_REASONS.

   MQOD version-2 lze také použít k otevření jedné fronty, která není v distribučním seznamu, tím, že zajistíte, že `RecsPresent` je nula.
2. V parametru **`Options`** jsou platné pouze následující volby otevření:
   - MQOOK_VÝSTUP
   - MQOO_PASS_ * _CONTEXT
   - MQOO_SET_ * _CONTEXT
   - MQO_ALTERNATE_USER_AUTHORITY.
   - UVÁDĚNÍ MQOO_FAIL_IF_QUIESCING
3. Cílové fronty v rozdělovníku mohou být lokální, alias nebo vzdálené fronty, ale nemohou být modelové fronty. Je-li zadána modelová fronta, tato fronta se neotevřou, s kódem příčiny MQRC_Q_TYPE_ERROR. To však nezabrání tomu, aby byly ostatní fronty v seznamu úspěšně otevřeny.
4. Kód dokončení a parametry kódu příčiny jsou nastaveny takto:
   - Pokud jsou operace otevření pro fronty v seznamu distribuce úspěšné nebo selžou stejným způsobem, jsou nastaveny parametry dokončení kódu dokončení a kódu příčiny popisující společný výsledek. Záznamy odpovědí MQRR (nejsou-li zadány aplikací) nejsou v tomto případě nastaveny.

     Je-li například každé otevření úspěšné, kód dokončení je nastaven na hodnotu MQCC_OK a kód příčiny je nastaven na hodnotu MQRC_NONE;, pokud každé otevření selže, protože žádná z front neexistuje, parametry jsou nastaveny na hodnotu MQCC_FAILED a MQRC_UNKNOWN_OBJECT_NAME.
   - Pokud operace otevření pro fronty v rozdělovníku nejsou všechny úspěšné nebo selžou stejným způsobem:
     - Je-li parametr kódu dokončení nastaven na hodnotu MQCC_WARNING, je-li alespoň jedno otevření úspěšné, a do stavu MQCC_FAILED, došlo-li k selhání.
     - Parametr kódu příčiny je nastaven na hodnotu MQRC_MULTIPLE_REASONS.
     - Záznamy odpovědí (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.
5. Když byl distribuční seznam úspěšně otevřen, popisovač `Hobj` vrácený voláním lze použít při následných voláních MQPUT k vložení zpráv do front v rozdělovníku a na volání MQCLOSE, aby se uvoll přístup k rozdělovníku. Jedinou platnou volbou zavření pro distribuční seznam je MQCO_NONE.

   Volání MQPUT1 lze také použít k vložení zprávy do distribučního seznamu; struktura MQOD, která definuje fronty v seznamu, je uvedena jako parametr v tomto volání.

6. Každý úspěšně otevřený cíl v rozdělovníku se počítá jako samostatný popisovač při kontrole, zda aplikace překročila povolený maximální počet popisovačů (viz atribut správce front **MaxHandles**). To platí i v případě, že se dvě nebo více míst určení v seznamu distribucí vyřeší do stejné fyzické fronty. Pokud volání MQOPEN nebo MQPUT1 pro distribuční seznam způsobí, že počet popisovačů v aplikaci převýší `MaxHandles`, volání selže s kódem příčiny MQRC_HANDLE_NOT_AVAILABLE.
7. Každé místo určení, které je úspěšně otevřeno, má hodnotu jeho atributu **OpenOutputCount** inkrementované o jednu. Pokud se dvě nebo více míst určení v seznamu distribucí vyřeší do stejné fyzické fronty, má tato fronta svůj atribut **OpenOutputCount** inkrementován počtem míst určení v seznamu distribucí, který se do této fronty rozejde.
8. Jakákoli změna definic front, která by způsobila, že se popisovač stanou neplatnými, byly fronty otevřeny jednotlivě (například změna v cestě vyřešení), nezpůsobí zneplatnění rozdělovníku pro seznam distribucí. Výsledkem je však selhání této konkrétní fronty, je-li popisovač distribučního seznamu použit v následném volání MQPUT.
9. Distribuční seznam může obsahovat pouze jedno místo určení.

Vzdálené fronty

Pro použití vzdálených front se používají následující poznámky.

Vzdálenou frontu lze zadat jedním ze dvou způsobů v parametru **ObjDesc** tohoto volání.

- Uvedením `ObjectName` název lokální definice vzdálené fronty. V tomto případě pojem `ObjectQMgrName` odkazuje na lokálního správce front a lze jej zadat jako mezery nebo (v programovacím jazyku C) prázdný řetězec.

  Ověření zabezpečení provedené lokálním správcem front ověřuje, zda je uživatel oprávněn k otevření lokální definice vzdálené fronty.
- Uvedením `ObjectName` název vzdálené fronty, jak je známo vzdálenému správci front. V tomto případě je `ObjectQMgrName` názvem vzdáleného správce front.

  Ověření zabezpečení provedené lokálním správcem front ověřuje, zda je uživatel autorizován k odesílání zpráv do přenosové fronty, která je výsledkem procesu rozlišování názvů.

V obou případech:

- Lokální správce front neodesílá do správce vzdálené fronty 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 vytvořil, není autorizován.

Další informace naleznete v polích `ObjectName` a `ObjectQMgrName` popsaných v příručce “MQOD-Popisovač objektu” na stránce 476 .

Objekty

Zabezpečení

Následující poznámky se vztahují k aspektům zabezpečení použití funkce MQOPEN.

Správce front provádí kontroly zabezpečení při vyvolání volání MQOPEN, aby bylo možné ověřit, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, příslušnou úroveň oprávnění, než je povolen přístup. Kontrola oprávnění se provádí na jméno objektu, který je otevíraný, a ne na názvu nebo názvech, výsledkem je, že byl vyřešen název.

Je-li otevíraný objekt alias fronta, která ukazuje na objekt tématu, provede správce front kontrolu zabezpečení s názvem alias fronty před provedením kontroly zabezpečení pro dané téma, jako kdyby byl objekt tématu použit přímo.

Je-li otevíraný objekt objektem tématu, ať už se samotným `ObjectName` nebo pomocí `ObjectString` (se založením `ObjectName`nebo bez něj), provede správce front kontrolu zabezpečení použitím

výsledného řetězce tématu, převzaté z objektu tématu uvedeného v produktu `ObjectName`, a je-li to nezbytné, zřetězením s objektem zadaným v produktu `ObjectString`a následným nalezením nejbližšího objektu tématu ve stromu témat nebo nad ním nalezeného ve stromu témat za účelem provedení kontroly zabezpečení. To nemusí být stejný objekt tématu, který byl zadán v produktu `ObjectName`.

Je-li otevíraný objekt modelová fronta, provede správce front úplnou kontrolu zabezpečení proti názvu modelové fronty a názvu vytvořené dynamické fronty. Je-li výsledná dynamická fronta otevřena explicitně, provede se další kontrola zabezpečení prostředku proti názvu dynamické fronty.

z/OS V systému z/OSprovádí správce front kontrolu 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 změnit, zatímco aplikace má otevřený objekt. V mnoha případech aplikace toto nezaznamenaní, ale u určitých atributů správce front označí popisovač jako již platný. Tyto atributy jsou:

- Jakýkoli atribut, který ovlivňuje rozpoznání názvu objektu. To platí bez ohledu na použité otevřené volby a zahrnuje následující:
  - Změna na atribut **BaseQName** fronty aliasů, která je otevřená.
  - Změna na atribut **TargetType** fronty aliasů, která je otevřená.
  - Změna atributů fronty **RemoteQName** nebo **RemoteQMgrName** pro všechny popisovače, které jsou otevřeny pro tuto frontu, nebo pro frontu, která se překládá prostřednictvím této definice jako alias správce front.
  - Jakákoli změna, která způsobí, že se aktuálně otevřená obsluha pro vzdálenou frontu vyřeší do jiné přenosové fronty, nebo aby se nevyhodnocla vůbec jedna. To může například zahrnovat:
    - Změna atributu **XmitQName** lokální definice vzdálené fronty bez ohledu na to, zda je definice použita pro frontu nebo pro alias správce front.
    - z/OS Pouze v systému z/OS , změna hodnoty atributu správce front **IntraGroupqueuing** nebo změna definice sdílené přenosové fronty (SYSTEM.QSG.TRANSMIT.QUEUE) používaná agentem IGQ.

    Existuje jedna výjimka pro toto: vytvoření nové přenosové fronty. Popisovač, který by byl interpretován jako tato fronta, byl při otevření popisovače přítomen, ale namísto toho vyřešen do výchozí přenosové fronty, není platný.
  - Změna na atribut správce front produktu **DefXmitQName** . V tomto případě jsou všechny otevřené popisovače, které se vyřešily do dříve pojmenované fronty (které se na něj budou interpretovat pouze proto, že se jednalo o výchozí přenosovou frontu), označeny jako neplatné. Ošetřeny, které byly pro tuto frontu rozpoznáné z jiných důvodů, nejsou ovlivněny.
- Atribut fronty **Shareability** , pokud existují dva nebo více manipulátorů, které aktuálně poskytují přístup MQOO_INPUT_SHARED pro tuto frontu, nebo pro frontu, která je převedena do této fronty. Pokud ano, všechny manipulátory, které jsou otevřeny pro tuto frontu, nebo pro frontu, která je přeložena do této fronty, jsou označena jako neplatná, bez ohledu na volby otevření.

  z/OS V systému z/OSjsou dříve popsané popisovače označeny jako neplatné, pokud jeden nebo více popisovačů aktuálně poskytuje přístup MQOO_INPUT_SHARED nebo MQOO_INPUT_EXCLERIVE do fronty.
- Atribut fronty produktu **Usage** pro všechny manipulátory, které jsou otevřeny pro tuto frontu, nebo pro frontu, která se interpretuje jako tato fronta bez ohledu na volby otevření.

Je-li popisovač označen jako neplatný, všechna následná volání (jiná než MQCLOSE) používající tento popisovač selžou s kódem příčiny MQRC_OBJECT_CHANGED. Aplikace musí vydat volání MQCLOSE (s použitím původní obslužné rutiny) a poté znovu otevřít frontu. Všechny nepotvrzené aktualizace oproti

starému popisovači z předchozích úspěšných volání lze stále potvrdit nebo odstranit, jak to vyžaduje logika aplikace.

Pokud změna atributu způsobí, že k tomu dojde, použijte speciální vynucený 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 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 musí být již otevřené.

Syntaxe

MQPUT (*Hconn*, *Hobj*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Reason*)

Parametry

Hconn

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota Hconn byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

HOBJ

Typ: MQHOTBJ-vstup

Tento popisovač představuje frontu, do níž je zpráva přidána, nebo téma, do kterého je zpráva publikována. Hodnota `Hobj` byla vrácena předchozím voláním MQOPEN, které určovalo volbu MQOO_OUTPUT.

MsgDesc

Typ: MQMD-I/O

Tato struktura popisuje atributy odesílané zprávy a přijímá informace o zprávě po dokončení požadavku na vložení. Podrobnosti viz “MQMD-Deskriptor zpráv” na stránce 419.

Pokud aplikace poskytuje version-1 MQMD, data zprávy mohou mít předponu ve struktuře MQMDE, aby určovaly hodnoty pro pole, která existují v MQMD version-2 , ale ne v version-1. Pole *Formát* v deskriptoru MQMD musí být nastaveno na hodnotu MQFMT_MD_EXTENSION, aby bylo zřejmé, že

je přítomen prvek MQMDE. Další informace viz část "MQMDE-Rozšíření deskriptoru zpráv" na stránce 468.

Aplikace nemusí poskytovat strukturu MQMD, je-li v polích `OriginalMsgHandle` nebo `NewMsgHandle` struktury MQPMO zadána platná obsluha zprávy. Pokud v některém z těchto polí není nic uvedeno, deskriptor zprávy bude převzaten z deskriptoru asociovaného s manipulátory zpráv.

Pokud používáte nebo plánujete použití rozhraní API, doporučujeme, abyste explicitně zadali strukturu MQMD a nepoužívali deskriptory zpráv přidružené k manipulátorům zpráv. Důvodem je skutečnost, že uživatelská procedura rozhraní API přidružená k volání MQPUT nebo MQPUT1 nemůže zjistit, které hodnoty MQMD používá správce front, aby dokončil požadavek MQPUT nebo MQPUT1 .

PutMsgOpts
Typ: MQPMO-input/output

Podrobnosti viz "MQPMO-Volby vložení zprávy" na stránce 496.

BufferLength
Typ: MQLONG-vstup

Délka zprávy v produktu `Buffer`. Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní mez pro `BufferLength` závisí na různých faktorech:

- Je-li cílem lokální fronta, nebo pokud se vyřeší do lokální fronty, horní limit závisí na tom, zda:
  - Lokální správce front podporuje segmentaci.
  - Odesílající aplikace uvádí příznak, který umožňuje správci front segmentovat zprávu. Tento parametr je MQMF_SEGMENTATION_ALLOWED a může být zadán buď v version-2 MQMD, nebo v MQMDE použitém s version-1 MQMD.

  Pokud jsou obě tyto podmínky splněny, `BufferLength` nemůže překročit 999 999 999 minus hodnotu pole `Offset` v MQMD. Nejdelší logická zpráva, která může být vložena, je proto 999 999 999 bajtů (když `Offset` je nula). Omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, však může vést k nižšímu omezení.

  Pokud jedna nebo obě předchozí podmínky nejsou splněny, `BufferLength` nemůže překročit menší hodnotu atributu **MaxMsgLength** fronty a atributu **MaxMsgLength** správce front.
- Je-li místem určení vzdálená fronta nebo je překládána do vzdálené fronty, použijí se podmínky pro lokální fronty, ale u každého správce front, přes který musí zpráva projít, aby dosáhla cílové fronty; zejména:
  1. Lokální přenosová fronta používaná k dočasnému ukládání zprávy v lokálním správci front
  2. Intermediační přenosové fronty (jsou-li nějaké) používané k ukládání zpráv ve správcích front na trase mezi lokálními a cílovými správci front.
  3. Cílová fronta v cílovém správci front

  Nejdelší zpráva, kterou lze vložit, se proto řídí nejrestriktivnějšími zprávami z těchto front a správců front.

  Je-li zpráva v přenosové frontě, jsou spolu s daty zprávy uloženy další informace a snižuje množství dat aplikace, které lze provést. V této situaci odečtěte bajty MQ_MSG_HEADER_LENGTH z hodnot `MaxMsgLength` přenosových front, když určujete limit pro `BufferLength`.

  Poznámka: Pouze nedodržení podmínky 1 může být diagnostikováno synchronně (s kódem příčiny MQRC_MSG_TOO_BIG_FOR_Q nebo MQRC_MSG_TOO_BIG_FOR_Q_MGR), když je zpráva vložena. Nejsou-li podmínky 2 nebo 3 splněny, je zpráva přesměrována do fronty nedoručených zpráv (nedoručené zprávy) buď v intermediačních správci front, nebo v cílovém správci front. Pokud k tomu dojde, vygeneruje se zpráva sestavy, pokud ji odesílatel požadoval.

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Jedná se o vyrovnávací paměť obsahující data aplikace, která se mají odeslat. Vyrovnávací paměť musí být zarovnána na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání je vhodné pro většinu

zpráv (včetně zpráv obsahujících záhlaví záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Pokud `Buffer` obsahuje znaková nebo číselná data, nastavte pole `CodedCharSetId` a `Encoding` v parametru **MsgDesc** na hodnoty odpovídající datům; to umožňuje přijímači zprávy převést data (je-li to nutné) na znakovou sadu a kódování používané příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT musí být ve znakové sadě a kódování lokálního správce front (daný atributem správce front **CodedCharSetId** a MQENC_NATIVE).

V programovacím jazyku C je tento parametr deklarován jako ukazatel-to-void; adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BufferLength** nulový, `Buffer` není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assembler s hodnotou null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující `CompCode`.

Je-li `CompCode` MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li `CompCode` MQCC_WARNING:

SKUPINA MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není úplná.

ZPRÁVA MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není úplná.

MQRC_INCONSISTENT_PERSISTENCE
(2185, X'889 ') Nekonzistentní specifikace perzistence.

NEKONZISTENCE MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') Vráceno více kódů příčiny.

MQRC_PRIORITY_EXCEEDS_MAXIMUM
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

VOLBA MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838 ') Volby Report v deskriptoru zpráv nebyly rozpoznány.

Je-li `CompCode` MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

ZMĚNĚNO ALIAS_ALIAS_MQRC_TARGETYPE_CHANGED
(2480, X'09B0') Typ cíle odběru se změnil z fronty na objekt tématu.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Unit of work backed out.

CHYBA MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CALL_INTERRUPTED
(2549, X'9F5') MQPUT nebo MQCMIT bylo přerušeno a zpracování opětovného připojení nemůže znovu vytvořit definitivní výsledek.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Prostředek Coupling Facility není k dispozici.

MQRC_CF_STRU_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility selhala.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

CHYBA MQRC_CFGR_ERROR
(2416, 'X' 970') struktura parametru skupiny PCF MQCFGR v datech zprávy je neplatná.

CHYBA MQRC_CFH_ERROR
(2235, X'8BB') Struktura hlavičky PCF není platná.

CHYBA MQRC_CFIF_ERROR
(2414, X'96E') Celočíselná struktura parametru filtru PCF v datech zprávy je neplatná.

CHYBA MQRC_CFIL_ERROR
(2236, X'8BC') Struktura parametru celočíselné struktury PCF nebo struktura parametru celého seznamu PCIF*64 nejsou platné.

CHYBA MQRC_CFIN_ERROR
(2237, X'8BD') Struktura celočíselného formátu PCF nebo struktura celočíselného parametru PCIF*64 nejsou platné.

CHYBA MQRC_CFSF_ERROR
(2415, X'96F') Struktura parametru filtru řetězce PCF v datech zprávy je neplatná.

CHYBA MQRC_CFSL_ERROR
(2238, X'8BE') Struktura řetězce seznamu řetězců PCF není platná.

CHYBA MQRC_CFST_ERROR
(2239, X'8BF') Struktura parametru řetězce PCF není platná.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

CHYBA MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Ukončení pracovní zátěže klastru se nezdařilo.

CHYBA MQRC_CLUSTER_RESOLVUTION_ERROR
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

CHYBA MQRC_CLUSTER_RESOURCE_
(2269, X'8DD') Chyba prostředku klastru.

MQRC_CED_NOT_VALID_FOR_XCF_Q
(2106, X'83A') Volba sestavy COD není platná pro frontu XCF.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CONTENT_ERROR
2554 (X'09FA') Obsah zprávy nemohl být analyzován za účelem určení, zda má být zpráva doručena odběrateli s rozšířeným voličem zpráv.

CHYBA OBJEKTU MQRC_CONTEXT_HANDLE_ERROR
(2097, X'831 ') Manipulátor fronty odkazovaný tak, aby neukládaný kontext.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832 ') Kontext není k dispozici pro uvedený popisovač fronty.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

CHYBA MQRC_DH_ERROR
(2135, X'857 ') Struktura hlavičky distribuce není platná.

CHYBA MQRC_DLH_ERROR
(2141, X'85D') Struktura záhlaví nedoručených zpráv není platná.

CHYBA MQRC_EF_ERROR
(2420, X' 974 ') Vložená struktura PCF není platná.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Doba vypršení platnosti není platná.

CHYBA MQRC_FEEDBACK_ERROR
(2014, X'7DE') Kód zpětné vazby není platný.

KONFLIKT MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální jednotky konfliktu práce.

CHYBA MQRC_GROUP_ID_ERROR
(2258, X'8D2') Identifikátor skupiny není platný.

FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HEADER_ERROR
(2142, X'85E') Struktura záhlaví MQ MQ není platná.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

CHYBA MQRC_IIH_ERROR
(2148, X'864 ') Struktura informačního záhlaví IMS není platná.

SKUPINA MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není úplná.

ZPRÁVA MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není úplná.

MQRC_INCONSISTENT_PERSISTENCE
(2185, X'889 ') Nekonzistentní specifikace perzistence.

NEKONZISTENCE MQRC_INCONSISTENT_UOW
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

KONFLIKT MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s místní jednotkou práce.

CHYBA MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zprávy není platný.

CHYBA MQRC_MDE_ERROR
(2248, X'8C8') Rozšíření deskriptoru zpráv není platné.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Chybí odpověď na frontu nebo MQPMO_SUPPRESS_REPLYTO byla použita.

MQRC_MISSING_WIH
(2332, X'91C') Data zprávy nezačínají řetězcem MQWIH.

CHYBA 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 je větší než maximum pro správce front.

CHYBA MQRC_MSG_TYPE_ERROR
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') 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 předání všech kontextů.

MQRC_NOT_OPEN_FOR_PASS_IDENT
(2094, X'82E') Fronta není otevřena pro kontext předání identity.

MQRC_NOT_OPEN_FOR_SET_ALL
(2095, X'82F') Fronta není otevřena pro nastavení všech kontextů.

MQRC_NOT_OPEN_FOR_SET_IDENT
(2096, X'830 ') Fronta není otevřena pro nastavení kontextu identity.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu byla od otevření změněna.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

CHYBA MQRC_OFFSET_ERROR
(2251, X'8CB') Odsazení segmentu zprávy není platné.

FUNKCE MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Původní délka není platná.

CHYBA OBJEKTU MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

ÚPLNÁ OPERACE MQRC_PAGESET_FULL
(2192, X'890 ') Externí paměťové médium je plné.

CHYBA MQRC_PCF_ERROR
(2149, X'865 ') struktur PCF nejsou platné.

CHYBA MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Perzistence není platná.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

CHYBA MQRC_PMO_ERROR
(2173, X'87D') Struktura volby vložení zprávy není platná.

CHYBOVÁ_CHYBA MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Příznaky vložení záznamu zprávy nejsou platné.

CHYBA MQRC_PRIORITY_ERROR
(2050, X'802 ') Priorita zprávy není platná.

MQRC_PUBLICATION_FAILURE.
(2502, X'9C6') Publikování nebylo doručeno žádnému z odběratelů.

MQRC_PUT_BLOKOVÁNO
(2051, X'803 ') Volání pro frontu, do které byla tato fronta přeložena, nebo téma, je blokováno pro volání fronty.

CHYBA MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Položit záznamy zpráv nejsou platné.

MQRC_PUT_NOT_RETAINED
(2479, X'09AF') Publikování nebylo možné zachovat.

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 není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

SELHÁNÍ OPERACE MQRC_RECONNECT_FAILED
(2548, X'9F4') Po opětovném připojení došlo k chybě při obnovení manipulátorů pro opětovné připojení připojení k tabulce.

CHYBA MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet záznamů přítomných záznamů není platný.

CHYBA NEPOVINNOSTI_SESTAVY_MQRC_REPORT
(2061, X'80D') Volby sestav v deskriptoru zpráv nejsou platné.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

CHYBA MQRC_RFH_ERROR
(2334, X'91E') MQRFH nebo struktura MQRFH2 nejsou platné.

CHYBA MQRC_RMH_ERROR
(2220, X'8AC') Struktura záhlaví referenční zprávy není platná.

MQRC_SEGMENT_LENGTH_ZERO
(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') Možný odběratel pro publikování existuje, ale správce front nemůže zkontrolovat, zda má být publikování odeslán odběrateli.

UŽIVATELSKÁ PROCEDURA MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo zamítnuto uživatelskou procedurou pracovní zátěže klastru.

CHYBA TŘÍDY MQRC_STORAGE_CLASS_ERROR
(2105, X'839 ') Chyba třídy úložiště.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

CHYBA MQRC_TM_ERROR
(2265, X'8D9') Struktura zprávy spouštěče není platná.

CHYBA MQRC_TMC_
(2191, X'88F') Struktura zpráv spouštěče znaků není platná.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

CHYBA MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotek práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unit of work not available for the queue manager to use.

CHYBA MQRC_WIH_ERROR
(2333, X'91D') Struktura MQWIH není platná.

VERZE MQRC_WRONG_MD_VERSION
(2257, X'8D1') Chybná verze dodaných MQMD.

CHYBA MQRC_XQHL_ERROR
(2260, X'8D4') Struktura záhlaví přenosové fronty není platná.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití tématu

1. Pro použití témat se používají následující poznámky:

a. Pokud pomocí příkazu MQPUT publikujete zprávy v tématu, kde jeden nebo více účastníků daného tématu nemohou být předány publikování kvůli problému s frontou odběratele (například je úplný), kód příčiny vrácený do volání MQPUT a chování doručení závisí na nastavení atributů PMSGDLV nebo NPMSGDLV na TOPIC. Všimněte si doručování publikování do fronty nedoručených zpráv, je-li uveden parametr MQRO_DEAD_LETTER_Q nebo když je zpráva MQRO_DISCARD_MSG uvedena jako úspěšná, považuje se za úspěšné doručení této zprávy. Pokud není dodána žádná z příruček, vrátí se MQPUT s MQRC_PUBLICATION_FAILURE. K tomu může dojít v následujících případech:

   - Zpráva se publikuje do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavené na ALL a každý odběr (trvalý či nikoli) má frontu, která nemůže přijmout publikaci.
   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALLDUR a trvalý odběr má frontu, která nemůže přijmout publikování.

   MQPUT se může vrátit s MQRC_NONE, přestože publikace nemohou být doručeny některým odběratelům v následujících případech:

   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na trvání zprávy) nastavené na ALLAVAIL a jakýkoli odběr, trvalý či nikoli, má frontu, která nemůže přijmout publikaci.
   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavené na ALLDUR a dočasné předplatné má frontu, která nemůže přijmout publikaci.

   Můžete použít atribut tématu USEDLQ k určení, zda se fronta nedoručených zpráv používá, když nelze publikační zprávy doručit do správné fronty odběratele. Další informace o použití volby USEDLQ viz DEFINE TOPIC.

b. Pokud nejsou k používanovanému tématu žádné odběratele, publikovaná zpráva se neodešle do žádné fronty a nebude vyřazena. Nezáleží na tom, zda je zpráva trvalá nebo trvalá, nebo zda má neomezené vypršení platnosti nebo má dobu vypršení platnosti, je stále vyřazena, pokud nejsou žádní odběratelé. Výjimkou je případ, kdy má být zpráva uchována, v takovém případě, ačkoli se neodesílá do front žádné odběratele, je uložena proti tématu, které má být doručeno na všechny nové odběry nebo na odběratele, kteří žádají o zachované publikace pomocí MQSUBRQ.

MQPUT a MQPUT1

Můžete použít volání MQPUT i MQPUT1 k vložení zpráv do fronty; volání, které má být používáno, závisí na okolnostech.

- Chcete-li umístit více zpráv do stejné fronty, použijte volání MQPUT.

  Bylo zadáno volání MQOPEN s určením volby MQOO_OUTPUT, za nímž následuje jeden nebo více požadavků MQPUT pro přidání zpráv do fronty. Nakonec je fronta uzavřena s voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .

- Použijte volání MQPUT1 k vložení pouze jedné zprávy do fronty.

  Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která musí být vydána.

Cílové fronty

Pro použití cílových front se používají následující poznámky:

1. Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny podmínky podrobné. Některé podmínky se vztahují na lokální i vzdálené cílové fronty; ostatní podmínky se vztahují pouze na vzdálené cílové fronty.

   Podmínky, které platí pro lokální a vzdálené cílové fronty

   - Všechna volání MQPUT se nacházejí ve stejné pracovní jednotce, nebo žádná z nich není v rámci pracovní jednotky.

     Uvědomte si, že když jsou zprávy vloženy do konkrétní fronty v rámci jedné pracovní jednotky, zprávy z jiných aplikací mohou být promíchány s posloupností zpráv ve frontě.

- Všechna volání MQPUT jsou prováděna pomocí stejného popisovače objektu *Hobj*.

  V některých prostředích je posloupnost zpráv také zachována při použití různých manipulátorů objektů, jsou-li volání prováděna ze stejné aplikace. Význam *stejné aplikace* je určen prostředím:

  – z/OSV systému z/OSje aplikace následující:
    - Pro CICS, úloha CICS
    - Pro IMSje úloha
    - Pro dávku produktu z/OS : úloha
  – IBM iV systému IBM ije aplikací úloha.
  – ALWV systému AIX, Linux, and Windowsje aplikace podproces.
- Všechny zprávy mají stejnou prioritu.
- Zprávy nejsou vloženy do fronty klastru s uvedeným parametrem MQOO_BIND_NOT_FIXED (nebo s MQOO_BIND_AS_Q_DEF, pokud má atribut fronty DefBind hodnotu MQBND_BIND_NOT_FIXED).

Další podmínky, které platí pro vzdálené cílové fronty

- Z odesílajícího správce front je k dispozici pouze jedna cesta ke správci cílové fronty.

  Pokud by některé zprávy v posloupnosti mohly pokračovat v jiné cestě (například kvůli změně konfigurace, vyrovnávání provozu nebo výběru cesty na základě velikosti zprávy), nelze zaručit pořadí zpráv v cílovém správci front.
- Zprávy nejsou dočasně umístěny do front nedoručených zpráv v odesílající, mezilehlé nebo cílové správci front.

  Je-li jedna nebo více zpráv dočasně umístěna do fronty nedoručených zpráv (například z důvodu dočasného zaplnění přenosové fronty nebo cílové fronty), mohou být zprávy doručeny do cílové fronty mimo pořadí.
- Zprávy jsou buď všechny trvalé, nebo všechny dočasné.

  Má-li kanál na trase mezi odesílajícím a cílovým správcem front nastaven atribut **NonPersistentMsgSpeed** na hodnotu MQNPMS_FAST, přechodné zprávy mohou skákat před trvalými zprávami, což má za následek pořadí trvalých zpráv vzhledem k nezachovalým netrvalým zprávám. Avšak pořadí trvalých zpráv ve vztahu k sobě navzájem a přechodných zpráv relativně k sobě navzájem je zachováno.

Pokud tyto podmínky nejsou splněny, můžete použít skupiny zpráv k uchování pořadí zpráv, ale to vyžaduje odesílající i přijímající aplikace k použití podpory seskupování zpráv. Další informace o skupinách zpráv viz:

- pole MQMD- MsgFlags
- MQPMO_LOGICAL_ORDER
- MQGMO_LOGICAL_ORDER

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

Distribuční seznamy jsou podporovány v následujících prostředích:

- AIXAIX
- IBM iIBM i
- LinuxLinux
- WindowsWindows

a pro IBM MQ MQI clients připojené k těmto systémům.

1. Do distribučního seznamu můžete vložit zprávy pomocí buď version-1 , nebo version-2 MQPMO. Pokud použijete položku version-1 MQPMO (nebo version-2 MQPMO s hodnotou `RecsPresent` rovnou nule), může aplikace poskytnout žádné záznamy vložení zpráv nebo záznamy odpovědí. Nemůžete identifikovat fronty, které se setkají s chybami, pokud je zpráva úspěšně odeslána do některých front v rozdělovníku a ne u jiných.

   Pokud aplikace poskytuje záznamy o vložení zpráv nebo záznamy odpovědí, nastavte pole `Version` na hodnotu MQPMO_VERSION_2.

   Můžete také použít version-2 MQPMO k odeslání zpráv do jediné fronty, která není v rozdělovníku, tím, že zajistíte, že `RecsPresent` je nula.
2. Kód dokončení a parametry kódu příčiny jsou nastaveny takto:
   - Pokud se vložení do front v rozdělovníku všechny nezdaří nebo selžou stejným způsobem, nastaví se kód dokončení a parametry kódu příčiny, aby popisoval společný výsledek. Záznamy odpovědí MQRR (nejsou-li zadány aplikací) nejsou v tomto případě nastaveny.

     Je-li například každé vložení úspěšné, kód dokončení a kód příčiny jsou nastaveny na MQCC_OK a MQRC_NONE;, pokud každé vložení selže, protože všechny fronty jsou blokovány pro operace put, jsou parametry nastaveny na hodnotu MQCC_FAILED a MQRC_PUT_INHIBITED.
   - Pokud vložení do front v rozdělovníku není úspěšné nebo selže stejným způsobem:
     - Je-li parametr kódu dokončení nastaven na hodnotu MQCC_WARNING, je-li alespoň jedna operace úspěšná, a do stavu MQCC_FAILED, došlo-li k selhání.
     - Parametr kódu příčiny je nastaven na hodnotu MQRC_MULTIPLE_REASONS.
     - Záznamy odpovědí (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.

     Pokud vložení do cíle selže, protože otevření pro toto místo určení se nezdařilo, pole v záznamu odpovědi jsou nastavena na hodnotu MQCC_FAILED a MQRC_OPEN_FAILED;, že místo určení je zahrnuto v produktu `InvalidDestCount`.
3. Pokud se cíl v rozdělovníku interpretuje jako lokální fronta, zpráva se umístí do této fronty v normálním formátu (tj. ne jako zpráva rozdělovníku). Pokud se do stejné lokální fronty vyhodnotí více než jeden cíl, jedna zpráva se umístí do fronty pro každé takové místo určení.

   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, lze do přenosové fronty umístit jednu zprávu distribučního seznamu obsahující taková místa určení i v případě, že tato místa určení nesousedí v seznamu míst určení poskytovaného danou 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 tvaru se umístí do přenosové fronty pro každé místo určení, které používá danou přenosovou frontu.

   Je-li distribuční seznam s daty zprávy aplikace příliš velký pro přenosovou frontu, zpráva distribučního seznamu se rozdělí na menší zprávy seznamu distribučních seznamů, z nichž každá obsahuje méně míst určení. Pokud se data zprávy aplikací pouze hodí do fronty, zprávy distribučního seznamu nelze vůbec použít a správce front vygeneruje jednu kopii zprávy v normálním formátu pro každý cíl, který používá danou přenosovou frontu.

   Pokud různá místa určení mají jinou prioritu zprávy nebo perzistenci zpráv (může k tomu dojít, když aplikace specifikuje MQPRI_PRIORITY_AS_Q_DEF nebo MQPER_PERSISTENCE_AS_Q_DEF), zprávy nejsou ve stejné zprávě distribučního seznamu. Namísto toho správce front generuje tolik zpráv v seznamu distribuce, kolik je třeba k umístění různých hodnot priority a perzistence.
4. Typ vložení do distribučního seznamu může mít za následek:
   - jedna zpráva distribučního seznamu nebo
   - počet menších zpráv v seznamu rozdělení, nebo
   - Směs zpráv distribučního seznamu a normálních zpráv, nebo
   - Pouze normální zprávy.

Který z výše uvedených situací nastane, závisí na tom, zda:

- Místa určení v seznamu jsou lokální, vzdálená, nebo směs.
- Místa určení mají stejnou prioritu zpráv a perzistenci zpráv.
- Přenosové fronty mohou obsahovat zprávy distribučního seznamu.
- Maximální délka zpráv pro přenosové fronty je dostatečně velká, aby pojmula zprávu do formuláře rozdělovníku.

Avšak bez ohledu na to, která z výše uvedených situací nastane, každá *fyzická* zpráva (tj. každá normální zpráva nebo zpráva distribučního seznamu, která je výsledkem příkazu put) se počítá jako jediná zpráva *jedna* , když:

- Kontrola, zda aplikace překročila povolený maximální počet zpráv v pracovní jednotce (viz atribut správce front produktu **MaxUncommittedMsgs**).
- Kontrola, zda jsou podmínky spouštěče splněny.
- Zvyšuje hloubku fronty a kontroluje, zda by byla překročena maximální hloubka fronty fronty.

5. Jakákoli změna definic front, která by způsobila, že se popisovač stanou neplatnými, byly fronty otevřeny jednotlivě (například změna v cestě vyřešení), nezpůsobí zneplatnění rozdělovníku pro seznam distribucí. Výsledkem je však selhání této konkrétní fronty, je-li popisovač distribučního seznamu použit v následném volání MQPUT.

Záhlaví

Je-li zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátku dat zprávy aplikace, provede správce front určité kontroly struktury záhlaví, aby ověřil, zda jsou platné. Pokud správce front zjistí chybu, volání selže s příslušným kódem příčiny. Provedená kontrola se liší podle konkrétních konstrukcí, které jsou k dispozici:

- Kontroly se provádějí pouze tehdy, je-li pro volání MQPUT nebo MQPUT1 použit MQMD version-2 nebo novější. Kontroly nejsou provedeny, je-li použit version-1 MQMD, i když je na začátku dat zprávy přítomen MQMDE.
- Struktury, které nejsou podporovány lokálním správcem front a strukturám následujících po prvním MQDLH ve zprávě, nejsou ověřeny.
- Struktury MQDH a MQMDE jsou správcem front kompletně ověřeny.
- Další struktury jsou ověřovány částečně správcem front (nejsou kontrolována všechna pole).

Mezi obecné kontroly prováděné správcem front patří následující:

- Pole `StrucId` musí být platné.
- Pole `Version` musí být platné.
- Pole `StrucLength` musí uvádět hodnotu, která je dostatečně velká, aby zahrnovala strukturu a data s proměnnou délkou, která tvoří část struktury.
- Pole `CodedCharSetId` nesmí být nula ani záporná hodnota, která není platná (hodnota MQCCSI_DEFAULT, MQCCSI_EMBEDDED, MQCCSI_Q_MGR a MQCCSI_UNDEFINED) není platná 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 konstrukcí musí být splněny tyto podmínky:

- Součet délek struktur v rámci zprávy PCF se musí rovnat délce určené parametrem **BufferLength** na 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 přípustné zkrácené struktury:
  - Zprávy, které jsou zprávami sestavy.

  - Zprávy příkazu PCF.
  - Zprávy obsahující strukturu MQDLH. (Struktury následující za prvním MQDLH mohou být zkráceny; struktury předcházející MQDLH nemohou.)
- Struktura IBM MQ nesmí být rozdělena na dva nebo více segmentů. Struktura musí být obsažena zcela v rámci jednoho segmentu.

Vyrovnávací paměť

V případě programovacího jazyka Visual Basic platí tyto body:

- Je-li velikost parametru **Buffer** menší než délka zadaná 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, není typu `String`, použijte příkazVolání MQPUTAny v místě MQPUT.

  Volání MQPUTAny má stejné parametry jako volání MQPUT, s výjimkou případů, kdy je parametr **Buffer** deklarován jako typ `Any`, což umožňuje umístění libovolného typu dat do fronty. To však znamená, že produkt `Buffer` nelze zkontrolovat, aby se zajistilo, že velikost bude mít 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 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 -Vložení jedné zprávy

Volání MQPUT1 vloží jednu zprávu do fronty nebo distribuční seznam nebo do tématu.

Fronta, distribuční seznam nebo téma nemusí být otevřené.

Syntaxe

MQPUT1 (*Hconn*, *ObjDesc*, *MsgDesc*, *PutMsgOpts*, *BufferLength*, *Buffer*, *CompCode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

ObjDesc
Typ: MQOD-input/output

Jedná se o strukturu, která identifikuje frontu, do níž je zpráva přidána, nebo téma, do kterého je zpráva publikována. Podrobnosti viz "MQOD-Popisovač objektu" na stránce 476.

Je-li struktura fronta, uživatel musí mít autorizaci k otevření fronty pro výstup. Fronta nesmí být modelová fronta.

MsgDesc
Typ: MQMD-I/O

Tato struktura popisuje atributy odesílané zprávy a přijímá informace zpětné vazby po dokončení požadavku na vložení. Podrobnosti viz "MQMD-Deskriptor zpráv" na stránce 419.

Pokud aplikace poskytuje version-1 MQMD, data zprávy mohou mít předponu ve struktuře MQMDE, aby určovaly hodnoty pro pole, která existují v MQMD version-2 , ale ne v version-1. Nastavte pole `Format` v MQMD na MQFMT_MD_EXTENSION, abyste označili, že je přítomen MQMDE. Další informace viz část "MQMDE-Rozšíření deskriptoru zpráv" na stránce 468.

Aplikace nemusí poskytovat strukturu MQMD, je-li v poli `MsgHandle` struktury MQGMO nebo v polích `OriginalMsgHandle` nebo `NewMsgHandle` struktury MQPMO dodána platná obsluha zprávy. Pokud v některém z těchto polí není nic uvedeno, deskriptor zprávy bude převzaten z deskriptoru asociovaného s manipulátory zpráv.

PutMsgOpts
Typ: MQPMO-input/output

Podrobnosti viz "MQPMO-Volby vložení zprávy" na stránce 496.

BufferLength
Typ: MQLONG-vstup

Délka zprávy v produktu `Buffer`. Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní limit závisí na různých faktorech; viz "MQPUT-Vložit zprávu" na stránce 743 , kde je uveden popis parametru **`BufferLength`** .

Vyrovnávací paměť
Typ: MQBYTExBufferDélka-vstup

Jedná se o vyrovnávací paměť obsahující data zprávy aplikace, která se mají odeslat. Zarovnejte vyrovnávací paměť na hranici odpovídající povaze dat ve zprávě. 4bajtové zarovnání je vhodné pro většinu zpráv (včetně zpráv obsahujících záhlaví záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Pokud `Buffer` obsahuje znaková nebo číselná data, nastavte pole `CodedCharSetId` a `Encoding` v parametru **`MsgDesc`** na hodnoty odpovídající datům; to umožňuje přijímači zprávy převést data (je-li to nutné) na znakovou sadu a kódování používané příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT1 musí být ve znakové sadě a kódování lokálního správce front (daný atributem správce front **`CodedCharSetId`** a MQENC_NATIVE).

V programovacím jazyku C je tento parametr deklarován jako ukazatel-to-void; adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BufferLength** nulový, Buffer není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assembler s hodnotou null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující CompCode.

Je-li CompCode MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li CompCode MQCC_WARNING:

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') Vráceno více kódů příčiny.

SKUPINA MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není úplná.

ZPRÁVA MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není úplná.

MQRC_PRIORITY_EXCEEDS_MAXIMUM
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

VOLBA MQRC_UNKNOWN_REPORT_OPTION
(2104, X'838 ') Volby sestav v deskriptoru zprávy nebyly rozpoznány.

Je-li CompCode MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_ALIAS_BASE_Q_TYPE_ERROR
(2001, X'7D1') Alias základní fronty není platný typ.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_BACKED_OUT
(2003, X'7D3') Unit of work backed out.

CHYBA MQRC_BUFFER_ERROR
(2004, X'7D4') Parametr vyrovnávací paměti není platný.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') zařízení pro spojení není k dispozici.

MQRC_CF_STRUC_AUTH_FAILED
(2348, X'92C') Kontrola autorizace struktury prostředku Coupling Facility se nezdařila.

CHYBA MQRC_CF_STRUC_STRUCT
(2349, X'92D') Struktura prostředku Coupling Facility není platná.

MQRC_CF_STRU_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility selhala.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

MQRC_CF_STRU_LIST_HDR_IN_USE
(2347, X'92B') Hlavička prostředku Coupling-facility-záhlaví se používá.

CHYBA MQRC_CFGR_ERROR
(2416, 'X' 970') struktura parametru skupiny PCF MQCFGR v datech zprávy je neplatná.

CHYBA MQRC_CFH_ERROR
(2235, X'8BB') Struktura hlavičky PCF není platná.

CHYBA MQRC_CFIF_ERROR
(2414, X'96E') Celočíselná struktura parametru filtru PCF v datech zprávy je neplatná.

CHYBA MQRC_CFIL_ERROR
(2236, X'8BC') Struktura parametru celočíselné struktury PCF nebo struktura parametru celého seznamu PCIF*64 nejsou platné.

CHYBA MQRC_CFIN_ERROR
(2237, X'8BD') Struktura celočíselného formátu PCF nebo struktura celočíselného parametru PCIF*64 nejsou platné.

CHYBA MQRC_CFSF_ERROR
(2415, X'96F') Struktura parametru filtru řetězce PCF v datech zprávy je neplatná.

CHYBA MQRC_CFSL_ERROR
(2238, X'8BE') Struktura řetězce seznamu řetězců PCF není platná.

CHYBA MQRC_CFST_ERROR
(2239, X'8BF') Struktura parametru řetězce PCF není platná.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

CHYBA MQRC_CLUSTER_EXIT_ERROR
(2266, X'8DA') Ukončení pracovní zátěže klastru se nezdařilo.

CHYBA MQRC_CLUSTER_RESOLVUTION_ERROR
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

CHYBA MQRC_CLUSTER_RESOURCE_
(2269, X'8DD') Chyba prostředku klastru.

MQRC_CED_NOT_VALID_FOR_XCF_Q
(2106, X'83A') Volba sestavy COD není platná pro frontu XCF.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CONTENT_ERROR
2554 (X'09FA') Obsah zprávy nebylo možné analyzovat a určit, zda může být zpráva doručena odběrateli s rozšířeným voličem zpráv.

CHYBA OBJEKTU MQRC_CONTEXT_HANDLE_ERROR
(2097, X'831 ') Manipulátor fronty odkazovaný tak, aby neukládaný kontext.

MQRC_CONTEXT_NOT_AVAILABLE
(2098, X'832 ') Kontext není k dispozici pro uvedený popisovač fronty.

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Subsystém Db2 není k dispozici.

CHYBA MQRC_DEF_XMIT_Q_TYPE_ERROR
(2198, X'896 ') Výchozí přenosová fronta není lokální.

CHYBA MQRC_DEF_XMIT_Q_USAGE_ERROR
(2199, X'897 ') Chyba použití předvolené přenosové fronty.

CHYBA MQRC_DH_ERROR
(2135, X'857 ') Struktura hlavičky distribuce není platná.

CHYBA MQRC_DLH_ERROR
(2141, X'85D') Struktura záhlaví nedoručených zpráv není platná.

CHYBA MQRC_EF_ERROR
(2420, X' 974 ') Vložená struktura PCF není platná.

MQRC_EXPIRY_ERROR
(2013, X'7DD') Doba vypršení platnosti není platná.

CHYBA MQRC_FEEDBACK_ERROR
(2014, X'7DE') Kód zpětné vazby není platný.

KONFLIKT MQRC_GLOBAL_UOW_CONFLICT
(2351, X'92F') Globální jednotky konfliktu práce.

CHYBA MQRC_GROUP_ID_ERROR
(2258, X'8D2') Identifikátor skupiny není platný.

FUNKCE MQRC_HANDLE_IN_USE_FOR_UOW
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'7E1') Nejsou k dispozici žádné další popisovače.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HEADER_ERROR
(2142, X'85E') struktura záhlaví IBM MQ není platná.

CHYBA MQRC_IIH_ERROR
(2148, X'864 ') Struktura informačního záhlaví IMS není platná.

KONFLIKT MQRC_LOCAL_UOW_CONFLICT
(2352, X' 930 ') Globální jednotka práce je v konfliktu s místní jednotkou práce.

CHYBA MQRC_MD_ERROR
(2026, X'7EA') Deskriptor zprávy není platný.

CHYBA MQRC_MDE_ERROR
(2248, X'8C8') Rozšíření deskriptoru zpráv není platné.

MQRC_MISSING_REPLY_TO_Q
(2027, X'7EB') Chybí odpověď na frontu.

MQRC_MISSING_WIH
(2332, X'91C') Data zprávy nezačínají řetězcem MQWIH.

CHYBA 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 je větší než maximum pro správce front.

CHYBA MQRC_MSG_TYPE_ERROR
(2029, X'7ED') Typ zprávy v deskriptoru zprávy není platný.

MQRC_MULTIPLE_PŘÍČINY
(2136, X'858 ') Vráceno více kódů příčiny.

MQRC_NO_DESTINATIONS_AVAILABLE
(2270, X'8DE') Nejsou k dispozici žádné cílové fronty.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

MQRC_OBJECT_IN_USE
(2042, X'7FA') Objekt je již otevřen s konfliktními volbami.

MQRC_OBJECT_LEVEL_INCOMPATIBLE
(2360, X' 938 ') Úroveň objektů není kompatibilní.

CHYBA MQRC_OBJECT_NAME_ERROR
(2152, X'868 ') Název objektu není platný.

MQRC_OBJECT_NOT_UNIQUE
(2343, X' 927 ') Objekt není jedinečný.

CHYBA MQRC_OBJECT_Q_MGR_NAME_ERROR
(2153, X'869 ') Název správce front objektu není platný.

CHYBA MQRC_OBJECT_RECORDS_ERROR
(2155, X'86B') Záznamy objektů nejsou platné.

CHYBA MQRC_OBJECT_TYPE_ERROR
(2043, X'7FB') Typ objektu není platný.

CHYBA MQRC_OD_ERROR
(2044, X'7FC') Struktura deskriptoru objektu není platná.

CHYBA MQRC_OFFSET_ERROR
(2251, X'8CB') Odsazení segmentu zprávy není platné.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

MQRC_ORIGINAL_LENGTH_ERROR
(2252, X'8CC') Původní délka není platná.

CHYBA OBJEKTU MQRC_PAGESET_ERROR
(2193, X'891 ') Chyba při přístupu k datové sadě sady stránek.

ÚPLNÁ OPERACE MQRC_PAGESET_FULL
(2192, X'890 ') Externí paměťové médium je plné.

CHYBA MQRC_PCF_ERROR
(2149, X'865 ') struktur PCF nejsou platné.

CHYBA MQRC_PERSISTENCE_ERROR
(2047, X'7FF') Perzistence není platná.

MQRC_PERSISTENT_NOT_ALLOWED
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

CHYBA MQRC_PMO_ERROR
(2173, X'87D') Struktura volby vložení zprávy není platná.

CHYBOVÁ_CHYBA MQRC_PMO_RECORD_FLAGS_ERROR
(2158, X'86E') Příznaky vložení záznamu zprávy nejsou platné.

CHYBA MQRC_PRIORITY_ERROR
(2050, X'802 ') Priorita zprávy není platná.

MQRC_PUBLICATION_FAILURE.
(2502, X'9C6') Publikování nebylo doručeno žádnému z odběratelů.

MQRC_PUT_BLOKOVÁNO
(2051, X'803 ') Volání s blokováno pro frontu.

CHYBA MQRC_PUT_MSG_RECORDS_ERROR
(2159, X'86F') Položit záznamy zpráv nejsou platné.

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 není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
(2161, X'871 ') Správce front je uváděn do klidového stavu.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

MQRC_Q_SPACE_NOT_AVAILABLE
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

CHYBA MQRC_Q_TYPE_ERROR
(2057, X'809 ') Typ fronty není platný.

CHYBA MQRC_RECS_PRESENT_ERROR
(2154, X'86A') Počet záznamů přítomných záznamů není platný.

CHYBA MQRC_REMOTE_Q_NAME_ERROR
(2184, X'888 ') Název vzdálené fronty není platný.

CHYBA NEPOVINNOSTI_SESTAVY_MQRC_REPORT
(2061, X'80D') Volby sestav v deskriptoru zpráv nejsou platné.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_RESPONSE_RECORDS_ERROR
(2156, X'86C') Záznamy odpovědí nejsou platné.

CHYBA MQRC_RFH_ERROR
(2334, X'91E') MQRFH nebo struktura MQRFH2 nejsou platné.

CHYBA MQRC_RMH_ERROR
(2220, X'8AC') Struktura záhlaví referenční zprávy není platná.

MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

MQRC_SEGMENT_LENGTH_ZERO
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Možný odběratel pro publikování existuje, ale správce front nemůže zkontrolovat, zda má být publikování odeslán odběrateli.

UŽIVATELSKÁ PROCEDURA MQRC_STOPPED_BY_CLUSTER_EXIT
(2188, X'88C') Volání bylo zamítnuto uživatelskou procedurou pracovní zátěže klastru.

CHYBA TŘÍDY MQRC_STORAGE_CLASS_ERROR
(2105, X'839 ') Chyba třídy úložiště.

MQRC_STORAGE_MEDIUM_FULL
(2192, X'890 ') Externí paměťové médium je plné.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

MQRC_SYNCPOINT_LIMIT_REACHED
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

MQRC_SYNCPOINT_NOT_AVAILABLE
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

CHYBA MQRC_TM_ERROR
(2265, X'8D9') Struktura zprávy spouštěče není platná.

CHYBA MQRC_TMC_
(2191, X'88F') Struktura zpráv spouštěče znaků není platná.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

MQRC_UNKNOWN_ALIAS_BASE_Q
(2082, X'822 ') Neznámá 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 objektu.

MQRC_UNKNOWN_REMOTE_Q_MGR
(2087, X'827 ') Neznámý vzdálený správce front.

MQRC_UNKNOWN_XMIT_Q
(2196, X'894 ') Neznámá přenosová fronta.

CHYBA MQRC_UOW_ENLISTMENT_ERROR
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

MQRC_UOW_MIX_NOT_SUPPORTED
(2355, X' 933 ') Směs volání jednotek práce není podporována.

MQRC_UOW_NOT_AVAILABLE
(2255, X'8CF') Unit of work not available for the queue manager to use.

CHYBA MQRC_WIH_ERROR
(2333, X'91D') Struktura MQWIH není platná.

MQRC_WRONG_CF_LEVEL
(2366, X'93E') Struktura prostředku Coupling Facility má nesprávnou úroveň.

VERZE MQRC_WRONG_MD_VERSION
(2257, X'8D1') Chybná verze dodaných MQMD.

CHYBA MQRC_XMIT_Q_TYPE_ERROR
(2091, X'82B') Přenosová fronta není lokální.

CHYBA MQRC_XMIT_Q_USAGE_ERROR
(2092, X'82C') Přenosová fronta s chybným použitím.

CHYBA MQRC_XQHL_ERROR
(2260, X'8D4') Struktura záhlaví přenosové fronty není platná.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Volání MQPUT i volání MQPUT1 lze použít k umístění zpráv do fronty. Volání, které má být použito, závisí na okolnostech:
   - Použijte volání MQPUT k umístění více zpráv ve *stejné* frontě.

     Bylo zadáno volání MQOPEN s určením volby MQOO_OUTPUT, za nímž následuje jeden nebo více požadavků MQPUT pro přidání zpráv do fronty. Nakonec je fronta uzavřena s voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
   - Pomocí volání MQPUT1 vložte do fronty pouze *jednu* zprávu.

     Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která musí být vydána.
2. Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny určité podmínky. Avšak ve většině prostředí volání MQPUT1 nesplňuje tyto podmínky, a proto nezachovává pořadí zpráv. Místo toho v těchto prostředích musí být místo volání MQPUT použito volání MQPUT. Podrobnosti naleznete v tématu Poznámky k použití MQPUT .
3. Volání MQPUT1 lze použít k umístění zpráv do distribučních seznamů. Obecné informace o tomto tématu najdete 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 se používají následující rozdíly:

   a. Pokud aplikace poskytuje záznamy odezvy MQRR, musí být poskytnuty s použitím struktury MQOD; nelze je poskytnout pomocí struktury MQPMO.

   b. Kód příčiny MQRC_OPEN_FAILED nebyl nikdy vrácen položkou MQPUT1 v záznamech odezvy; pokud se nepodaří otevřít frontu, obsahuje záznam odpovědi pro tuto frontu kód příčiny, který je výsledkem operace otevření.

      Pokud operace otevření fronty uspěje s kódem dokončení MQCC_WARNING, kód dokončení a kód příčiny v záznamu odpovědi pro danou frontu se nahradí kódem dokončení a kódem příčiny, které jsou výsledkem operace put.

      Stejně jako v případě volání MQOPEN a MQPUT správce front nastaví záznamy odpovědí (je-li k dispozici) pouze v případě, že výsledek volání není stejný pro všechny fronty v rozdělovníku; to je indikováno dokončením volání s kódem příčiny MQRC_MULTIPLE_REASONS.
4. Je-li volání MQPUT1 použito k vložení zprávy do fronty klastru, volání se bude chovat, jako by byl v rámci volání MQOPEN zadán parametr MQOO_BIND_NOT_FIXED.
5. Je-li zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátku dat zprávy aplikace, provede správce front určité kontroly struktury záhlaví, aby ověřil, zda jsou platné. Další informace o tomto tématu naleznete v poznámkách k použití volání MQPUT.

6. Pokud se objeví více než jedna z varovných situací (viz parametr **CompCode**), vrácený kód příčiny je první v následujícím seznamu, který se používá:
   a. MQRC_MULTIPLE_PŘÍČINY
   b. ZPRÁVA MQRC_INCOMPLETE_MSG
   c. SKUPINA MQRC_INCOMPLETE_GROUP
   d. Funkce MQRC_PRIORITY_EXCEES_MAXIMUM nebo MQRC_UNKNOWN_REPORT_OPTION
7. V případě programovacího jazyka Visual Basic platí tyto body:
   - Je-li velikost parametru **Buffer** menší než délka zadaná 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, není typu `String`, použijte příkazVolání MQPUT1Any na místo 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 produkt `Buffer` nelze zkontrolovat, aby se zajistilo, že velikost bude mít velikost alespoň `BufferLength` bajtů.
8. Je-li volání MQPUT1 vydáno s MQPMO_SYNCPOINT, změní se výchozí chování, takže je operace vložení dokončena asynchronně. To může způsobit změnu chování některých aplikací, které závisí na určitých polích ve strukturách MQOD a MQMD, které jsou vraceny, ale které nyní obsahují nedefinované hodnoty. Aplikace může určit MQPMO_SYNC_RESPONSE, aby se zajistilo, že je operace vložení provedena 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 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-Nastavit atributy objektu

Použijte volání MQSET pro změnu atributů objektu reprezentovaného popisovačem. Objekt musí být fronta.

Syntaxe

MQSET (*Hconn*, *Hobj*, *SelectorCount*, *Selektory*, *IntAttrCount*, *IntAttrs*, *CharAttrLength*, *CharAttrs*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota Hconn byla vrácena předchozím voláním MQCONN nebo MQCONNX.

z/OS V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

HOBJ
Typ: MQHOTBJ-vstup

Tento manipulátor představuje objekt fronty s atributy, které mají být nastaveny. Manipulátor byl vrácen předchozím voláním MQOPEN, které je určeno volbou 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: MQLONGxSelectorCount-vstup

Jedná se o pole selektorů atributů produktu **`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 hodnoty MQIA_ * a MQCA_ *; viz níže.

Selektory mohou být zadány v libovolném pořadí. Hodnoty atributů odpovídající celočíselným selektorům atributů (selektory MQIAK_ *) musí být určeny v produktu `IntAttrs` ve stejném pořadí, v jakém se tyto selektory vyskytují v produktu `Selectors`. Hodnoty atributu, které odpovídají selektorům atributu znaku (selektory MQCA_ *), musí být specifikovány v produktu `CharAttrs` ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory MQIA_ * mohou být prokládané selektory MQCA_ *; důležité je pouze relativní pořadí v rámci každého typu.

Stejný selektor můžete zadat více než jednou, pokud tak učiníte, poslední hodnota zadaná pro konkrétní selektor je ta, která se projeví.

Poznámka:

1. Selektory atributů celého čísla a znakového atributu jsou přiděleny ve dvou různých rozsazích; selektory MQIA_ * jsou umístěny v rozsahu MQIA_FIRST až MQIA_LAST a selektorů MQCA_ * v rozsahu MQCA_FIRST až MQCA_LAST.

   Pro každý rozsah definují konstanty MQIA_LAST_USED a MQCA_LAST_USED nejvyšší hodnotu, kterou správce front přijme.
2. Pokud se všechny selektory MQIA_ * selektují jako první, lze použít stejná čísla prvků k adresování příslušných prvků v polích `Selectors` a `IntAttrs` .
3. Pokud je argument **`SelectorCount`** nulový, `Selectors` není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům zapsaným v C nebo System/390 assembler s hodnotou null.

Atributy, které lze nastavit, jsou vypsány v následující tabulce. Pomocí tohoto volání nelze nastavit žádné další atributy. Pro selektory atributů MQCA_ * je konstanta, která definuje délku řetězce požadovaného parametrem `CharAttrs` v závorce, zadána v bajtech.

Tabulka 555. Selektory atributů MQSET pro fronty

Selektor	Popis	Poznámka
DATA MQCA_TRIGGER_DATA	Data spouštěče (MQ_TRIGGER_DATA_LENGTH).	
MQIA_DICT_LISTS	Podpora distribučního seznamu.	1
MQIA_INHIBIT_GET	Zda jsou povoleny operace get.	
MQIA_INHIBIT_PUT	Zda jsou povoleny operace vložení.	
MQIA_TRIGGER_CONTROL	Řízení spouštěče.	
HLOUBKA MQIA_TRIGGERU_T	Hloubka spouątěče.	
MQIA_TRIGGER_MSG_PRIORITY	Priorita zprávy prahové hodnoty pro spouątěče.	
TYP_SPOUŠTĚČE_MQIA_TYPE	Typ spouątěče.	

Poznámka:

1. Podporováno pouze na následujících platformách:
   - AIX
   - IBM i
   - Linux
   - Windows

   a pro IBM MQ MQI clients připojené k těmto systémům.

IntAttrCount
Typ: MQLONG-vstup

Jedná se o počet prvků v poli `IntAttrs` a musí být alespoň počtem selektorů MQIA_ * v parametru **`Selectors`** . Nula je platná hodnota, pokud neexistují žádné.

IntAttrs
Typ: MQLONGxIntAttrCount -vstup

Toto je pole celočíselných hodnot atributů `IntAttrCount` . Tyto hodnoty atributů musí být ve stejném pořadí jako selektory MQIA_ * v poli `Selectors` .

Pokud je argument **`IntAttrCount`** nebo **`SelectorCount`** nula, `IntAttrs` není na ně odkazováno; v tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

Délka CharAttr
Typ: MQLONG-vstup

Toto je délka v bajtech parametru **`CharAttrs`** a musí být alespoň součtem délek znakových atributů uvedených v poli `Selectors` . Nula je platná hodnota, pokud v `Selectors`nejsou žádné selektory MQCA_ *.

CharAttrs
Typ: MQCHAR x CharAttrDélka-vstup

Jedná se o vyrovnávací paměť obsahující hodnoty atributu znaku zřetězené dohromady. Délka vyrovnávací paměti je dána parametrem **`CharAttrLength`** .

Atributy znaků musí být zadány ve stejném pořadí jako selektory MQCA_ * v poli `Selectors` . Délka každého atributu znaku je pevná (viz Selektory). Pokud hodnota, která má být pro atribut nastavena, obsahuje méně nemezerových znaků, než je definovaná délka atributu, zarovnat hodnotu v `CharAttrs` vpravo s mezerami, aby se hodnota atributu shodovala s definovanou délkou atributu.

Pokud je argument **CharAttrLength** nebo **SelectorCount** nula, `CharAttrs` není na ně odkazováno; v tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující `CompCode`.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li `CompCode` MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

MQRC_CF_NOT_AVAILABLE
(2345, X' 929 ') Prostředek Coupling Facility není k dispozici.

MQRC_CF_STRU_FAILED
(2373, X' 945 ') Struktura prostředku Coupling Facility selhala.

MQRC_CF_STRUC_IN_USE
(2346, X'92A') Struktura prostředku Coupling Facility se používá.

MQRC_CF_STRU_LIST_HDR_IN_USE
(2347, X'92B') Hlavička prostředku Coupling-facility-záhlaví se používá.

MQRC_CHAR_ATTR_LENGTH_ERROR
(2006, X'7D6') Délka znakových atributů není platná.

CHYBA MQRC_CHAR_ATTRS_ERROR
(2007, X'7D7') Řetězec atributů znaků není platný.

MQRC_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

AUTORIZOVANÝ MQRC_CONNECTION_NOT_AUTHORIZED
(2217, X'8A9') Chybí autorizace pro připojení.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

MQRC_DB2_NOT_AVAILABLE
(2342, X' 926 ') Subsystém Db2 není k dispozici.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HOBJ_ERROR
(2019, X'7E3') Popisovač objektu není platný.

CHYBA MQRC_INHIBIT_VALUE_ERROR
(2020, X'7E4') Hodnota pro atribut inhibit-get nebo inhibit-put queue není platná.

CHYBA_MQRC_INT_ATTR_COUNT_ERROR
(2021, X'7E5') Počet celočíselných atributů není platný.

CHYBA POLE MQRC_INT_ATTRS_ARRAY_ERROR
(2023, X'7E7') Pole celočíselné atributy není platné.

MQRC_NOT_OPEN_FOR_SET
(2040, X'7F8') Fronta není otevřena pro nastavení.

MQRC_OBJECT_CHANGED
(2041, X'7F9') Definice objektu byla od otevření změněna.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

CHYBA OBJEKTU 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 není platný nebo je neznámý.

MQRC_Q_MGR_NOT_AVAILABLE
(2059, X'80B') Správce front není k dispozici pro připojení.

MQRC_Q_MGR_STOPPING
(2162, X'872 ') Správce front se vypíná.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_SELECTOR_COUNT_ERROR
(2065, X'811 ') Počet selektorů není platný.

CHYBA MQRC_SELECTOR_ERROR
(2067, X'813 ') Selektor atributu není platný.

MQRC_SELECTOR_LIMIT_EXCEEDED
(2066, X'812 ') Počet selektorů je příliš velký.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_SUPPRESSED_BY_EXIT
(2109, X'83D') Volání potlačeno ukončovacím programem.

CHYBA ŘÍZENÍ MQRC_TRIGGER_CONTROL_ERROR
(2075, X'81B') Hodnota pro atribut řízení spouštěče není platná.

CHYBA 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.

CHYBA 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 najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Při použití tohoto volání může aplikace určit pole celočíselných atributů nebo kolekci řetězců znakových atributů, nebo obojí. Pokud se nevyskytnou žádné chyby, uvedené atributy jsou všechny nastaveny současně. Pokud dojde k chybě (například, pokud je selektor neplatný nebo je proveden pokus o nastavení atributu na hodnotu, která není platná), volání selže a nejsou nastaveny žádné atributy.
2. Hodnoty atributů lze určit pomocí volání MQINQ; podrobnosti naleznete v části “MQINQ-Dotaz na atributy objektu” na stránce 699 .

   Poznámka: Ne všechny atributy s hodnotami, které mohou být dotazovány pomocí volání MQINQ, mohou mít své hodnoty změněny pomocí volání MQSET. Pomocí tohoto volání lze například nastavit žádné atributy objektu procesu nebo správce front.
3. Změny atributů jsou zachovány po restartu správce front (jiné než změny dočasných dynamických front, které nepřečkají restarty správce front).
4. Atributy modelové fronty nelze změnit pomocí volání MQSET. Pokud však otevřete modelovou frontu pomocí volání MQOPEN s volbou MQOO_SET, můžete použít volání MQSET k nastavení atributů dynamické lokální fronty vytvořené voláním MQOPEN.
5. Je-li nastavovaný objekt fronta klastru, musí existovat lokální instance fronty klastru, aby byla otevřená úspěšná.

Další informace o atributech objektů najdete v tématech:

- “Atributy pro fronty” na stránce 827
- “Atributy pro seznamy názvů” na stránce 860
- “Atributy pro definice procesu” na stránce 862
- “Atributy správce front” na stránce 791

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 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

Použijte volání MQSETMP k nastavení nebo úpravě vlastnosti obslužné rutiny zprávy.

Syntaxe

MQSETMP (*Hconn*, *Hmsg*, *SetPropOpts*, *Name*, *PropDesc*, *Type*, *ValueLength*, *Value*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** . Pokud byl popisovač zprávy vytvořen pomocí MQHC_UNASSOCIATED_HCONN, musí být ustanoveno platné připojení na podprocesu, který nastavuje vlastnost obslužné rutiny zprávy, jinak se volání nezdaří 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.

SetPropOpts
Typ: MQSMPO-vstup

Řídí, jak jsou nastaveny vlastnosti zpráv.

Tato struktura umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zpráv. Struktura je vstupním parametrem 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 naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

PropDesc
Typ: MQPD-input/output

Tato struktura se používá k definování atributů vlastnosti, včetně:

- co se stane, pokud vlastnost není podporována
- jaký kontext zprávy vlastnost patří
- Jaké zprávy je vlastnost kopírována do průběhu toku

Další informace o této struktuře viz MQPD .

Typ
Typ: MQLONG-vstup

Datový typ nastavované vlastnosti. Může se jednat o jednu z následujících možností:

LOGICKÁ HODNOTA MQTYPE_BOOLEAN
Logická hodnota. *ValueLength* musí být 4.

ŘETĚZEC MQTYPE_BYTE_STRING
Řetězec bajtů. Hodnota *ValueLength* musí být nula nebo větší.

MQTYPE_INT8
8bitové podepsané celé číslo. *ValueLength* musí být 1.

MQTYPE_INT16
16bitové podepsané celé číslo. *ValueLength* musí být 2.

MQTYPE_INT32
32bitové celé číslo se znaménkem. *ValueLength* musí být 4.

MQTYPE_INT64
64bitové podepsané celé číslo. *ValueLength* musí být 8.

MQTYPE_FLOAT32
32-bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 4.

Poznámka: Tento typ není podporován u aplikací, které používají produkt IBM COBOL for z/OS.

MQTYPE_FLOAT64
64-bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 8.

Poznámka: Tento typ není podporován u aplikací, které používají produkt IBM COBOL for z/OS.

ŘETĚZEC MQTYPE_STRING
Znakový řetězec. Hodnota *ValueLength* musí být nula nebo větší nebo se 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 *hodnota* v bajtech. Nula je platná pouze pro hodnoty null, nebo pro řetězce nebo bajtové řetězce. Nula označuje, že vlastnost existuje, ale že tato hodnota neobsahuje žádné znaky ani bajty.

Hodnota musí být větší než nula nebo rovna nule nebo následující speciální hodnota, pokud má parametr *Type* nastaven typ 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 nastaven také parametr MQTYPE_STRING.

Pozn.: Znak null použitý k ukončení řetězce, pokud je hodnota MQVL_NULL_TERMINATED nastavena na 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 tento parametr deklarován jako ukazatel-to-void; adresa libovolného typu dat může být zadána jako parametr.

Je-li *ValueLength* nula, *Hodnota* není odkazována. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Má-li parametr *CompCode* hodnotu MQCC_WARNING:

CHYBA MQRC_RFH_FORMAT_ERROR
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nebylo možné analyzovat.

Je-li položka *CompCode* MQCC_FAILED:

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'089C') Adaptér není k dispozici.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

NESROVNALOST MQRC_ASID_
(2157, X'86D') Primární a domovské ASID se liší.

CHYBA MQRC_BUFFER_ERROR
(2004, X'07D4') Hodnota parametru hodnoty není platná.

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'07D5') Hodnota parametru délky hodnoty není platná.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

CHYBA MQRC_HMSG_ERROR
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

MQRC_MSG_HANDLE_IN_USE
(2499, X'09C3') Popisovač zprávy je již používán.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_PD_ERROR
(2482, X'09B2') Struktura deskriptoru vlastností není platná.

CHYBA MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

CHYBA MQRC_PROPERTY_TYPE_ERROR
(2473, X'09A9') Neplatný typ dat vlastnosti.

CHYBA MQRC_PROP_NUMBER_FORMAT_ERROR
(2472, X'09A8') Chyba formátu čísla zjištěna v datech hodnoty.

CHYBA MQRC_SMPO_ERROR
(2463, X'099F') Nastavení struktury voleb vlastností zprávy není platné.

CHYBA MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

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-Načíst informace o stavu

Použijte volání MQSTAT k získání informací o stavu. Typ vrácených informací o stavu je určen hodnotou typu uvedenou ve volání.

Syntaxe

MQSTAT (*Hconn*, *Type*, *Stat*, *Compcode*, *Reason*)

Parametry

Hconn

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

Typ

Typ: MQLONG-vstup

Typ požadovaných informací o stavu. Platné hodnoty jsou:

CHYBA MQSTAT_TYPE_ASYNC_ERROR
Vrátit informace o předchozích asynchronních operacích vložení.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Vrátit informace o opětovném připojení. Pokud se připojení znovu připojuje nebo selhalo opětovné připojení, informace popisují selhání, které způsobilo, že připojení začalo znovu navázat spojení.

Tato hodnota je platná pouze pro připojení klienta. U jiných typů připojení volání selže s kódem příčiny **MQRC_ENVIRONMENT_ERROR**

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Vrátí informace o předchozím selhání, které souvisí se znovu navázat spojení. Pokud se připojení nezdařilo znovu připojit, v informacích je popsáno selhání, které způsobilo selhání opětovného připojení.

Tato hodnota je platná pouze pro připojení klienta. U jiných typů připojení se volání nezdaří s kódem příčiny **MQRC_ENVIRONMENT_ERROR**.

Statistika
Typ: MQSTS-input/output

Struktura informací o stavu. Podrobnosti viz "MQSTS-Struktura vykazování stavu" na stránce 586.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_API_EXIT_ERROR
(2374, X' 946 ')-ukončení rozhraní API se nezdařilo

CHYBA MQRC_API_EXIT_LOAD_ERROR
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

MQRC_CALL_IN_PROGRESS
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

MQRC_Q_MGR_STOPPING
(2162,X'872'-Zastavení správce front

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_STAT_TYPE_ERROR
(2430, X'97E' Chyba s typem MQSTAT

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

CHYBA MQRC_STS_ERROR
(2426, X'97A') Chyba struktury MQSTS

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Došlo k neočekávané chybě.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

1. Volání MQSTAT uvádějící typ MQSTAT_TYPE_ASYNC_ERROR vrací informace o předchozích asynchronních operacích MQPUT a MQPUT1 . Struktura MQSTS předaná při návratu z volání MQSTAT obsahuje první zaznamenané asynchronní varování nebo informace o chybě pro toto připojení. Pokud další chyby nebo varování následují za prvními, tyto hodnoty obvykle neupravují. Pokud však dojde k chybě s kódem dokončení MQCC_WARNING, je místo toho vráceno následné selhání s kódem dokončení MQCC_FAILED .
2. Pokud nedošlo k žádným chybám od té doby, kdy bylo připojení ustanoveno, nebo od posledního volání k MQSTAT , pak se CompCode z MQCC_OK a z důvodu MQRC_NONE vrátí ve struktuře MQSTS .
3. Počty počtu asynchronních volání, která byla zpracována pod manipulátorem připojení, jsou vraceny třemi poli čítačů; PutSuccessCount, PutWarningCount a PutFailureCount. Tyto čítače jsou zvyšovány správcem front při každém zpracování asynchronní operace, která má varování nebo selže (všimněte si, že pro účely účtování se na distribuční seznam místo jednou na seznam rozdělení počítá na distribuční seznam jednou). Počitadlo není zvyšováno nad maximální kladnou hodnotu AMQ_LONG_MAX.
4. Úspěšné volání příkazu MQSTAT má za následek zrušení všech předchozích chybových informací nebo počtů chyb.
5. Chování parametru MQSTAT závisí na hodnotě parametru **MQSTAT Type** , který jste zadali.
6. **CHYBA MQSTAT_TYPE_ASYNC_ERROR**
    a. Volání MQSTAT uvádějící typ MQSTAT_TYPE_ASYNC_ERROR vrací informace o předchozích asynchronních operacích MQPUT a MQPUT1 . Struktura MQSTS předaná při návratu z volání MQSTAT obsahuje první zaznamenané asynchronní varování nebo informace o chybě pro toto připojení. Pokud další chyby nebo varování následují za prvními, tyto hodnoty obvykle neupravují. Pokud však dojde k chybě s kódem dokončení MQCC_WARNING, je místo toho vráceno následné selhání s kódem dokončení MQCC_FAILED .
    b. Pokud nedošlo k žádným chybám od té doby, kdy bylo připojení ustanoveno, nebo od posledního volání k MQSTAT , pak se CompCode z MQCC_OK a z důvodu MQRC_NONE vrátí ve struktuře MQSTS .
    c. Počty počtu asynchronních volání, která byla zpracována pod manipulátorem připojení, jsou vraceny třemi poli čítačů; PutSuccessCount, PutWarningCount a PutFailureCount. Tyto čítače jsou zvyšovány správcem front při každém zpracování asynchronní operace, která má varování nebo selže (všimněte si, že pro účely účtování se na distribuční seznam místo jednou na seznam rozdělení počítá na distribuční seznam jednou). Počitadlo není zvyšováno nad maximální kladnou hodnotu AMQ_LONG_MAX.
    d. Úspěšné volání příkazu MQSTAT má za následek zrušení všech předchozích chybových informací nebo počtů chyb.

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

Předpokládejme, že voláte MQSTAT s parametrem Type nastaveným na MQSTAT_TYPE_RECONNECTION uvnitř obslužné rutiny událostí během opětovného připojení. Zvažte tyto příklady.

Klient se pokouší znovu připojit, nebo se nezdařilo znovu navázat spojení.
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 prázdný.

Klient úspěšně dokončil opětovné připojení nebo se nikdy neodpojil.
CompCode ve struktuře MQSTS je MQCC_OK a Reason je MQRC_NONE

Následná volání do MQSTAT vracejí stejné výsledky.

CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR

Předpokládejme, že voláte produkt MQSTAT s parametrem Type nastaveným na hodnotu MQSTAT_TYPE_RECONNECTION_ERROR v reakci na příjem volání MQRC_RECONNECT_FAILED na volání MQI. Zvažte tyto příklady.

Došlo k selhání autorizace při opětovném otevření fronty během opětovného připojení k jinému správci front.
CompCode ve struktuře MQSTS je MQCC_FAILED a Reason je důvodem, proč selhalo opětovné připojení, jako například MQRC_NOT_AUTHORIZED. ObjectType je typ objektu, který způsobil problém, jako například MQOT_QUEUE, ObjectName je název fronty a ObjectQMgrName název správce front, který frontu vlastní.

Během opětovného připojení došlo k chybě soketového připojení.
CompCode ve struktuře MQSTS je MQCC_FAILED a Reason je důvodem, proč selhalo opětovné připojení, jako například MQRC_HOST_NOT_AVAILABLE. ObjectType je MQOT_Q_MGR, ObjectName je název správce front a ObjectQMgrName je prázdný.

Následná volání do MQSTAT vracejí 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 assembler

```
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

Použijte volání MQSUB k registraci odběru aplikací pro konkrétní téma.

Syntaxe

MQSUB (*Hconn*, *SubDesc*, *Hobj*, *Hsub*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

SubDesc
Typ: MQSD-vstup a výstup

Jedná se o strukturu, která identifikuje objekt, který je registrován aplikací. Další informace viz “MQSD-deskriptor odběru” na stránce 562.

HOBJ
Typ: MQHOBJ-vstupní/výstupní

Tento popisovač představuje přístup, který byl vytvořen za účelem získání zpráv odeslaných do tohoto odběru. Tyto zprávy mohou být buď uloženy ve specifické frontě, nebo správce front může spravovat jejich úložiště bez použití určité fronty.

Chcete-li použít specifickou frontu, musíte ji přidružit k odběru, když je vytvořen odběr. To lze provést dvěma způsoby:

- Použijte příkaz DEFINE SUB MQSC a zadejte tento příkaz s názvem objektu fronty.
- Poskytnutím této obslužné rutiny při volání MQSUB s MQSO_CREATE

  Je-li tento popisovač zadán jako vstupní parametr ve volání, musí se jednat o platný popisovač objektu vrácený z předchozího volání MQOPEN fronty pomocí alespoň jedné z následujících voleb:

  – MQO_INPUT_*
  – MQOOK_BROWSE
  – MQOO_OUTPUT (je-li fronta vzdálenou frontou)

  Pokud se nejedná o tento případ, volání selže s chybou MQRC_HOBJ_ERROR. Nemůže to být popisovač objektu pro frontu aliasů, která se interpretuje jako objekt tématu. Je-li tomu tak, volání selže s chybou MQRC_HODBJ_ERROR.

Pokud má správce front spravovat ukládání zpráv odeslaných do tohoto odběru, mělo by být toto nastavení nastaveno při vytváření odběru pomocí volby MQSO_MANAGED. Správce front tento popisovač vrátí jako výstupní parametr ve volání. Vrácený popisovač je známý jako spravovaný popisovač. Je-li zadána hodnota MQHO_NONE, ale není zadána hodnota MQSO_MANAGED, volání selže s chybou MQRC_HODBJ_ERROR.

Pokud je spravovaný manipulátor vrácen správcem front, můžete jej použít při volání MQGET nebo MQCB s volbami procházení MQINQ nebo MQCLOSE nebo bez voleb procházení. Nelze ji použít na MQPUT, MQSUB, MQSET; pokus o provedení tak selže s chybou MQRC_NOT_OPEN_FOR_OUTPUT, MQRC_HODBJ_ERROR nebo MQRC_NOT_OPEN_FOR_SET.

Je-li tento odběr obnoven pomocí volby MQSO_RESUME ve struktuře MQSD, lze obslužnou rutinu vrátit do aplikace v tomto parametru nastavením hodnoty MQSO_MANAGED na hodnotu MQHO_NONE. Můžete to provést, zda odběr používá spravovaný popisovač, nebo ne a může být užitečný k poskytnutí odběrů vytvořených pomocí příkazu DEFINE SUB s manipulátorem na frontu odběru definovanou v daném příkazu. V případě, kdy je obnovován administrativně vytvořený odběr, se otevře fronta s MQOO_INPUT_AS_Q_DEF a MQOO_BROWSE. Potřebujete-li zadat jiné volby, musí aplikace explicitně otevřít frontu odběru a poskytnout obslužnou rutinu objektu ve volání. Pokud se vyskytne problém při otevírání fronty, volání selže s hodnotou MQRC_INVALID_DESTINATION. Je-li zadán parametr *Hobj* , musí být ekvivalentní příkazu *Hobj* v rámci původního volání MQSUB. To znamená, že pokud se poskytuje popisovač objektu vrácený z volání MQOPEN, musí být daný popisovač ke stejné frontě, jak byla dříve použita. Pokud se nejedná o stejnou frontu, volání selže s chybou MQRC_HODBJ_ERROR.

Pokud je tento odběr změněn pomocí volby MQSO_ALTER ve struktuře MQSD, může být poskytnut jiný produkt *Hobj* . Všechny publikace, které byly doručeny do fronty a byly dříve identifikovány prostřednictvím tohoto parametru, zůstanou v této frontě a za předpokladu, že parametr **Hobj** nyní představuje jinou frontu, je odpovědností aplikace načítat tyto zprávy.

Tabulka 556. Použití objektu hobj s různými volbami odběru

Volby	***Hobj***	**Popis**
MQSO_CREATE + MQSO_MANAGED	Ignorováno na vstupu	Vytvoří odběr zpráv spravovaných správcem front s úložištěm zpráv
VYTVOŘENÉ MQSO_CREATE	Platný popisovač objektu	Vytvoří odběr obsahující specifickou frontu jako místo určení pro zprávy.
MQSO_RESUME	MQHO_NONE	Obnoví dříve vytvořený odběr bez ohledu na to, zda byl spravován či nikoli, a že správce front vrátil popisovač objektu pro použití aplikací.

Tabulka 556. Použití objektu hobj s různými volbami odběru (pokračování)		
Volby	***Hobj***	**Popis**
MQSO_RESUME	Platný, vyhovující, popisovač objektu	Obnoví dříve vytvořený odběr, který používá specifickou frontu jako místo určení pro zprávy a používá popisovač objektu se specifickými volbami otevření.
MQSO_ALTER + MQSO_MANAGED	MQHO_NONE	Pozmění existující odběr, který byl již dříve používán specifickou frontou, takže se nyní jedná o spravovaný odběr. Třídu cíle (spravovaného či nikoli) nelze změnit.
MQSO_ALTER	Platný popisovač objektu	Pozmění existující odběr bez ohledu na to, zda byl spravován či nikoli, takže nyní používá specifickou frontu. Není-li použita volba MQSO_MANAGED, lze danou frontu změnit, ale třídu cíle (spravovanou či nikoli) nelze změnit.

Zda byla poskytnuta nebo vrácena, musí být v následujících voláních MQGET nebo MQCB zadána hodnota *Hobj* , která má přijímat zprávy publikování odeslané do tohoto odběru.

Popisovač *Hobj* již není platný, když je na něm vydán volání MQCLOSE, nebo když se jednotka zpracování, která definuje rozsah popisovače, ukončí (až se aplikace odpojí). Rozsah vráceného manipulátorů objektu je stejný jako rozsah manipulátoru připojení zadaného při volání. Informace o oboru popisovači viz Hconn (MQHCONN)-output . MQCLOSE obslužné rutiny *Hobj* nemá vliv na popisovač *Hsub* .

HSub
Typ: MQHOTBJ-výstup

Tento popisovač představuje odběr, který byl proveden. Může být použit pro další dvě operace:

- Lze ji použít v následujícím volání MQSUBRQ k požadavku na odeslání publikování, pokud byla při vytváření odběru použita volba MQSO_PUBLICATIONS_ON_REQUEST.
- Může být použit v následném volání MQCLOSE k odebrání odběru, který byl proveden. Popisovač *Hsub* přestane být platný, když je vydáno volání MQCLOSE, nebo když se jednotka zpracování, která definuje rozsah popisovače, ukončí. Rozsah vráceného manipulátorů objektu je stejný jako rozsah manipulátoru připojení zadaného při volání. MQCLOSE obslužné rutiny *Hsub* nemá vliv na popisovač *Hobj* .

Tento manipulátor nelze předat do volání MQGET nebo MQCB. Je třeba použít argument **Hobj** . Tento popisovač nelze použít u jiných volání IBM MQ než MQCLOSE nebo MQSUBRQ. Předání tohoto popisovače do jiných volání příkazu IBM MQ způsobí chybu MQRC_HODBJ_ERROR.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení)

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Příčina

Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK, kód příčiny vypadá takto:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED, kód příčiny je jeden z následujících:

CHYBA MQRC_CLUSTER_RESOLVUTION_ERROR
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

MQRC_DURABILITY_NOT_ALLOWED
2436 (X'0984 ') Volání MQSUB pomocí volby MQSO_DURABLE se nezdařilo.

PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

CHYBA MQRC_HOBJ_ERROR
2019 (X'07E3') Objekt Hobj popisovače objektu 2019 není platný.

NESROVNALOST MQRC_IDENTITY_
2434 (X'0982 ') Název odběru odpovídá existujícímu odběru.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
2035 (X'07F3') Uživatel není autorizován k provedení operace.

MQRC_NO_SUBSCRIPTION
2428 (X'097C') Identifikovaný název odběru neexistuje.

CHYBA MQRC_OBJECT_STRING_ERROR
2441 (X'0989 ') Pole Objectstring není platné.

CHYBA MQRC_OPTIONS_ERROR
2046 (X'07FE') Parametr nebo pole voleb obsahuje volby, které nejsou platné, nebo kombinace voleb, které nejsou platné.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
2161 (X'0871 ') Správce front je uváděn do klidového stavu.

POŽ. Q_MGR_QM_Q_MGR_QM_Q_MGR_
2555 (X'09FB' X) Je požadována volba MQCNO_RECONNECT_Q_MGR.

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') Zachovaná publikování, která existují pro řetězec odebíraného tématu, nelze načíst.

MQRC_RETAINED_NOT_DELIVERED, DORUČENO
2526 (X'09DE') Zachované publikace, které existují pro odebíraný řetězec témat, nelze doručit do cílové fronty odběru a nelze ji doručit do fronty nedoručených zpráv.

CHYBA MQRC_SD_ERROR
2424 (X'0978 ') Deskriptor odběru (MQSD) není platný.

MQRC_SELECTION_NOT_AVAILABLE
2551 (X'09F7') Řetězec výběru se neřídí syntaxí selektoru IBM MQ a nebyl k dispozici žádný rozšířený poskytovatel výběru zpráv.

CHYBA MQRC_SELECTION_STRING_ERROR
2519 (X'09D7') Řetězec výběru musí být zadán podle popisu v dokumentaci struktury MQCHARV.

CHYBA MQRC_SELECTOR_SYNTAX_ERROR
2459 (X'099B') Bylo vydáno volání MQOPEN, MQPUT1nebo MQSUB, ale byl zadán výběrový řetězec, který obsahoval chybu syntaxe.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') Datové pole SubUsernení platné.

CHYBA MQRC_SUB_NAME_ERROR
2440 (X'0988 ') Pole SubName není platné.

MQRC_SUB_ALREADY_EXISTS
2432 (X'0980 ') Odběr již existuje.

MQRC_SUB_USER_DATA_ERROR
2431 (X'097F') Datové pole SubUsernení platné.

CHYBA MQRC_TOPIC_STRING_ERROR
2425 (X'0979 ') Řetězec tématu není platný.

MQRC_UNKNOWN_OBJECT_NAME
2085 (X'0825 ') Objekt identifikovaný v poli ObjectName MQSD nemůže být nalezen.

MQRC_SUB_JOIN_NOT_ALTERABLE
29440 (X'7300 ') Režim sdílení odběru není kompatibilní s existujícím odběrem. Tato chyba mohla být vrácena při pokusu o obnovení sdíleného odběru JMS 2.0 v jiné aplikaci než JMS.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

- Odběr se provádí na téma s názvem buď pomocí krátkého názvu předdefinovaného objektu tématu, úplného názvu řetězce tématu, nebo se vytvoří zřetězením dvou částí. Viz popis *ObjectName* a *ObjectString* v “MQSD-deskriptor odběru” na stránce 562.
- Správce front provádí kontroly zabezpečení při vydání volání MQSUB, aby ověřil, zda má identifikátor uživatele, pod kterým je spuštěna aplikace, odpovídající úroveň oprávnění, než je povolen přístup. Příslušný objekt tématu se nachází v hierarchii témat a na tomto objektu tématu je provedena kontrola oprávnění, aby bylo zajištěno, že je nastaveno oprávnění k odběru. Není-li použita volba MQSO_MANAGED, je v cílové frontě provedena kontrola oprávnění, aby bylo zajištěno, že je nastaveno oprávnění pro výstup. Je-li použita volba MQSO_MANAGED, neprovádí se žádná kontrola oprávnění ve spravované frontě pro výstup nebo přístup s možností dotazu.
- Pokud jako vstup nezadáte žádný objekt Hobj, volání MQSUB přidělí dva popisovače, popisovač objektu (Hobj) a popisovač odběru (Hsub).
- Objekt Hobj vrácený při volání MQSUB při použití volby MQSO_MANAGED může být dotazován, aby bylo možné zjistit atributy, jako je například prahová hodnota vrácení a nadměrné vrácení zpráv vrácení zpět. Můžete také zadat dotaz na název spravované fronty, ale nesmíte se pokusit o přímé otevření této fronty.
- Odběry mohou být seskupeny tak, aby bylo možné doručit pouze jednu publikaci do skupiny odběrů, a to dokonce i tam, kde se více než jedna ze skupin shoduje s publikací. Odběry jsou seskupeny pomocí volby MQSO_GROUP_SUB a v pořadí skupinového odběru, které musí být
  - pomocí stejné pojmenované fronty (která nepoužívá volbu MQSO_MANAGED) na stejném správci front-reprezentovaný parametrem Hobj v volání MQSUB
  - sdílí stejné ID SubCorrel
  - být stejné SubLevel

  Tyto atributy definují sadu odběrů, které jsou považovány za odběry ve skupině, a také atributy, které nelze změnit, je-li seskupen odběr. Změna SubLevel výsledků ve funkci MQRC_SUBLEVEL_NOT_ALTERABLE a změna kteréhokoli z ostatních změn (které lze změnit, pokud není 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, přečtěte si krok DEFINE SUB v tématu Kontrola, zda byly dokončeny asynchronní příkazy pro distribuované sítě.
- Pole v MQSD jsou vyplněna při návratu z volání MQSUB, které používá volbu MQSO_RESUME. Vrácené MQSD lze předat přímo do volání MQSUB, které používá volbu MQSO_ALTER s libovolněmi změnami, které je třeba provést u odběru použitého pro MQSD. Některá pole mají speciální posouzení, jak je uvedeno v tabulce.

Tabulka 557. Speciální pokyny pro pole v MQSD	
Název pole v MQSD	**Speciální aspekty.**
Přístup nebo volby vytvoření	Některé z voleb lze vynulovat při návratu z volání MQSUB. Pokud znovu použijete MQSD v rámci volání MQSUB, musí být volba, kterou požadujete, explicitně nastavena.
Možnosti trvalost, volby cíle, volby registrace a zástupné znaky	Tyto volby jsou nastaveny podle potřeby.
Volby publikování	Tyto volby jsou nastaveny podle potřeby s výjimkou volání MQSO_NEW_PUBLICATIONS_ONLY, které lze použít pouze pro objekt MQSO_CREATE.
Další volby	Tyto volby se při návratu z volání MQSUB nemění. Dořídí způsob, jakým je volání rozhraní API vydáno a které není uloženo s odběrem. Musí být nastaveny podle potřeby na všech následných volání MQSUB s opětným použitím struktury MQSD.
ObjectName	Toto vstupní pole je beze změny při návratu z volání MQSUB.
ObjectString	Toto vstupní pole je beze změny při návratu z volání MQSUB. Úplný název tématu, který se používá, je vrácen v poli *ResObjectString* , pokud je k dispozici vyrovnávací paměť.
ID AlternateUsera ID AlternateSecurity	Tato vstupní pole se nezmění po návratu z volání MQSUB. Dořídí způsob, jakým je volání rozhraní API vydáno a které není uloženo s odběrem. Musí být nastaveny podle potřeby na všech následných volání MQSUB s opětovným použitím produktu 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 znovu použijete MQSD v rámci volání MQSUB s použitím volby MQSO_ALTER, resetujte vypršení platnosti odběru znovu, aby se znovu počítaly.
SubName	Toto pole je vstupní pole pro volání MQSUB a není ve výstupu změněno.
SubUserData a SelectionString .	Tato pole s proměnnou délkou jsou vrácena ve výstupu z volání MQSUB s použitím volby MQSO_RESUME, je-li k dispozici vyrovnávací paměť, a také kladná délka vyrovnávací paměti v produktu *VSBufSize*. Pokud není poskytnuta žádná vyrovnávací paměť, je vrácena pouze délka v poli *VSLength* MQCHARV. Je-li poskytnutá vyrovnávací paměť menší než prostor potřebný k vrácení pole, vrátí se ve vyrovnávací paměti pouze *VSBufSize* bajtů. Pokud pak znovu použijete MQSD v rámci volání MQSUB pomocí volby MQSO_ALTER a vyrovnávací paměť není poskytnuta, ale je poskytnuta nenulová hodnota *VSLength* , pokud tato délka odpovídá existující délce pole, nedojde k žádné změně v poli.

Tabulka 557. Speciální pokyny pro pole v MQSD (pokračování)	
Název pole v MQSD	**Speciální aspekty.**
Token SubCorrelID a PubAccounting	Pokud nepoužíváte hodnotu MQSO_SET_CORREL_ID, správce front vygeneruje produkt *SubCorrelId* . Pokud nepoužíváte MQSO_SET_IDENTITY_CONTEXT, vygeneruje správce front produkt *PubAccountingToken* . Tato pole se vrací v MQSD z volání MQSUB s použitím volby MQSO_RESUME. Pokud jsou generovány správcem front, je generovaná hodnota vrácena v rámci volání MQSUB s použitím volby MQSO_CREATE nebo MQSO_ALTER.
PubPriority, SubLevel & PubApplIdentityData	Tato pole jsou vrácena ve struktuře MQSD.
Řetězec ResObject	Toto výstupní pole je vráceno ve struktuře MQSD, pokud je k dispozici 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 pro zachované publikování, pokud byl odběratel registrován u funkce MQSO_PUBLICATIONS_ON_REQUEST.

Syntaxe

MQSUBRQ (*Hconn*, *Hsub*, *Action*, *SubRqOpts*, *Compcode*, *Reason*)

Parametry

Hconn
Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

HSub
Typ: MQHOTBJ-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 na odběru. Musí být uvedena následující hodnota:

MQSR_ACTION_PUBLICATION

Tato akce vyžaduje odeslání publikování aktualizací pro určené téma. Lze ji použít pouze v případě, že odběratel určil volbu MQSO_PUBLICATIONS_ON_REQUEST při volání MQSUB při odběru daného odběru. Má-li správce front zachované publikování pro dané téma, odešle se tomuto odběrateli. Pokud tomu tak není, volání selže. Je-li aplikace odeslána publikování, která byla uchována, je tato publikace označena vlastností zprávy MQIsRetained této publikace.

Vzhledem k tomu, že téma v existujícím odběru představované parametrem Hsub může obsahovat zástupné znaky, může odběratel obdržet více zachovaných publikování.

SubRqOpts
Typ: MQSRO-input/output

Tyto volby řídí akci MQSUBRQ, podrobnosti viz “MQSRO-Volby požadavku na odběr” na stránce 583 .

Nejsou-li vyžadovány žádné volby, programy napsané v assembleru C nebo S/390 mohou místo zadání adresy struktury MQSRO uvádět adresu parametru null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení)

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
2298 (X'08FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

MQRC_NO_RETAINED_MSG
2437 (X'0985 ') Pro toto téma nejsou aktuálně uložena žádná zachovaná publikování.

CHYBA MQRC_OPTIONS_ERROR
2046 (X'07FE') Parametr nebo pole voleb obsahuje volby, které nejsou platné, nebo kombinace voleb, které nejsou platné.

UVÁDĚNÍ MQRC_Q_MGR_QUIESCING
2161 (X'0871 ') Správce front je uváděn do klidového stavu.

CHYBA MQRC_SRO_ERROR
2438 (X'0986 ') V rámci volání MQSUBRQ není volba MQSRO požadavku na odběr platná.

MQRC_RETAINED_MSG_Q_ERROR
2525 (X'09DD') Zachovaná publikování, která existují pro řetězec odebíraného tématu, nelze načíst.

MQRC_RETAINED_NOT_DELIVERED, DORUČENO
2526 (X'09DE') Zachované publikace, které existují pro odebíraný řetězec témat, nelze doručit do cílové fronty odběru a nelze ji doručit do fronty nedoručených zpráv.

Podrobné informace o těchto kódech najdete v tématu Zprávy a kódy příčin.

Poznámky k použití

Pro použití kódu akce MQSR_ACTION_PUBLIKACE se používají následující poznámky k použití:

1. Je-li toto příkazové slovo dokončeno úspěšně, zachované publikace odpovídající uvedenému odběru byly odeslány na odběr a lze je přijmout pomocí příkazu MQGET nebo MQCB pomocí objektu Hobj vráceného v původním příkazu MQSUB, který vytvořil odběr.

2. Pokud téma přihlášené k odběru původního příkazu MQSUB, které vytvořilo daný odběr, obsahovalo zástupný znak, může být odeslán více zachovaných publikování. Počet publikování odeslaných jako výsledek tohoto volání se zaznamenává do pole NumPubs ve struktuře Opts SubRq.
3. Pokud je toto příkazové slovo dokončeno s kódem příčiny MQRC_NO_RETAINED_MSG, pak nebyly v aktuálně zachovaných příručkách pro uvedené téma uvedeny žádné aktuálně zachované publikace. #
4. Je-li toto slovo dokončeno s kódem příčiny MQRC_RETAINED_MSG_Q_ERROR nebo MQRC_RETAINED_NOT_DELIVERED, jsou v daném tématu aktuálně zachované publikace, ale došlo k chybě, že to znamenalo, že nebylo možné je doručit.
5. Aplikace musí mít aktuální odběr pro dané téma, než bude moci toto volání provést. Pokud byl odběr proveden v předchozí instanci aplikace a není k dispozici platný popisovač pro daný odběr, musí aplikace nejprve zavolat funkci MQSUB s volbou MQSO_RESUME, aby získal popisovač pro použití v rámci tohoto volání.
6. Publikace se posílají na místo určení, které je registrováno pro použití s aktuálním odběrem této aplikace. Pokud je třeba publikace odeslat někde jinde, je třeba nejprve provést změnu odběru 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

Tato kolekce témat uvádí pouze ty objekty IBM MQ , které mohou být předmětem volání funkce MQINQ, a uvádí podrobnosti o atributech, které lze požadovat, a selektory, které se mají použít.

Atributy správce front

Některé atributy správce front jsou opraveny pro konkrétní implementace; ostatní lze změnit pomocí příkazu MQSC ALTER QMGR.

Atributy lze také zobrazit pomocí příkazu DISPLAY QMGR. Většinu atributů správce front lze provést otevřením speciálního objektu MQOT_Q_MGR a pomocí volání MQINQ s vraceným manipulátorem.

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 spolu s voláním MQINQ; názvy jsou stejné jako u příkazů PCF. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; další informace najdete v tématu Příkazy MQSC .

Tabulka 558. Atributy správce front

Atribut	Popis
AccountingConnOverride	Přepsat nastavení evidence.
AccountingInterval	Jak často zapisovat intermediační evidenční záznamy.
ActivityConnOverride	Přepsat nastavení aktivity.
ActivityTrace	Ovládá shromažďování trasování aktivity aplikace IBM MQ MQI.
AdoptNewMCACheck	Prvky zkontrolované, zda mají být přijaty nové MCA.
AdoptNewMCAType	Zda se má automaticky restartovat osamocená instance agenta 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 (neautorizované)
BridgeEvent	Řídicí atribut pro události mostu.
ChannelAutoDef	Řídí, zda je povolena automatická definice kanálu
ChannelAutoDefEvent	Řídí, zda jsou generovány události automatické definice kanálu
ChannelAutoDefExit	Název uživatelské procedury pro automatické definování kanálů
ChannelEvent	Řídicí atribut pro události kanálu.
ChannelInitiatorControl	Řídicí atribut pro inicializátor kanálu

Tabulka 558. Atributy správce front (pokračování)	
Atribut	**Popis**
ChannelMonitoring	Online monitorovací data pro kanály
ChannelStatistics	Řídí shromažďování statistických dat pro kanály.
ChinitAdapters	Počet podúloh adaptéru pro zpracování volání IBM MQ .
ChinitDispatchers	Počet dispečerů, který má být použit pro inicializátor kanálu.
	Vyhrazeno pro použití IBM .
ChinitTraceAutoStart	Určuje, zda má být trasování inicializátoru kanálu spuštěno automaticky.
ChinitTraceTableSize	Velikost datového prostoru pro trasování inicializátoru kanálu.
ClusterSenderMonitoringDefault	Výchozí údaje monitorování online pro odesílací kanály klastru
ClusterSenderStatistika	Ovládá 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 zpráv předaných uživatelskou proceduru 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
DEFCLXQ	Výchozí typ přenosové fronty klastru
DefXmitQName	Výchozí název přenosové fronty
DistLists	Podpora seznamu distribuce
DNSGroup	Název skupiny pro modul listener TCP při použití podpory služeb DNS (Dynamic Domain Name Services) správce pracovní zátěže.
DNSWLM	Zda se modul listener TCP registruje se správcem pracovní zátěže pro služby DNS (Dynamic Domain Name Services)
ExpiryInterval	Interval mezi skenováními pro vypršelé
IGQPutAuthority	Řazení do front v rámci skupiny
IGQUserId	Identifikátor uživatele fronty v rámci skupiny
InhibitEvent	Řídí, zda jsou generovány události inhibice (Inhibit Get a Inhibit Put)
IPAddressVersion	Verze adresy Internet Protocol
IntraGroupqueuing	Podpora řazení do front v rámci skupiny
ListenerTimer	Časový interval mezi pokusy o restartování 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	Generický název LU pro modul listener LU 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front.
LUName	Název jednotky 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 aktivní kdykoli.
MaxChannels	Maximální počet aktuálních kanálů.
MaxHandles	Maximální počet popisovačů

Tabulka 558. Atributy správce front (pokračování)	
Atribut	**Popis**
MaxMsgLength	Maximální délka zprávy v bajtech
MaxPriority	Maximální priorita
MaxPropertiesLength	Maximální délka dat vlastnosti v bajtech
MaxUncommittedMsgs	Maximální počet nepotvrzených zpráv v rámci jednotky práce
MQIAccounting	Řídí shromažďování účtovacích informací pro data MQI.
MQIStatistics	Ovládá shromažďování informací o monitorování statistiky pro správce front.
MsgMarkBrowseInterval	Interval, po jehož uplynutí může správce front odebrat značku ze procházených zpráv.
OutboundPortMin	S *OutboundPortMin*definuje rozsah čísel portů, které se mají použít při vázání odchozích kanálů.
OutboundPortMin	S *OutboundPortMax*definuje rozsah čísel portů, které se mají použít při vázání odchozích kanálů.
PerformanceEvent	Řídí, zda jsou generovány události související s výkonem
Platforma	Platforma, na které je správce front spuštěn.
PubSubNPInputMsg	Zda se má vyřadit (nebo uchovat) nedoručenou vstupní zprávu
PubSubNPResponse	Řídí chování nedoručené
PubSubMaxMsgRetryCount	Počet opakovaných pokusů při zpracování (pod synchronizačním bodem) zprávu příkazu se selháním
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
QSGName	Název skupiny sdílení front
QueueAccounting	Řídí shromažďování účtovacích informací pro fronty.
QueueMonitoring	Online monitorování dat pro fronty
QueueStatistics	Řídí shromažďování statistických dat pro fronty.
ReceiveTimeout	Jak dlouho bude kanál TCP/IP čekat na data, než se vrátí 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 události vzdálené chyby
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 profilů zabezpečení
NázevSharedQMgr	Název správce front sdílené fronty
“SPLCAP” na stránce 824	IBM MQ Rozšířená ochrana zabezpečení zpráv pro správce front byla zapnutá nebo vypnutá.
SSLCRLNamelist 1	Název objektu seznamu názvů obsahujícího názvy objektů ověřovacích informací.
SSLCryptoHardware 1	Řetězec konfigurace kryptografického hardwaru.
SSLEvent	Řídicí atribut pro události TLS.
SSLFIPSRequired	Pro šifrování používejte pouze certifikované algoritmy FIPS.
SSLKeyRepository 1	Umístění úložiště klíčů TLS.
PočetSSLKeyResetCount	Počet klíčů pro resetování klíče TLS.
SSLTasks 1	Počet podúloh serveru pro zpracování volání TLS.
StatisticsInterval	Jak často zapisovat 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.

Tabulka 558. Atributy správce front (pokračování)	
Atribut	**Popis**
TCPKeepAlive	Zda se má použít TCP KEEPALIVE ke kontrole dalšího konce připojení.
TCPName	Název systému TCP/IP, který používáte.
TCPStackType	Jak iniciátor kanálu může používat adresy TCP/IP.
TraceRouteAtribut záznamu	Ovládá záznam informací o přenosové cestě trasování.
TriggerInterval	Trigger-interval zpráv
verze	Verze
XrCapability	Určuje, zda jsou podporovány příkazy Telemetry.
Notes: 1. Tento atribut nelze provést pomocí volání MQINQ a není popsán v této sekci. Podrobnosti o tomto atributu najdete v tématu Změna správce front .	

Související úlohy

Určení, že pro běhové prostředí klienta MQI je použit pouze certifikovaný standard FIPS CipherSpecs

Související odkazy

Federální standardy zpracování informací (FIPS) pro AIX, Linux, and Windows

Přepsání AccountingConn(MQLONG)

To umožňuje aplikacím potlačit nastavení hodnot ACCTMQI a ACCTQDATA v atributu Qmgr.

Hodnota je jedna z následujících možností:

MQMON_DISABLED
Aplikace nemohou přepsat nastavení atributů ACCTMQI a ACCTQ Qmgr pomocí pole Volby ve struktuře MQCNO v rámci volání MQCONNX. Toto je výchozí hodnota.

MQMON_POVOLENO
Aplikace mohou přepsat atributy ACCTQ a ACCTMQI Qmgr pomocí pole Volby ve struktuře MQCNO.

Změny této hodnoty jsou platné pouze pro 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)

Uvádí, jak dlouho před zápisem intermediačních záznamů evidence (v sekundách).

Hodnota je celé číslo v rozsahu od 0 do 604800, s výchozí hodnotou 1800 (30 minut). Chcete-li vypnout mezilehlé 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.

Potlačení ActivityConn(MQLONG)

To umožňuje aplikacím potlačit nastavení hodnoty ACTVTRC v atributu správce front.

Hodnota je jedna z následujících možností:

MQMON_DISABLED
Aplikace nemůže přepsat nastavení atributu správce front ACTVTRC pomocí pole Volby ve struktuře MQCNO v rámci volání MQCONNX. Toto je výchozí hodnota.

MQMON_POVOLENO
Aplikace mohou přepsat atribut správce front ACTVTRC pomocí pole Volby ve struktuře MQCNO.

Změny této hodnoty jsou platné pouze pro připojení ke správci front po změně atributu.

Tento atribut je podporován pouze v systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVITY_CONN_OVERRRIDE s voláním MQINQ .

ActivityTrace (MQLONG)

Tento ovládací prvek řídí shromažďování dat trasování aktivity aplikace IBM MQ MQI.

Hodnota je jedna z následujících možností:

MQMON_ON
Shromažďovat trasování aktivity aplikace MQI produktu IBM MQ .

MQMON_OFF
Neshromažďovat trasování aktivity aplikace IBM MQ MQI. Toto je výchozí hodnota.

Pokud nastavíte atribut správce front ACTVCONO na hodnotu ENABLED, může být tato hodnota potlačena pro jednotlivá připojení s použitím pole Volby ve struktuře MQCNO.

Změny této hodnoty jsou platné pouze pro připojení ke správci front po změně atributu.

Tento atribut je podporován pouze v systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVITY_TRACE s voláním MQINQ .

AdoptNewMCACheck (MQLONG)

Definuje prvky, které mají být zkontrolovány, zda má být převzata sběrnice MCA při zjištění nového příchozího kanálu, který má stejný název jako agent MCA, který je již aktivní.

Hodnota je jedna z následujících možností:

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. Je-li to možné, proveďte tuto kontrolu, abyste ochránili své kanály před vypnutím, nechtěným nebo nevědomým způsobem. Toto je výchozí hodnota.

MQADOPT_CHECK_NONE
Nekontrolovat žádné prvky.

Změny tohoto atributu se projeví až při příštím pokusu kanálu o přijetí kanálu.

z/OS Tento atribut je podporován pouze v systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ADOPT_NEWWCA_CHECK s voláním MQINQ.

AdoptNewMCATType (MQLONG)

Uvádí, zda se má automaticky restartovat osiřelá instance MCA určitého typu kanálu, když je zjištěn nový požadavek příchozího kanálu odpovídající atributu MCACheck AdoptNew.

Je to jedna z následujících hodnot:

MQADOPT_TYPE_NO
Adopce osiřelých instancí kanálu není vyžadováno. Toto je výchozí hodnota.

MQADOPT_TYPY_VŠE
Převzetí všech typů kanálů.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ADOPTNEWCA_TYPE s voláním MQINQ.

AlterationDate (MQCHAR12)

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, doplněno 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)

Jedná se o 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)

Tento ovládací prvek určuje, zda jsou generovány události autorizace (neautorizováno). Je to jedna z následujících hodnot:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_AUTHORITY_EVENT s voláním MQINQ.

BridgeEvent (MQLONG)

Určuje, zda jsou generovány události mostu IMS .

Hodnota je jedna z následujících možností:

POVOLENÝ MQEVR_
Generujte události mostu IMS následujícím způsobem:

MQRC_BRIDGE_STARTED
MQRC_BRIDGE_STOPPED

MQEV_DISABLED
Negenerovat události mostu IMS ; jedná se o výchozí hodnotu.

Tento atribut je podporován pouze pro 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. Automatické definování kanálů MQCHT_CLUSSDR je vždy povoleno. Hodnota je jedna z následujících možností:

MQCHAD_DISABLED
Automatická definice kanálu je zakázána.

MQCHAD_ENABLED
Automatická definice kanálu je povolena.

Multi Tento atribut je podporován pouze v systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHANNEL_AUTO_DEF s voláním MQINQ.

ChannelAutoDefEvent (MQLONG)

Tento ovládací prvek určuje, zda jsou generovány události automatické definice kanálu. Vztahuje se na kanály typu MQCHT_RECEIVER, MQCHT_SVRCONN a MQCHT_CLUSSDR. Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Multi Tento atribut je podporován pouze v 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 automatické definování kanálu. Pokud je tento název neprázdný a *ChannelAutoDef* má hodnotu MQCHAD_ENABLED, je uživatelská procedura volána pokaždé, když správce front chystá vytvořit definici kanálu. Toto platí pro kanály typu MQCHT_RECEIVER, MQCHT_SVRCONN a MQCHT_CLUSSDR. Ukončení může poté provést jednu z následujících možností:

- Vytvořte definici kanálu beze změny.
- Upravte atributy definice kanálu, která je vytvořena.
- Zcela potlačte vytvoření kanálu.

Poznámka: Jak délka, tak i hodnota tohoto atributu jsou specifické pro prostředí. Podrobné informace o hodnotě tohoto atributu v různých prostředích najdete v úvodu ke struktuře MQCD v produktu “MQCD-Definice kanálu” na stránce 1464 .

z/OS V systému z/OSse tento atribut používá pouze pro kanály odesílatele klastru a příjemce 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)

Určuje, zda jsou generovány události kanálu.

Je to jedna z následujících hodnot:

VÝJIMKA MQEVR_EXCEPTION
Generovat pouze následující události kanálu:

- MQRC_CHANNEL_ACTIVATED
- MQRC_CHANNEL_CONV_ERROR

- MQRC_CHANNEL_NOT_ACTIVATED
- Objekt MQRC_CHANNEL_STOPPED s následujícím kódem příčiny ReasonQualifiers:

  MQRQ_CHANNEL_STOPPED_ERROR
  MQRQ_CHANNEL_STOPPED_RETRY
  MQRQ_CHANNEL_STOPPED_DISABLED

  MQRC_CHANNEL_STOPPED_BY_USER

POVOLENÝ MQEVR_
Generujte všechny události kanálu. Kromě těch generovaných VÝJIMKOU EXCEPTION vygenerují následující události kanálu:

- MQRC_CHANNEL_STARTED
- Objekt MQRC_CHANNEL_STOPPED s následujícím kódem příčiny ReasonQualifier:

  MQRQ_CHANNEL_STOPPED_OK

MQEV_DISABLED
Negenerovat události kanálu; jedná se o výchozí hodnotu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHANNEL_EVENT s voláním MQINQ.

Řízení ChannelInitiator(MQLONG)
To určuje, zda má být inicializátor kanálu spuštěn při spuštění správce front.

Je to jedna z následujících hodnot:

MQSVC_CONTROL_MANUAL
Inicializátor kanálu není třeba spustit automaticky.

MQSVC_CONTROL_Q_MGR
Inicializátor kanálu má být spuštěn automaticky při spuštění správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_CONTROL s voláním MQINQ.

ChannelMonitoring (MQLONG)
Tento atribut určuje online monitorování dat pro kanály.

Hodnota je jedna z následujících možností:

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
Zapne shromažďování dat monitorování s nízkým poměrem shromažďování dat pro kanály, které v atributu kanálu MONCHL uvádí QMGR.

MQMON_MEDIUM
Zapnout shromažďování dat monitorování se středním poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu MONCHL.

MQMON_HIGH
Zapnout shromažďování dat monitorování s vysokým poměrem shromažďování dat pro kanály, které uvádí 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)

Tento ovládací prvek řídí shromažďování statistických dat pro kanály.

Hodnota je jedna z následujících možností:

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 určují QMGR v atributu kanálu STATCHL.

MQMON_LOW
Zapnout shromažďování statistických dat s nízkým poměrem shromažďování dat pro kanály, které uvádí QMGR v atributu kanálu STATCHL.

MQMON_MEDIUM
Zapnout shromažďování statistických dat se středním poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu STATCHL.

MQMON_HIGH
Zapnout shromažďování statistických dat s vysokým poměrem shromažďování dat pro kanály uvádějící QMGR v atributu kanálu STATCHL.

Pro většinu systémů se doporučuje používat MEDIUM. Avšak pro kanál, který zpracovává vysoký objem zpráv každou sekundu, byste mohli chtít snížit úroveň vzorkování výběrem NÍZKÝ. Také u kanálu, který zpracovává pouze několik zpráv a u kterých jsou nejaktuálnější informace důležité, můžete chtít vybrat hodnotu HIGH (vysoká).

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 mají být použity ke zpracování volání produktu IBM MQ . Hodnota musí být 0-9999, přičemž výchozí hodnota je 8.

Poměr adaptérů k dispečerům (atribut ChinitDispatchers) by měl být přibližně 8 až 5. Pokud však máte pouze málo 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 míru paralelizmu volání produktu IBM MQ . To je důležité pro trvalé zprávy. Méně adaptérů může být lepší pro přechodné zprávy.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_ADAPTERS s voláním MQINQ.

ChinitDispatchers (MQLONG)

Jedná se o počet dispečerů, který má být použit pro inicializátor kanálu. Hodnota musí být 0-9999, přičemž výchozí hodnota je 5.

Jako vodítko můžete povolit jeden dispečer pro 50 aktuálních kanálů. Pokud však máte pouze málo kanálů, nemusíte hodnotu tohoto atributu snižovat z výchozí hodnoty. Pokud používáte protokol TCP/IP, je největší počet dispečerů použitých pro kanály TCP/IP 100, a to i v případě, že zde uvedete větší hodnotu. Můžete použít následující nastavení: testovací systémy, 5 (výchozí); produkční systémy, 20 (potřebujete 20 dispečerů, které mají zpracovat až 1000 aktivních kanálů).

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_DISPATCHERS s voláním MQINQ.

ChinitTraceAutoStart (MQLONG)

Tato volba určuje, zda má být trasování inicializátoru kanálu provedeno automaticky.

Hodnota je jedna z následujících možností:

MQTRAXSTR_YES
Spustit trasování inicializátoru kanálu automaticky. Toto je výchozí hodnota.

MQTRAXSTR_NO
Nespouštějte trasování inicializátoru kanálu automaticky.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_AUTO_START s voláním MQINQ.

ChinitTraceTableSize (MQLONG)

Jedná se o velikost datového prostoru trasování inicializátoru kanálu (v MB).

Hodnota musí být v rozsahu 0 až 2048, s výchozí hodnotou 2.

Poznámka: Kdykoli použijete velké datové prostory produktu z/OS , ujistěte se, že máte v systému dostatek pomocné paměti pro podporu všech souvisejících aktivit stránkování z/OS . Pravděpodobně bude potřeba také zvýšit velikost datových sad SYS1.DUMP.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_SIZE s voláním MQINQ.

ClusterSenderMonitoringDefault (MQLONG)

Tato hodnota určuje hodnotu, která má být nahrazena atributem ChannelMonitoring automaticky definovaného odesílacího kanálu klastru.

Hodnota je jedna z následujících možností:

MQMON_Q_MGR
Shromažďování online monitorovacích dat je zděděno z nastavení atributu správce front **ChannelMonitoring** . Toto je výchozí hodnota.

MQMON_OFF
Monitorování pro kanál je zakázáno

MQMON_LOW
Pokud produkt *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 účinkem na výkon systému.

MQMON_HIGH
Pokud produkt *ChannelMonitoring* není MQMON_NONE, monitorování je povoleno s vysokou mírou shromažďování dat s pravděpodobným vlivem na výkon systému. Shromážděná data jsou nejaktuálnějším dostupným.

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 mohou být automaticky definovány z definice CLUSRCVR v úložišti, nemůžete změnit nastavení atributu STATCHL pro tyto automaticky definované kanály odesilatele klastru pomocí kanálu ALTER. Pro tyto kanály je rozhodnutí, zda shromažďovat online data monitorování, založeno na nastavení tohoto atributu správce front.

Hodnota je jedna z následujících možností:

MQMON_Q_MGR
Shromažďování statistických dat pro automaticky definované kanály odesílatele klastru je založeno na hodnotě atributu správce front STATCHL. Toto je výchozí hodnota.

MQMON_OFF
Vypněte shromažďování statistických dat pro automaticky definované kanály odesílatele klastru.

MQMON_LOW
Povolit shromažďování statistických dat pro automaticky definované kanály odesílatele klastru s nízkým poměrem shromažďování dat.

MQMON_MEDIUM
Povolit shromažďování statistických dat pro automaticky definované kanály odesílatele klastru se středním poměrem shromažďování dat.

MQMON_HIGH
Povolit shromažďování statistických dat pro automaticky definované kanály odesílatele klastru s vysokým poměrem shromažďování dat.

Pro většinu systémů doporučujeme hodnotu MEDIUM. Avšak u automaticky definovaného odesílacího kanálu klastru, který zpracovává vysoký objem zpráv každou sekundu, můžete chtít snížit úroveň vzorkování výběrem NÍZKÝ. Také u kanálu, který zpracovává pouze několik zpráv a u kterých jsou nejaktuálnější informace důležité, můžete chtít vybrat hodnotu HIGH (vysoká).

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.

Data ClusterWorkloadData (MQCHAR32)

Jedná se o uživatelsky definovaný 32bajtový řetězec znaků, který je předán uživatelské proceduře pracovní zátěže klastru, když je volán. Nejsou-li k dispozici žádná data pro předání do procedury ukončení, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CLUSTER_WORKLOAD_DATA s voláním MQINQ.

ClusterWorkloadExit (MQCHARn)

Jedná se o název uživatelské procedury pro správu pracovní zátěže klastru. Pokud tento název není prázdný, je uživatelská procedura volána při každém vložení zprávy do fronty klastru nebo přesunu z jedné fronty odesílatele klastru do jiné fronty. Uživatelská procedura pak může buď přijmout instanci fronty vybranou správcem front jako místo určení zprávy, nebo vybrat jinou instanci fronty.

Poznámka: Jak délka, tak 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.

Délka ClusterWorkload(MQLONG)

Jedná se o maximální délku dat zpráv, která jsou předána uživatelské proceduře pracovní zátěže klastru. Skutečná délka dat předaných do uživatelské procedury je minimálně:

- 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 hodnota určuje maximální počet nejčastěji používaných kanálů klastru, které mají být brány v úvahu pro použití algoritmem volby pracovní zátěže klastru.

Jedná se o hodnotu v rozsahu od 1 do 999999999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_MRU_CHANNELS s voláním MQINQ.

CLWLUseQ (MQLONG)

Tato volba určuje, zda mají být použity vzdálené fronty pro pracovní zátěž klastru.

Hodnota je jedna z následujících možností:

MQCLWL_USEQ_ANY
Použijte lokální i vzdálené fronty.

MQCLWL_USEQ_LOCAL
Nepoužívejte vzdálené fronty. Toto je výchozí hodnota.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_USEQ s voláním MQINQ.

CodedCharSetId (MQLONG)

Definuje znakovou sadu používanou správcem front pro všechna pole znakového řetězce, která jsou definována v rozhraní MQI, jako jsou například názvy objektů a datum a čas vytvoření fronty. Znaková sada musí být taková, která má jednobajtové znaky pro znaky, které jsou platné v názvech objektů. Nevztahuje se na data aplikace přenášené ve zprávě. Hodnota závisí na prostředí:

- V systému z/OSje tato hodnota nastavena ze systémových parametrů při spuštění správce front; výchozí hodnota je 500.
- V systému Windowsje hodnota primární CODEPAGE uživatele, který vytváří správce front.
- V systému IBM ise jedná o hodnotu, která je nastavena v prostředí při prvním vytvoření správce front.
- V systému AIX and Linuxje hodnota výchozí hodnotou 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)

Uvádí, zda jsou generovány události příkazu, jak je uvedeno níže:

MQEV_DISABLED
Negenerovat události příkazu. Toto nastavení je výchozí.

POVOLENÝ MQEVR_
Generovat události příkazu.

MQEVR_NO_DISPLAY
Události příkazů 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; lze do něj odesílat příkazy MQSC a PCF. Podrobné informace o příkazech MQSC a Definice programových formátů příkazů naleznete v části Příkazy MQSC . Podrobné informace o příkazech PCF najdete v tématu Příkazy MQSC.
- Ve všech ostatních prostředích je název fronty SYSTEM.ADMIN.COMMAND.QUEUEa lze do ní odesílat pouze příkazy PCF. Avšak do této fronty lze odeslat příkaz MQSC, pokud je příkaz MQSC uzavřen v rámci příkazu PCF typu MQCMD_ESCAPE. Informace o příkazu Escape naleznete v části 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 pro operační systém HP-UX pro všechny komponenty produktu IBM MQ , včetně serveru a klientů, je odebrána z produktu IBM MQ 9.1.

Značí úroveň příkazů řízení systému podporovaných správcem front. Může jít 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 řídicích příkazů 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 řídicích příkazů 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 řídicích příkazů 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 řídicích příkazů 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 řídicích příkazů 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 řídicích příkazů 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 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.0
- IBM MQ for IBM i 9.1.0
- IBM MQ for Linux 9.1.0
- IBM MQ for Windows 9.1.0
- IBM MQ for z/OS 9.1.0

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 Windows9.1.1
- IBM MQ for z/OS 9.1.1

MQCMDL_LEVEL_912
Úroveň 912 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.2
- IBM MQ for Linux 9.1.2
- IBM MQ for Windows9.1.2
- 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 Windows9.1.3
- 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 Windows9.1.4
- IBM MQ for z/OS 9.1.4

MQCMDL_LEVEL_915
Úroveň 915 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.1.5
- IBM MQ for Linux 9.1.5
- IBM MQ for Windows9.1.5
- IBM MQ for z/OS 9.1.5

MQCMDL_LEVEL_920
Úroveň 920 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.0
- IBM MQ for IBM i 9.2.0
- IBM MQ for Linux 9.2.0
- IBM MQ for Windows 9.2.0
- IBM MQ for z/OS 9.2.0

MQCMDL_LEVEL_921
Úroveň 921 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.1
- IBM MQ for Linux 9.2.1
- IBM MQ for Windows9.2.1
- IBM MQ for z/OS 9.2.1

MQCMDL_LEVEL_922
Úroveň 922 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.2
- IBM MQ for Linux 9.2.2
- IBM MQ for Windows9.2.2
- IBM MQ for z/OS 9.2.2

MQCMDL_LEVEL_923
Úroveň 923 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.3
- IBM MQ for Linux 9.2.3
- IBM MQ for Windows9.2.3
- IBM MQ for z/OS 9.2.3

MQCMDL_LEVEL_924
Úroveň 924 řídicích příkazů systému.

Tato hodnota je vrácena následujícími verzemi:

- IBM MQ for AIX 9.2.4
- IBM MQ for Linux 9.2.4
- IBM MQ for Windows9.2.4
- IBM MQ for z/OS 9.2.4

MQCMDL_LEVEL_925
Úroveň 925of 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 Windows9.2.5
- IBM MQ for z/OS 9.2.5

Nastavení řídicích příkazů systému, které odpovídají určité hodnotě atributu **CommandLevel** , se liší v závislosti na hodnotě atributu **Platform** ; oba musí být použity při rozhodování o tom, které řídicí příkazy systému jsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_COMMAND_LEVEL s voláním MQINQ .

CommandServerControl (MQLONG)
Určuje, zda má být spuštěn příkazový server při spuštění správce front.

Hodnota může být některá z následujících:

MQSVC_CONTROL_MANUAL
Příkazový server nemá být spuštěn automaticky.

MQSVC_CONTROL_Q_MGR
Příkazový server má být spuštěn automaticky při spuštění správce front.

Tento atribut není v produktu 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:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Multi V 9.2.0 *Velikost CurrentQFile(MQLONG)*

Aktuální velikost souboru fronty v megabajtech zaokrouhlená nahoru na nejbližší megabajt.

Tabulka 559. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota pro tento atribut stavu fronty je libovolná velikost fronty, která je aktuálně rovna, zaokrouhlená na nejbližší megabajt. Pro novou frontu s výchozími atributy je hodnotou **CurrentQFileSize** hodnota 1.

Maximální hodnota tohoto atributu je 99,999,9999 MB a pro tento atribut neexistuje žádná výchozí hodnota.

Multi V 9.2.0 *CurrentMaxQFileSize (MQLONG)*

Aktuální maximální velikost souboru fronty může růst zaokrouhlený na nejbližší megabajt, vzhledem k aktuální velikosti bloku, který je ve frontě použit.

Tabulka 560. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Použití tohoto pole je dvojnásobné:

- Pokud nastavíte **MaxQFileSize** na výchozí hodnotu pro aktuální velikost bloku, **CurrentMaxQFileSize** zobrazí skutečnou hodnotu, která se rovná výchozí hodnotě.
- Pokud se **CurrentMaxQFileSize** neshoduje s **MaxQFileSize**, musíte frontu vyprázdnit, aby bylo možné přijmout větší granularitu.

Poznámka: Další informace o změně velikosti souborů fronty a velikosti bloků a granularity naleznete v tématu Úprava souborů fronty produktu IBM MQ .

Maximální hodnota tohoto atributu je 99,999,9999 MB a není zde žádná výchozí hodnota. Hodnota je libovolná maximální hodnota, která je momentálně nastavena; pro novou frontu s výchozími atributy je hodnota **CurrentMaxQFileSize** 2,088,960 MB.

DeadLetterQName (MQCHAR48)

Jedná se o název fronty definované v lokálním správci front jako frontu nedoručených zpráv (nedoručená zpráva). Zprávy se odesílají do této fronty, pokud nemohou být směrovány na jejich správné místo určení.

Například zprávy jsou vloženy do této fronty, když:

- Zpráva dorazí do správce front, který je určen pro frontu, která dosud není definována v daném správci front.
- Zpráva dorazí do správce front, ale fronta, pro kterou je určena, ji nemůže přijmout, protože pravděpodobně:
  - Fronta je plná
  - Požadavky PUT jsou blokovány
  - Odesílající uzel nemá oprávnění vkládat zprávy do fronty

Aplikace mohou také vkládat zprávy do fronty nedoručených zpráv.

Zprávy sestav se zpracovávají stejným způsobem jako běžné zprávy; pokud nelze zprávu sestavy doručit do cílové fronty (obvykle do fronty zadané v poli *ReplyToQ* v deskriptoru zprávy původní zprávy), bude zpráva sestavy umístěna do fronty nedoručených zpráv (nedoručená zpráva).

Poznámka: Zprávy, které prošly dobou vypršení platnosti (viz pole MQMD-Expiry) **nejsou** převedeny do této fronty, když jsou zahozeny. Avšak zpráva o vypršení platnosti (MQRO_EXPIRATION) je stále generována a odeslána do fronty *ReplyToQ* , pokud ji požaduje odesílající aplikace.

Zprávy nejsou vloženy do fronty nedoručených zpráv (nedoručená zpráva), pokud byla aplikace, která vydala požadavek na vložení, oznámena synchronně prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 (například zpráva vložena do lokální fronty, pro kterou jsou zakázány žádosti).

Zprávy ve frontě nedoručených zpráv (undelivered-message) mají někdy k dispozici data zpráv aplikace s předponou ve struktuře MQDLH. Tato struktura obsahuje další informace, které ukazují, proč byla zpráva vložena do fronty nedoručených zpráv (nedoručená zpráva). Další podrobnosti o této struktuře viz “MQDLH-Záhlaví nedoručených zpráv” na stránce 349 .

Tato fronta musí být lokální frontou, s atributem **Usage** MQUS_NORMAL.

Pokud správce front nepodporuje frontu nedoručených zpráv (nedoručená zpráva) nebo nebyla definována, je název prázdný. Všichni správci front produktu IBM MQ podporují frontu nedoručených zpráv (nedoručená zpráva), ale při výchozím nastavení není definována.

Není-li fronta nedoručených zpráv (undelivered-message) definována, plná nebo nepoužitelná z nějakého jiného důvodu, bude místo přenosové fronty uchována zpráva, která by byla agentem kanálu zpráv přenesena do tohoto agenta.

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.

DefClusterXmitQueueType (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.

SCTQ se nastaví při definici správce front. Toto chování je implicitní ve verzích produktu IBM WebSphere MQ před verzí IBM WebSphere MQ 7.5. Ve starších verzích nebyl parametr správce front `DefClusterXmitQueueType` přítomen.

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`.

Je-li atribut správce front `DefClusterXmitQueueType`nastaven 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 na příkaz MQINQnebo odešlete příkaz PCF produktu Inquire Queue Manager (MQCMD_INQUIRE_Q_MGR), nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE . Chcete-li změnit hodnotu, odešlete příkaz PCF správce front změn (MQCMD_CHANGE_Q_MGR) a nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE .

Související odkazy

Změnit správce front

Zjistit správce front

“MQINQ-Dotaz na atributy objektu” na stránce 699

Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

DefXmitQName (MQCHAR48)

Jedná se o název přenosové fronty, která se používá pro přenos zpráv do vzdálených správců front, pokud neexistuje žádná jiná indikace toho, jakou přenosovou frontu použít.

Pokud neexistuje žádná předvolená přenosová fronta, jméno je zcela prázdné. 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)

To označuje, zda lokální správce front podporuje distribuční seznamy na volání MQPUT a MQPUT1 . Je to jedna z následujících hodnot:

PODPOROVANÁ MQDL_
Podporované seznamy distribucí.

PODPOROVÁNO MQDL_NOT_SUPPORTED
Distribuční seznamy nejsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DIR_LISTS s voláním MQINQ.

Skupina DNSGroup (MQCHAR18)

Tento parametr není již používán. Viz Co se změnilo v IBM MQ 8.0.

Tento atribut je podporován pouze pro 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 IBM MQ 8.0.

Hodnota je jedna z následujících možností:

MQDNSWLM_YES
Tato hodnota se může zobrazit ve správci front migrovaných z dřívějšího vydání. Hodnota je ignorována.

MQDNSWLM_NO
Jedná se o jedinou hodnotu podporovanou správcem front.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DNS_WLM s voláním MQINQ.

ExpiryInterval (MQLONG)

Toto označuje frekvenci, se kterou správce front prohledává fronty s ohledem na zprávy s prošlou platností. Je to buď časový interval v sekundách v rozsahu od 1 do 99 999 999, nebo následující speciální hodnota:

MQEXPI_OFF
Správce front neskenuje fronty, které hledají 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 v systému z/OS.

IGQPutAuthority (MQLONG)

Tento atribut se použije pouze v případě, že lokální správce front je členem skupiny sdílení front. Označuje typ kontroly oprávnění, která se provádí, když lokální agent fronty v rámci skupiny (agent IGQ) odebere zprávu ze sdílené přenosové fronty a umístí zprávu do lokální fronty. Hodnota je jedna z následujících možností:

MQIGQPA_DEFAULT
Identifikátor uživatele kontrolovaný pro autorizaci je hodnota pole *UserIdentifier* v *samostatném* MQMD, který je přidružen ke zprávě, když se zpráva nachází 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, je kontrolován také identifikátor uživatele lokálního IGQ agenta (*IGQUserId*).

KONTEXT MQIGQPA_CONTEXT
Identifikátor uživatele kontrolovaný pro autorizaci je hodnota pole *UserIdentifier* v *samostatném* MQMD, který je přidružen ke zprávě, když se zpráva nachází 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, identifikátor uživatele lokálního IGQ (*IGQUserId*) a hodnota pole *UserIdentifier* ve *vloženém* MQMD jsou také zkontrolovány. Posledně jmenovaný identifikátor uživatele je obvykle identifikátor uživatele aplikace, která je původcem zprávy.

MQIGQPA_ONLY_IGQ
Identifikátor uživatele kontrolovaný pro autorizaci je identifikátorem 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_NEBOIGQ
Identifikátor uživatele kontrolovaný pro autorizaci je identifikátorem uživatele lokálního agenta IGQ (*IGQUserId*).

Pokud profil RESLEVEL označuje, že se má zkontrolovat více než jeden identifikátor uživatele, je také zkontrolována hodnota pole *UserIdentifier* ve *vloženém* MQMD. Tento identifikátor uživatele je obvykle identifikátor uživatele aplikace, která pochází ze zprávy.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_IGQ_PUT_AUTHORITY s voláním MQINQ.

z/OS Tento atribut je podporován pouze v systému z/OS.

IGQUserId (MQLONG)

Tento atribut lze použít pouze v případě, je-li lokální správce front členem skupiny sdílení front. Určuje identifikátor uživatele, který je přidružen k lokálnímu agentovi front v rámci skupiny (agent IGQ). Tento

identifikátor je jedním z identifikátorů uživatelů, které lze zkontrolovat při autorizaci, když agent IGQ ukládá zprávy do lokálních front. Kontrolované skutečné identifikátory uživatelů závisí na nastavení atributu **IGQPutAuthority** a na externích volbách zabezpečení.

Pokud je parametr *IGQUserId* prázdný, není k agentovi IGQ přidružen žádný identifikátor uživatele a odpovídající kontrola autorizace se neprovede (ačkoli mohou být další identifikátory uživatelů 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 v systému z/OS.

InhibitEvent (MQLONG)

Tento ovládací prvek určuje, zda jsou generovány události blokování (Inhibit Get a Inhibit Put). Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INHIBIT_EVENT s voláním MQINQ.

V systému z/OSnelze použít volání MQINQ k určení hodnoty tohoto atributu.

IntraGroupqueuing (MQLONG)

Tento atribut se použije pouze v případě, že lokální správce front je členem skupiny sdílení front. Určuje, zda je pro skupinu sdílení front povoleno řazení do fronty v rámci skupiny. Hodnota je jedna z následujících možností:

MQIGQ_DISABLED
Všechny zprávy určené pro ostatní správce front v rámci skupiny sdílení front jsou přenášeny pomocí konvenčních kanálů.

MQIGQ_ENABLED
Zprávy určené pro ostatní správce front v rámci skupiny sdílení front jsou přenášeny prostřednictvím sdílené přenosové fronty, pokud je splněna následující podmínka:

- Délka dat zprávy a záhlaví přenosu nepřesahuje 63 kB (64 512 bajtů).

  Doporučuje se, aby byl pro záhlaví přenosu přidělen o něco více prostoru, než je velikost MQXQH; pro tento účel je k dispozici konstanta MQ_MSG_HEADER_LENGTH.

Pokud tato podmínka není splněna, zpráva se přenáší pomocí konvenčních kanálů.

Poznámka: Je-li povoleno ukládání do front v rámci skupiny, není pořadí zpráv přenesených pomocí sdílené přenosové fronty zachováno vzhledem k přenášeným přenosovým kanálem s použitím konvenčních kanálů.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INTRA_GROUP_GROUPs s voláním MQINQ.

z/OS Tento atribut je podporován pouze v systému z/OS.

IPAddressVersion (MQLONG)

Určuje, která verze adresy IP se použije buď IPv4 , nebo IPv6.

Tento atribut je relevantní pouze pro systémy, které spouštějí jak IPv4 , tak IPv6 a mají vliv pouze na kanály definované jako *TransportType* MQXPY_TCP, je-li jedna z následujících podmínek pravdivá:

- *ConnectionName* kanálu je název hostitele, který se interpretuje jak na adresu IPv4 , tak na adresu IPv6 a jeho argument **LocalAddress** není zadán.
- *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 IBM MQ pokusy o restartování modulu listener, pokud došlo k selhání APPC nebo TCP/IP. Hodnota musí být mezi 5 a 9999, přičemž výchozí hodnota je 60.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LISTENER_TIMER s voláním MQINQ.

LocalEvent (MQLONG)

To řídí, zda se generují lokální chybové události. Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LOCAL_EVENT s voláním MQINQ.

V systému z/OSnelze použít volání MQINQ k určení hodnoty tohoto atributu.

LoggerEvent (MQLONG)

Tento příkaz řídí, zda jsou generovány události protokolu o zotavení. Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LOGGER_EVENT s voláním MQINQ.

Tento atribut je podporován pouze v systému Multiplatforms.

LUGroupName (MQCHAR8)

Jedná se o generický název jednotky LU pro modul listener LU 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front. Ponecháte-li tento název prázdný, nebude možné tento modul listener použít.

Tento atribut je podporován pouze pro 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 LUN (MQCHAR8)

Jedná se o název jednotky LU, která má být použita pro odchozí přenosy LU 6.2 . Nastavte ji na stejnou logickou jednotku, kterou modul listener používá pro příchozí přenosy. Ponecháte-li tento název prázdný, použije se výchozí LU APPC/MVS; jedná se o proměnnou, takže je vždy nastaveno, pokud používáte LU6.2.

Tento atribut je podporován pouze pro 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 se vydá, když ARM restartuje inicializátor kanálu. Ponecháte-li toto jméno prázdné, nebude vydáno žádné SET APPC=xx.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU62_ARM_SUFFIX pomocí volání 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 které používají přenosový protokol LU 6.2 .

Hodnota musí být v rozsahu 0 až 9999, přičemž výchozí hodnota je 200. Pokud nastavíte tuto hodnotu na nulu, nebude přenosový protokol LU 6.2 použit.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_LU62_CHANNELS s voláním MQINQ.

MaxActivekanálů (MQLONG)

Tento atribut je maximální počet kanálů, které mohou být *aktivní* kdykoli.

Výchozí hodnota je extrahována z atributu `MaxChannels`.

Pro z/OSmusí být hodnota v rozsahu od 1 do 9 999.

Pro všechny ostatní platformy je výchozí hodnota 999 999 999, což znamená, že počet aktivních kanálů je neomezený, nebo může být nastaven na skutečné číslo, aby se vynutili omezení.

Hodnotu **MaxActiveChannels** v systému IBM MQ Appliancebyste neměli měnit. Chcete-li omezit maximální počet kanálů klienta, použijte k definování limitů pro každý kanál SVRCONN atributy MAXINST a MAXINSTC v definicích kanálu SVRCONN, viz téma Konfigurace správce front v produktu IBM MQ Appliance v dokumentaci produktu IBM MQ Appliance .

Argument **MaxActiveChannels** je atribut správce front pouze v systému z/OS . Na ostatních platformách **MaxActiveChannels** je atribut v souboru `qm.ini` . Informace o tom, jak nastavíte atribut **MaxActiveChannels** na jiných platformách, najdete v tématu Stanzy konfiguračního souboru pro distribuované řazení do fronty .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVE_CHANNELS s voláním **MQINQ** .

Související pojmy

Stavy kanálů

MaxChannels (MQLONG)

Tento atribut je maximální počet kanálů, které mohou být *aktuální* (včetně kanálů připojení serveru s připojenými klienty).

Pro z/OSmusí být hodnota v rozsahu od 1 do 9 999, přičemž výchozí hodnota je 200.

MQ Appliance Pro IBM MQ Applianceje výchozí hodnota 999 999 999 a neměla by se měnit. Chcete-li omezit maximální počet kanálů klienta, použijte k definování limitů pro každý kanál SVRCONN atributy MAXINST a MAXINSTC v definicích kanálu SVRCONN, viz téma Konfigurace správce front v produktu IBM MQ Appliance v dokumentaci produktu IBM MQ Appliance .

Systém, který je zaneprázdněn obsluhováním připojení ze sítě, může vyžadovat vyšší číslo než výchozí nastavení. Určete hodnotu, která je správná pro vaše prostředí, v ideálním případě pozorováním chování vašeho systému během testování.

Pro všechny ostatní platformy je výchozí hodnota 100. Můžete nastavit **MaxChannels** na jinou hodnotu, abyste omezili maximální počet aktuálních kanálů, je-li to potřeba.

Argument **MaxChannels** je atribut správce front pouze v systému z/OS . Na ostatních platformách **MaxChannels** je atribut v souboru `qm.ini` . Informace o tom, jak nastavíte atribut **MaxChannels** na jiných platformách, najdete v tématu Stanzy konfiguračního souboru pro distribuované řazení do 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 jedna úloha používat souběžně. Každé úspěšné volání MQOPEN pro jednu frontu (nebo pro objekt, který není frontou) používá jeden popisovač. Tento popisovač bude k dispozici pro opětovné použití, když je objekt uzavřen. Když je však otevřen distribuční seznam, každá fronta v rozdělovníku je alokována jako samostatná obsluha, takže volání MQOPEN používá tolik popisovačů, kolik je ve frontách v rozdělovníku. To musí být vzato v úvahu při rozhodování o vhodné hodnotě pro *MaxHandles*.

Volání MQPUT1 provádí volání MQOPEN jako součást jeho zpracování; v důsledku toho hodnota MQPUT1 používá tolik obslužných rutin jako MQOPEN, ale manipulátory jsou použity pouze po dobu trvání volání MQPUT1 .

V produktu z/OSse *úlohou* rozumí úloha CICS , úloha MVS nebo závislý region IMS .

Hodnota je v rozsahu od 1 do 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_HANDLES s voláním MQINQ.

MaxMsgDélka (MQLONG)

Toto je délka nejdelší *fyzické* zprávy, kterou může správce front zpracovat. Vzhledem k tomu, že atribut správce front produktu **MaxMsgLength** lze nastavit nezávisle na atributu fronty produktu **MaxMsgLength** , je tato nejdelší fyzická zpráva, kterou lze umístit do fronty, menší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, může aplikace vložit logickou zprávu, která je delší než menší z hodnot dvou atributů **MaxMsgLength** , ale pouze v případě, že aplikace určuje příznak MQMF_SEGMENTATION_ALLOWED v deskriptoru MQMD. Je-li tento parametr zadán, horní mez pro délku logické zprávy je 999 999 999 bajtů, ale obvykle omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, výsledkem je nižší mezní hodnota.

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.

Délka MaxProperties(MQLONG)

Používá se k řízení velikosti vlastností, které mohou tékat se zprávou. To zahrnuje jak název vlastnosti v bajtech, tak i velikost hodnoty vlastnosti také v bajtech.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_PROPERTIES_LENGTH s voláním MQINQ.

Multi V 9.2.0 *Velikost MaxQFile(MQLONG)*

Maximální velikost (v megabajtech), do které může růst soubor fronty.

Tabulka 561. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Je možné, že soubor fronty překročí maximální velikost, je-li konfigurován na hodnotu nižší, než je aktuální velikost souboru fronty. Pokud k tomu dojde, nebude již soubor fronty přijímat nové zprávy, ale umožní spotřebování existujících zpráv. Pokud velikost souboru fronty klesla pod konfigurovanou hodnotu, lze do fronty vkládat nové zprávy.

Poznámka: Tento obrázek se může lišit od hodnoty atributu konfigurovaného ve frontě, protože interně správce front může potřebovat použít větší velikost bloku k dosažení zvolené velikosti. Další informace o změně velikosti souborů fronty a velikosti bloků a granularity naleznete v tématu Úprava souborů fronty produktu IBM MQ .

Pokud se granularita potřebuje změnit, protože tento atribut byl zvýšen, je do protokolů AMQERR zapsána varovná zpráva AMQ7493W `Granularita změněna` . To vám dává indikaci, že je třeba naplánovat vyprázdnění fronty, aby produkt IBM MQ přijal 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 fronty s granularitou, která se rovná hodnotě 512.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_Q_FILE_SIZE s voláním MQINQ.

MaxUncommittedzprávy (MQLONG)

Toto je maximální počet nepotvrzených zpráv, které mohou existovat v rámci jednotky práce. Počet nepotvrzených zpráv je součtem následujících od začátku aktuální transakce:

- Zprávy vkládané aplikací s volbou MQPMO_SYNCPOINT
- Zprávy načtené aplikací s volbou MQGMO_SYNCPOINT
- Zprávy spouštěče a zprávy sestav COA generované správcem front pro zprávy vkládané do volby MQPMO_SYNCPOINT
- Zprávy COD zprávy generované správcem front pro zprávy načtené pomocí volby MQGMO_SYNCPOINT

Následující zprávy se nepočítají jako nepotvrzené:

- Zprávy vkládané nebo načtené aplikací mimo jednotku práce
- Zprávy spouštěče nebo zprávy sestav COA/COD generované správcem front jako výsledek zpráv vložených nebo načtených mimo jednotku práce
- Zprávy oznámení o vypršení platnosti generované správcem front (i v případě, že volání způsobující vypršení zprávy o vypršení platnosti zprávy MQGMO_SYNCPOINT)
- Zprávy událostí generované správcem front (i přesto, že volání způsobující zprávu události MQPMO_SYNCPOINT nebo MQGMO_SYNCPOINT)

Poznámka:

1. Zprávy o výjimkách jsou generovány agentem MCA (Message Channel Agent) nebo aplikací a jsou zpracovávány stejným způsobem jako běžné zprávy, které byly aplikací vloženy nebo načítány.

2. Když je zpráva nebo segment vložen s volbou MQPMO_SYNCPOINT, počet nepotvrzených zpráv se zvýší o jednu, bez ohledu na to, kolik fyzických zpráv ve skutečnosti pochází z vložení. (Může nastat více než jedna fyzická zpráva, pokud musí správce front rozdělit zprávu nebo segment.)
3. Je-li distribuční seznam vložen s volbou MQPMO_SYNCPOINT, počet nepotvrzených zpráv se zvýší o jednu *pro každou vygenerovanou fyzickou zprávu*. Může být tak malý jako jeden nebo velký jako počet míst určení v rozdělovníku.

Spodní 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)

Tento příkaz řídí shromažďování informací evidence pro data MQI.

Hodnota je jedna z následujících možností:

MQMON_ON
Shromažďovat data evidence rozhraní API.

MQMON_OFF
Neshromažďovat data evidence rozhraní API. Toto je výchozí hodnota.

Nastavíte-li atribut ACCTCONO správce front na hodnotu ENABLED, může být tato hodnota přepsána pro jednotlivá připojení pomocí pole Volby ve struktuře MQCNO. Změny této hodnoty jsou platné pouze pro připojení ke správci front, k nimž došlo po změně atributu.

Tento atribut je podporován pouze v systému Multiplatforms.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_MQI s voláním MQINQ.

MQIStatistics (MQLONG)

Tento ovládací prvek řídí kolekci informací o monitorování statistiky pro správce front.

Hodnota je jedna z následujících možností:

MQMON_ON
Shromažďovat statistiku modulu MQI.

MQMON_OFF
Neshromažďovat statistiku MQI. Toto je výchozí hodnota.

Tento atribut je podporován pouze v 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 jehož uplynutí může správce front automaticky odebrat značku z procházení zpráv.

Tento atribut popisuje časový interval, po který se očekává, že zprávy, které byly označeny jako procházené voláním MQGET za použití volby get message MQGMO_MARK_BROWSE_CO_OP, budou nadále označeny jako procházené.

Správce front může automaticky zrušit označení procházených zpráv, které byly označeny jako zkontrolované pro spolupracující sadu manipulátorů, pokud byly označeny pro více než tento přibližný interval.

To nemá vliv na stav žádné zprávy označené jako procházení, které bylo získáno voláním MQGET, pomocí 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 pod výchozí hodnotou 5000.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MSG_MARK_BIL_INTERVAL s voláním MQINQ.

OutboundPortMaximum (MQLONG)

Jedná se o nejvyšší číslo portu v rozsahu, který je definován hodnotou OutboundPortMin a OutboundPortMax, čísla portů, která mají být použita pro vazbu odchozích kanálů.

Hodnota je celé číslo v rozsahu 0 až 65535 a musí být rovno nebo větší než hodnota OutboundPortMin hodnota. Výchozí hodnota je 0.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OUTBOUND_PORT_MAX s voláním MQINQ.

OutboundPortMin (MQLONG)

Jde o nejnižší číslo portu v rozsahu, který je definován hodnotou OutboundPortMin a OutboundPortMax, čísla portů, která mají být použita pro vazbu odchozích kanálů.

Hodnota je celé číslo v rozsahu od 0 do 65535 a musí být rovny nebo menší než maximální hodnota OutboundPort. Výchozí hodnota je 0.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OUTBOUND_PORT_MIN s voláním MQINQ.

PerformanceEvent (MQLONG)

Tento ovládací prvek řídí, zda jsou generovány události související s výkonem. Je to jedna z následujících hodnot:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PERFORMANCE_EVENT s voláním MQINQ.

Platforma (MQLONG)

Označuje operační systém, na kterém je spuštěný správce front:

MQPL_AIX
AIX (stejná hodnota jako MQPL_UNIX).

ZAŘÍZENÍ 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.

POČ MQPL_WINDOWS_NT
Windows .

NEZOS_MQPL_
z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PLATFORM s voláním MQINQ.

PubSubNPInputMsg (MQLONG)
Zda se má vyřadit nebo uchovat nedoručenou vstupní zprávu.

Hodnota je jedna z následujících možností:

NEDORUČENOVANÉ_ZAHOZENÍ
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 publikování/odběru ve frontě pokračovat v opakování procesu v přiměřených intervalech a nebude pokračovat ve zpracování následujících zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_NP_MSG s voláním MQINQ.

PubSubNPResponse (MQLONG)
Řídí chování nedoručených zpráv s odpovědí.

Hodnota je jedna z následujících možností:

MQUNDELIVERED_NORMAL
Netrvalé odpovědi, které nelze umístit do fronty odpovědí, jsou umístěny do fronty nedoručených zpráv, pokud nemohou být umístěny do fronty nedoručených zpráv, budou zrušeny.

MQUNDELIVERED_SAFE
Netrvalé odpovědi, které nelze umístit do fronty odpovědí, jsou umístěny do fronty zablokovaných zpráv (DLQ). Pokud nelze nastavit odezvu a nelze ji umístit na frontu nedoručených zpráv, rozhraní publikování/odběru ve frontě vrátí aktuální operaci a pak se pokusí znovu v příslušných intervalech a nebude pokračovat ve zpracování následujících zpráv.

NEDORUČENOVANÉ_ZAHOZENÍ
Netrvalé odpovědi nejsou umístěny do fronty odpovědí jsou zrušeny.

Jedná se o výchozí hodnotu pro nové správce front.

MQUNDELIVERED_KEEP
Netrvalé odpovědi nejsou umístěny do fronty nedoručených zpráv nebo zahozeny. Místo toho bude rozhraní publikování/odběru ve frontě zazálohovat aktuální operaci a poté ji v příslušných intervalech zopakovat.

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 IBM MQ V6.0, počáteční hodnota tohoto atributu závisí na hodnotách *DiscardNonPersistentResponse* a *DLQNonPersistentResponse* před migrací, jak je zobrazeno v následující tabulce.

		Odpověď DLQNonPersistent		
		Ano	**Ne**	**Nenastaveno**
DiscardNonPersistentResponse	**Ano**	MQUNDELIVERED_NORMAL	NEDORUČENOVANÉ_ZAHOZENÍ	MQUNDELIVERED_NORMAL
	Ne	MQUNDELIVERED_SAFE	MQUNDELIVERED_KEEP	MQUNDELIVERED_SAFE
	Nenastaveno	Pokud SyncPointPersistent = No, MQUNDELIVERED_SAFE else MQUNDELIVERED_NORMAL	Pokud SyncPointPersistent = No, MQUNDELIVERED_KEEP else MQUNDELIVERED_DISCARD	Pokud SyncPointPersistent = No, MQUNDELIVERED_SAFE else MQUNDELIVERED_NORMAL

PubSubMaxMsgRetryCount (MQLONG)
Počet opakovaných pokusů při zpracování zprávy příkazu se selháním pod synchronizačním bodem.

Hodnota je jedna z následujících možností:

0 - 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)
Určuje, zda jsou v rámci synchronizačního bodu zpracovány pouze trvalé zprávy nebo všechny zprávy.

Hodnota je jedna z následujících možností:

MQSYNCPOINT_IFPER
Díky tomu bude rozhraní publikování/odběru ve frontě přijímat netrvalé 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
Díky tomu bude rozhraní publikování/odběru ve frontě přijímat všechny zprávy pod synchronizačním bodem.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_PUBSUB_SYNC_PT s voláním MQINQ.

Režim PubSub(MQLONG)
Určuje, zda je stroj pro publikování/odběr a rozhraní publikování/odběru zařazené do fronty spuštěné, a proto umožňuje aplikacím publikovat/přihlásit se k odběru prostřednictvím rozhraní API a front, které jsou monitorovány rozhraním publikování/odběru ve frontě.

Hodnota je jedna z následujících možností:

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 se žádná zpráva, která je vložena do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, nepostupuje. Toto nastavení se používá pro kompatibilitu s WebSphere Message Broker V6 nebo staršími verzemi pomocí tohoto správce front, protože musí číst stejné fronty, ze kterých normálně čte rozhraní publikování/odběru ve frontě.

MQPSM_DISABLED
Stroj pro publikování/odběr a rozhraní pro publikování/odběr ve frontě nejsou spuštěny. Proto není možné publikovat/přihlásit se k odběru pomocí rozhraní API. Jakékoli zprávy publish/subscribe, které jsou vloženy do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, nepracují.

MQPSM_ENABLED
Stroj publikování/odběru a rozhraní publikování/odběru ve frontě jsou spuštěny. Proto je možné publikovat/přihlásit se k odběru pomocí rozhraní API a front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě. 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 slouží k popisu komentáře popisujícího správce front. Obsah tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace DBCS může toto pole obsahovat znaky DBCS (s výhradou maximální délky 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.

- V systému z/OSje výchozí hodnotou název produktu a číslo verze.
- Ve všech ostatních prostředích jsou 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 klientů IBM MQ 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 MQMD- MsgId field). Správci front, kteří mohou komunikovat, musí mít proto názvy, které se v prvních 12 znacích liší, aby identifikátory zpráv byly jedinečné v síti správce front.

V systému z/OSje název shodný s názvem 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.

QSGName (MQCHAR4)

Jedná se o název skupiny sdílení front, do níž patří lokální správce front. Pokud lokální správce front nepatří do skupiny sdílení front, je tento 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 v systému z/OS.

QueueAccounting (MQLONG)

Tím se řídí kolekce informací o účtování pro fronty.

Hodnota je jedna z následujících možností:

MQMON_NONE
Neshromažďovat data evidence pro fronty, bez ohledu na nastavení atributu evidence fronty ACCTQ. Toto je výchozí hodnota.

MQMON_OFF
Neshromažďovat účtovací data pro fronty, které v atributu fronty ACCTQ uvádí QMGR.

MQMON_ON
Shromážděte data evidence pro fronty, které určují QMGR v atributu fronty ACCTQ.

Změny této hodnoty jsou platné pouze pro připojení ke správci front, k nimž došlo po změně atributu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACCOUNTING_Q s voláním MQINQ.

QueueMonitoring (MQLONG)
Tato hodnota určuje výchozí nastavení pro online monitorování front.

Je-li atribut fronty **QueueMonitoring** nastaven na hodnotu MQMON_Q_MGR, určuje tento atribut hodnotu, kterou kanál předpokládá. Hodnota může být následující:

MQMON_OFF
Shromažďování online monitorování dat je vypnuto. Jedná se o počáteční výchozí hodnotu správce front.

MQMON_NONE
Shromažďování online monitorování dat je vypnuto pro fronty bez ohledu na nastavení jejich atributu **QueueMonitoring** .

MQMON_LOW
Shromažďování online monitorování dat je zapnuto, s nízkým poměrem shromažďování dat.

MQMON_MEDIUM
Shromažďování online monitorování dat je zapnuto, se středním poměrem shromažďování dat.

MQMON_HIGH
Shromažďování online monitorování dat je zapnuto, s vysokým poměrem shromažďování dat.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MONITORING_Q s voláním MQINQ.

QueueStatistics (MQLONG)

Tento ovládací prvek řídí shromažďování statistických dat pro fronty.

Je to jedna z následujících hodnot:

MQMON_NONE
Neshromažďovat statistiky fronty pro fronty, bez ohledu na nastavení atributu fronty produktu **QueueStatistics** . Toto je výchozí hodnota.

MQMON_OFF
Neshromažďovat statistická data pro fronty, které specifikují správce front v atributu fronty **QueueStatistics** .

MQMON_ON
Shromážděte statistické údaje pro fronty, které určují správce front v atributu fronty produktu **QueueStatistics** .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_Q s voláním MQINQ.

ReceiveTimeout (MQLONG)
Uvádí, jak dlouho kanál TCP/IP čeká na příjem dat, včetně synchronizačních signálů, od svého partnera, než se vrátí do neaktivního stavu. Vztahuje se pouze na kanály zpráv a nikoli na kanály MQI.

Přesný význam parametru ReceiveTimeout se mění podle hodnoty uvedené v poli ReceiveTimeoutType. Typ ReceiveTimeoutTyp může být nastaven na jednu z následujících možností:

- MQRCVTIME_EQUAL-tato hodnota je číslo v sekundách, po který má kanál čekat. Uveďte hodnotu v rozsahu od 0 do 999999.
- MQRCVTIME_ADD-Tato hodnota je počet sekund, které se mají přidat do vyjednaného adaptéru HBINT a určuje, jak dlouho bude kanál čekat. Uveďte hodnotu v rozsahu od 1 do 999999.

- MQRCVTIME_MULTIPLY-Tato hodnota je multiplikátorem, která se má použít u vyjednaného HBINT. Uveďte hodnotu 0 nebo hodnotu v rozsahu 2-99.

Výchozí hodnota je 0.

Nastavte typ ReceiveTimeoutna hodnotu MQRCVTIME_MULTIPLY nebo MQRCVTIME_EQUAL a ReceiveTimeout na hodnotu 0, chcete-li kanál zastavit z vypršení časového limitu čekání na příjem dat od příslušného partnera.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT s voláním MQINQ.

Minimální hodnota ReceiveTimeoutMin (MQLONG)

Toto je minimální doba, v sekundách, po kterou kanál TCP/IP čeká na příjem dat, včetně synchronizačních 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.

Použijete-li typ ReceiveTimeoutk uvedení, že doba čekání kanálu TCP/IP má být vypočtena relativně k vyjednané hodnotě HBINT, a výsledná hodnota je menší než hodnota tohoto parametru, bude použita tato hodnota.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT_MIN s voláním MQINQ.

Typ ReceiveTimeoutType (MQLONG)

Toto je kvalifikátor, který se použije na ReceiveTimeout , aby definoval, jak dlouho kanál TCP/IP čeká na příjem dat, včetně synchronizačních 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 možností:

MQRCVTIME_MULTIPLY
ReceiveTimeout je multiplikátor, který se použije na vyjednanou hodnotu HBINT, aby určil, jak dlouho kanál čeká. Toto je výchozí hodnota.

MQRCVTIME_ADD
ReceiveTimeout je hodnota (v sekundách), která se má přidat k vyjednané hodnotě HBINT, aby určil, jak dlouho kanál čeká.

MQRCVTIME_EQUAL
Hodnota ReceiveTimeout je hodnota, v sekundách, po kterou kanál čeká.

Chcete-li zastavit čekání kanálu na vypršení časového limitu čekání na příjem dat od partnera, nastavte parametr ReceiveTimeoutna hodnotu MQRCVTIME_MULTIPLY nebo MQRCVTIME_EQUAL a ReceiveTimeout na hodnotu 0.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RECEIVE_TIMEOUT_TYPE s voláním MQINQ.

RemoteEvent (MQLONG)

Tento ovládací prvek určuje, zda jsou generovány události vzdálené chyby. Je to jedna z následujících hodnot:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu 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, *RepositoryNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a *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, objekt seznamu názvů obsahuje pouze jedno jméno. Alternativně lze *RepositoryName* použít k uvedení 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)

Uvádí, zda správce front podporuje názvy profilů zabezpečení ve smíšených případech nebo pouze velkými písmeny.

Hodnota je jedna z následujících možností:

MQSCYC_UPPER
Názvy profilů zabezpečení musí být velkými písmeny.

MQSCYC_SMÍŠENÝ
Názvy profilů zabezpečení mohou být velkými písmeny nebo velkými i malými písmeny.

Změny tohoto atributu se projeví, když je spuštěn příkaz Obnovit zabezpečení s uvedeným *SecurityType(MQSECTYPE_CLASSES)* .

z/OS Tento atribut je podporován pouze v systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SECURITY_CASE s voláním MQINQ.

Název SharedQMgr(MQLONG)

Určuje, zda má být příkaz *ObjectQmgrName* použit nebo považován za lokálního správce front v rámci volání MQOPEN pro sdílenou frontu v případě, že *ObjectQmgrName* je jiným správcem front v rámci skupiny sdílení front.

Hodnota může být některá z následujících:

MQSQQM_USE
ObjectQmgrName se používá a je otevřena příslušná přenosová fronta.

MQSQQM_IGNORE
Je-li cílová fronta sdílena a *ObjectQmgrName* je fronta správce front ve stejné skupině sdílení front, operace otevření se provede lokálně.

Tento atribut je platný pouze na z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SHARED_Q_MGR_NAME s voláním MQINQ.

SPLCAP
Označuje, zda jsou funkce zabezpečení produktu Advanced Message Security dostupné pro správce front.

PODPOROVANÁ MQCAP_
Jedná se o výchozí hodnotu, pokud je komponenta AMS nainstalována pro instalaci, pod kterou je spuštěn správce front.

PODPOROVÁNO MQCAP_NOT_SUPPORTED

SSLEvent (MQLONG)
Uvádí, zda jsou generovány události TLS.

Je to jedna z následujících hodnot:

POVOLENÝ MQEVR_
Vygenerujte události TLS následujícím způsobem:

MQRC_CHANNEL_SSL_ERROR

MQEV_DISABLED
Negenerovat události TLS; jedná se o výchozí hodnotu.

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" . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C certificate a měli by si být vědomi jakýchkoli 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 SSLKeyResetCount (MQLONG)
Určuje, kdy mají být agenti kanálu zpráv kanálu TLS (MCA), kteří iniciují komunikaci, resetovali 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 agentem MCA.

Hodnota je číslo v rozsahu od 0 do 999 999 999, přičemž výchozí hodnota je 0. Určíte-li počet obnovení tajných klíčů TLS v rozsahu od 1 bajtu do 32 KB, budou kanály TLS používat počet obnovení tajných klíčů

32 KB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se mohly vyskytnout u malých hodnot resetu tajného klíče TLS.

Tajný klíč se znovu vyjednává, když celkový počet nezašifrovaných bajtů odeslaných a přijatých inicializací MCA kanálu překročí uvedenou hodnotu. Jsou-li povoleny prezenční signály kanálu, je tajný klíč znovu vyjednán před odesláním nebo přijetím dat po synchronizačním signálu kanálu, nebo pokud celkový počet nezašifrovaných bajtů překročí zadanou hodnotu, podle toho, co nastane dříve.

Počet bajtů odeslaných a přijatých pro opětovné vyjednávání zahrnuje informace o řízení odeslané a přijaté kanálem MCA kanálu a je resetován, kdykoli dojde k opětovnému vyjednávání.

Použijte hodnotu 0, abyste označili, že tajné klíče nejsou nikdy znovu vyjednávány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_RESET_COUNT s voláním MQINQ.

Událost StartStop(MQLONG)

Tento ovládací prvek řídí, zda jsou generovány události spuštění a zastavení. Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_START_STOP_EVENT s voláním MQINQ.

StatisticsInterval (MQLONG)

Určuje, jak často (v sekundách) mají být zapsána data monitorování statistiky do fronty monitorování.

Hodnota je celé číslo v rozsahu od 0 do 604800, s výchozí hodnotou 1800 (30 minut).

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_STATISTICS_INTERVAL s voláním MQINQ.

SyncPoint (MQLONG)

To označuje, zda lokální správce front podporuje jednotky práce a syncpointing s voláními MQGET, MQPUT a MQPUT1 .

MQSP_AVAILABLE
Jednotky práce a syncpointing jsou k dispozici.

MQSP_NOT_AVAILABLE
Jednotky práce a syncpointing nejsou k dispozici.

- V případě z/OS se tato hodnota nikdy nevrátí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SYNCPOINT s voláním MQINQ.

TCPChannels (MQLONG)

Jedná se o maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni a které používají přenosový protokol TCP/IP.

Hodnota musí být v rozsahu 0 až 9999, přičemž výchozí hodnota je 200. Pokud uvedete 0, TCP/IP se nepoužije.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_CHANNELS s voláním MQINQ.

TCPKeepAlive (MQLONG)

Uvádí, zda použít TCP KEEPALIVE pro kontrolu toho, zda je druhý konec připojení stále dostupný. Jestliže není k dispozici, je kanál uzavřen.

Hodnota je jedna z následujících možností:

MQTCPKEEP_YES
Použijte TCP KEEPALIVE, jak je uvedeno v datové sadě konfigurace profilu TCP. Uvedete-li atribut kanálu KeepAliveInterval (KAINT), použije se hodnota, na kterou se sada používá.

MQTCPKEEP_NO
Nepoužívejte TCP KEEPALIVE. Toto je výchozí hodnota.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_KEEP_ALIVE s voláním MQINQ.

TCPName (MQCHAR8)

Jedná se o název buď jediné nebo preferované sady protokolů TCP/IP, která bude použita, v závislosti na hodnotě TCPStackType. Tento parametr lze použít pouze v prostředí s více zásobníky CINET. Výchozí hodnota je TCPIP.

Tento atribut je podporován pouze pro 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)

Uvádí, zda iniciátor kanálu může použít pouze zásobník TCP/IP uvedený v TCPName nebo může být volitelně svázán s libovolným vybraným zásobníkem TCP/IP. Tento parametr lze použít pouze v prostředí s více zásobníky CINET.

Hodnota je jedna z následujících možností:

MQTCPSTACK_SINGLE
Inicializátor kanálu může použít pouze adresní prostory TCP/IP pojmenované v TCPName. Toto je výchozí hodnota.

MQTCPSTACK_MULTIPLE
Inicializátor kanálu může použít jakýkoli adresní prostor TCP/IP, který má k dispozici. Výchozí hodnota je uvedena v poli TCPName, pokud není pro kanál nebo modul listener zadána žádná jiná hodnota.

Tento atribut je podporován pouze pro z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_STACK_TYPE s voláním MQINQ.

Zaznamenávání TraceRoute(MQLONG)

Tento ovládací prvek řídí záznam informací o přenosové cestě trasování.

Hodnota je jedna z následujících možností:

MQRECORDING_DISABLED
Není povoleno žádné připojení zpráv trasování cesty.

MQRECORDING_Q
Umístit zprávy trasování do pevné pojmenované fronty.

ZPRÁVA MQRECORDING_MSG
Umístěte zprávy trasování přenosové cesty do fronty určené pomocí samotné zprávy. 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), který se používá k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je MQTT_FIRST. V tomto případě se zprávy spouštěče obvykle generují pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých

okolností lze však vygenerovat dodatečnou zprávu spouštěče s parametrem MQTT_FIRST, a to i v případě, že fronta nebyla prázdná. Tyto další zprávy triggeru se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění najdete v tématu Spouštění kanálů.

Hodnota není menší než 0 a není větší než 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), který se používá k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je MQTT_FIRST. V tomto případě se zprávy spouštěče obvykle generují pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých okolností lze však vygenerovat dodatečnou zprávu spouštěče s parametrem MQTT_FIRST, a to i v případě, že fronta nebyla prázdná. Tyto další zprávy triggeru se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění najdete v tématu Spouštění kanálů.

Hodnota není menší než 0 a není větší než 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)

Jedná se o verzi kódu IBM MQ jako VVRRMMFF, kde:

VV-Verze

RR-Vydání

MM-Úroveň údržby

FF-Úroveň opravy

XrCapability(MQLONG)

Tento parametr určuje, zda správce front podporuje příkazy produktu MQ Telemetry .

Hodnota je jedna z následujících možností:

PODPOROVANÁ MQCAP_
Jsou podporovány příkazy produktu MQ Telemetry a jsou podporovány příkazy Telemetry.

PODPOROVÁNO MQCAP_NOT_SUPPORTED
Komponenta produktu MQ Telemetry není nainstalována.

Tento atribut je podporován pouze v 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. Ostatní atributy fronty se vztahují pouze na určité typy front.

Typy front

Správce front podporuje následující typy definic front:

Lokální fronta
Zprávy můžete uložit 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 na serveru 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, která je vlastníkem sdíleného úložiště.

Aplikace připojené k libovolnému správci front ve skupině sdílení front mohou umisťovat zprávy do front tohoto typu a odebírat zprávy z fronty tohoto typu. Takové fronty jsou ve skutečnosti stejné jako lokální fronty. Hodnota atributu fronty **QType** je `MQQT_LOCAL`.

Aplikace připojené k lokálnímu správci front mohou umísťovat zprávy a odebírat zprávy z front tohoto typu. Hodnota atributu fronty **QType** je `MQQT_LOCAL`.

Fronta klastru

Zprávy ve frontě klastru můžete uložit ve správci front, kde je definován. 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 říkají lokálnímu správci front, jak 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 odebrat zprávy ze vzdálených front. Hodnota atributu fronty **QType** je `MQQT_REMOTE`.

Také můžete použít definici vzdálené fronty pro:

- Aliasy fronty odpovědí

V tomto případě je název definice názvem fronty pro odpověď. Další informace naleznete v tématu Aliasy fronty odpovědí a klastry.

- Aliasy správce front

  V tomto případě je název definice alias pro správce front, nikoli název fronty. Další informace naleznete v tématu Aliasy správce front a klastry.

Fronta aliasů
Nejedná se o fyzickou frontu; jedná se o alternativní název pro lokální frontu, sdílenou frontu, frontu klastru nebo vzdálenou frontu. Název fronty, do níž je rozlišen alias, je součástí definice alias fronty.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front tohoto typu; zprávy jsou umístěny do fronty, na kterou je alias interpretováno. Aplikace mohou odebírat 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
Toto není fyzická fronta; je to sada atributů fronty, ze které lze vytvořit lokální frontu.

Zprávy nemohou být uloženy ve frontách tohoto typu.

limity front

V 9.2.0

V produktu IBM MQ 9.2.0máte možnost konfigurace a monitorování front, které budou výrazně podporovat více než dva terabajtové výchozí limity použité v dřívějších vydáních produktu IBM MQ. Také máte možnost snížit velikost souboru fronty, který může růst.

Chcete-li povolit konfiguraci front, můžete použít atribut **MAXFSIZE** na lokálních a modelových frontách a monitorovat fronty, můžete použít atributy stavu fronty **CURFSIZE** a **CURMAXFS**.

Další informace naleznete v tématu Úprava souborů fronty produktu IBM MQ.

Atributy fronty

Některé atributy fronty platí pro všechny typy front. Ostatní atributy fronty se vztahují pouze na určité typy front. Typy front, na které se atribut vztahuje, jsou zobrazeny v Tabulka 562 na stránce 830 a v následných tabulkách.

Tabulka 562 na stránce 830 shrnuje atributy, které jsou specifické pro fronty. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů zobrazené v této sekci jsou popisné názvy použité spolu s voláními MQINQ a MQSET ; názvy jsou stejné jako u příkazů PCF. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; podrobnosti najdete v příkazech MQSC .

V následující tabulce se sloupce použijí takto:

- Sloupec pro lokální fronty platí také pro sdílené fronty.
- Sloupec pro modelové fronty označuje, které atributy jsou děděny lokální frontou vytvořenou z modelové fronty.
- Sloupec pro fronty klastru označuje atributy, které mohou být dotazovány, když je fronta klastru otevřena pro dotaz samostatně nebo pro zjištění a výstup. Jsou-li dotazovány jakékoli jiné atributy, volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_SELECTOR_NOT_FOR_TYPE (2068).

  Je-li fronta klastru otevřena pro dotaz s jedním nebo více vstupními, procházením nebo sadou, použije se místo toho sloupec pro lokální fronty.

  Je-li fronta klastru otevřena pro dotaz samostatně nebo pro zjištění a výstup a zadání názvu základního správce front, použije se místo toho sloupec pro lokální fronty.

Tabulka 562. 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 vrácených zpráv	X	X			
BackoutThreshold	Práh vrácení	X	X			
BaseQName	Název fronty, na kterou se rozlišuje alias			X		
CFStrucName	Název struktury prostředku Coupling Facility	X	X			
CLCHNAME	Názvy odesílacích kanálů klastru	✓	✓			
ClusterName	Název klastru, do kterého fronta patří	X		X	X	X
ClusterNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, do kterých fronta patří	X		X	X	
CLWLQueuePriority	Priorita fronty pracovní zátěže klastru	X		X	X	X
CLWLQueueRank	Hodnocení fronty 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 562. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
NonPersistentMessage Class	Cíl spolehlivosti pro netrvalé 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 vysoké hloubky fronty	X	X			
QDepthHighLimit	Horní mez hloubky fronty	X	X			
QDepthLowEvent attribute	Zda se generují události nízké hloubky fronty	X	X			
QDepthLowLimit attribute	Dolní mez hloubky fronty	X	X			
QDepthMaxEvent	Zda se generují 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 se generují události servisního intervalu vysoké nebo servisní interval OK	X	X			
QSGDisp attribute	Dispozice skupiny sdílení front	X		X	X	
QueueAccounting	Shromažďování dat evidence front	X	X	X	X	X
QueueMonitoring	Online monitorování dat pro fronty	X	✓			
QueueStatistics	Kolekce 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 také existuje v adresáři buňky	X		X	X	
Shareability	Možnost sdílení front	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 se rozhodnout, jaký typ přenosové fronty klastru použít

AlterationDate (MQCHAR12)

Datum, kdy byla definice naposledy změněna.

Tabulka 563. Typy front, na které se vztahuje tento atribut

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 jeden prázdný znak).

Hodnoty určitých atributů (například *CurrentQDepth*) se mění s tím, jak pracuje správce front. Změny těchto atributů nemají vliv na *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 564. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Jedná se o čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS pomocí 24hodinového formátu, s počáteční nulou, je-li hodina menší než 10 (například 09.10.20).

- V systému z/OSje čas GMT (Greenwich Mean Time), kdy se časová základna systému nastavuje přesně na GMT.
- V jiných prostředích je čas místní čas.

Hodnoty určitých atributů (například *CurrentQDepth*) se mění s tím, jak pracuje správce front. Změny těchto atributů nemají vliv na *AlterationTime*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_ALTERATION_TIME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TIME_LENGTH.

BackoutRequeue(MQCHAR48)

Jedná se o nadměrný název fronty vrácených zpráv. Kromě možnosti dotazování na hodnotu, která má být dotazována, nepodniká správce front žádnou akci založenou na hodnotě tohoto atributu.

Tabulka 565. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace spuštěné uvnitř WebSphere Application Server a ty, které používají IBM MQ Application Server Facilities, používají tento atribut k určení toho, kam se mají vrátit zprávy, které byly vráceny. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu.

IBM MQ classes for JMS používá tento atribut k určení, kam se má přenést zpráva, která již byla vrácena, maximální počet, kolikrát je zadán atributem *BackoutThreshold* .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_BACOUT_REQ_Q_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

BackoutThreshold (MQLONG)

Jedná se o práh vrácení. Kromě možnosti dotazování na hodnotu, která má být dotazována, nepodniká správce front žádnou akci založenou na hodnotě tohoto atributu.

Tabulka 566. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace běžící uvnitř portálu WebSphere Application Server a ty, které používají IBM MQ Application Server Facilities, použijí tento atribut k určení, zda by se měla zpráva zazálohovat. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu.

IBM MQ classes for JMS používá tento atribut k určení, kolikrát se má před přenosem zprávy do fronty zadané atributem *BackoutRequeueQName* povolit, aby byla zpráva vrácena zpět.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_BACKOUT_THRESHOLD spolu s voláním MQINQ.

BaseQName (MQCHAR48)

Jedná se o název fronty, která je definována pro lokálního správce front.

Tabulka 567. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
		X		

(Další informace o názvech front naleznete v tématu MQOD- ObjectName.) Fronta je jedním z následujících typů:

MQQ_LOCAL
Lokální fronta.

MQQT_REMOTE
Lokální definice vzdálené fronty.

KLASTR MQQ_CLUSTER
Fronta klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_BASE_Q_NAME spolu s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_NAME_LENGTH.

BaseType (MQCFIN)

Typ objektu, na který je alias vyřešen.

Tabulka 568. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
		X		

Je to jedna 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, ve které jsou uloženy zprávy ve frontě. První znak jména je v rozsahu A až Z a zbývající znaky jsou v rozsahu A až Z, 0 až 9, nebo prázdné.

Tabulka 569. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Chcete-li získat úplný název struktury ve spojovacím zařízení, zadejte hodnotu atributu správce front produktu **QSGName** s hodnotou atributu fronty produktu **CFStrucName** .

Tento atribut se používá pouze pro 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 v systému z/OS.

Název ClusterChannel(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 570. Typy front, na které se vztahuje tento atribut

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, do kterého fronta patří.

Tabulka 571. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Pokud fronta patří do více než jednoho klastru, *ClusterNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a *ClusterName* je prázdný. 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 572. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Pokud fronta náleží pouze jednomu klastru, objekt seznamu názvů obsahuje pouze jeden název. Alternativně lze *ClusterName* použít k uvedení 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, hodnotu v rozsahu 0 až 9 představující prioritu fronty.

Tabulka 573. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Další informace najdete v tématu Fronty klastru.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CLWL_Q_PRIORITY s voláním MQINQ.

CLWLQueueRank (MQLONG)

Jedná se o očíslování pořadí fronty pracovní zátěže klastru, hodnoty v rozsahu 0 až 9 představující úroveň řazení fronty.

Tabulka 574. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Další informace najdete 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, pokud má cílová fronta jak lokální instanci, tak alespoň jednu vzdálenou instanci klastru. Tento atribut se nepoužije v případě, že je zdrojem operace vložení kanál klastru.

Tabulka 575. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X				

Hodnota je jedna z následujících možností:

MQCLWL_USEQ_ANY
Použít vzdálené a lokální fronty.

MQCLWL_USEQ_LOCAL
Nepoužívejte vzdálené fronty.

MQCLWL_USEQ_AS_Q_MGR
Zdědit definici z MQIA_CLWL_USEQ správce front.

Další informace najdete 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 576. Typy front, na které se vztahuje tento atribut

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¬¬, ¬ představuje jeden prázdný znak).

- V systému IBM ise datum vytvoření fronty může lišit od data vytvoření příslušné entity operačního systému (soubor nebo uživatelská prostor), která představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_CREATION_DATE s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CRECREATION_DATE_LENGTH.

CreationTime (MQCHAR8)
Toto je čas, kdy byla fronta vytvořena.

Tabulka 577. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X				

Formát času je HH.MM.SS pomocí 24hodinového formátu, s počáteční nulou, je-li hodina menší než 10 (například 09.10.20).

- V systému z/OSje čas GMT (Greenwich Mean Time), kdy se časová základna systému nastavuje přesně na GMT.
- V jiných prostředích je čas místní čas.
- V systému IBM ise čas vytvoření fronty může lišit od času vytvoření entity základního operačního systému (soubor nebo uživatelská oblast), 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 578. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X				

Během volání MQPUT se inkrementuje a během odvolání se volání MQGET znovu zobrazí. Je snižován během volání operace MQGET bez procházení a během odvolání volání MQPUT. Výsledkem je, že počet

zahrnuje zprávy, které byly vloženy do fronty v rámci pracovní jednotky, ale které ještě nebyly potvrzeny, i když nejsou způsobilé k načtení voláním MQGET. Podobně vyloučí zprávy, které byly načteny v rámci transakce pomocí volání MQGET, ale které dosud nebyly potvrzeny.

Tento počet také zahrnuje zprávy, které předaly svůj čas vypršení platnosti, ale ještě nebyly vyřazeny, ačkoli tyto zprávy nejsou vhodné k načtení. Další informace viz Pole MQMD-Expiry .

Zpracování jednotek práce a segmentace zpráv může způsobit, že *CurrentQDepth* překročí *MaxQDepth*. To však nemá vliv na schopnost načítání zpráv; *všechny* zprávy ve frontě lze načítat pomocí volání MQGET běžným způsobem.

Hodnota tohoto atributu kolísá, jak pracuje správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CURRENT_Q_DEPTH s voláním MQINQ.

Odpověď DefaultPut(MQLONG)

Určuje typ odezvy, který má být použit pro operace vložení do fronty, když aplikace určuje MQPMO_RESPONSE_AS_Q_DEF.

Tabulka 579. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	

Je to jedna z následujících hodnot:

MQPRT_SYNC_RESPONSE
Operace vložení je vydávána synchronně a vrací se odezva.

ODEZVA MQPRT_ASYNC_RESPONSE
Operace vložení je vydána asynchronně a vrací podmnožinu polí MQMD.

DefBind (MQLONG)

Jedná se o výchozí vazbu, která se používá, když je MQOO_BIND_AS_Q_DEF zadán v rámci volání MQOPEN a fronta je fronta klastru.

Tabulka 580. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Hodnota je jedna z následujících možností:

MQBND_BIND_ON_OPEN
Vazba byla opravena voláním MQOPEN.

MQBND_BIND_NOT_FIXED
Vazba nebyla opravena.

SKUPINA MQBND_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby skupina zpráv byla alokována do stejné cílové instance. Vzhledem k tomu, že tato hodnota je v produktu IBM WebSphere MQ 7.1nová, nesmí být použita, pokud se některé z aplikací otevírající tuto frontu připojují ke správci front produktu IBM WebSphere MQ 7.0.1 nebo k dřívějším správcům front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_BIND s voláním MQINQ.

DefinitionType (MQLONG)

Označuje, jak byla fronta definována.

Tabulka 581. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota je jedna z následujících možností:

MQQDT_PREDEFINED
Fronta je trvalá fronta vytvořená administrátorem systému; tuto frontu může odstranit pouze administrátor systému.

Předdefinované fronty se vytvářejí pomocí příkazu DEFINE MQSC a lze je odstranit pouze pomocí příkazu MQSC DELETE . Předdefinované fronty nelze vytvořit z modelových front.

Příkazy může být vydáno buď operátorem, nebo autorizovaným uživatelem odesláním zprávy příkazu do vstupní fronty příkazů (viz CommandInputQName attribute), kde získáte další informace).

MQQDT_PERMANENT_DYNAMIC
Fronta je trvalá fronta, která byla vytvořena aplikací, která vydala volání MQOPEN s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má hodnotu MQQDT_PERMANENT_DYNAMIC pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE-Zavřít objekt” na stránce 644.

Hodnota atributu **QSGDisp** pro trvalou dynamickou frontu je MQQSGD_Q_MMGR.

MQQDT_DOČASNÝ_DYNAMICKÝ
Fronta je dočasná fronta vytvořená aplikací, která vydala volání MQOPEN, s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má hodnotu MQQDT_TEMPORARY_DYNAMIC 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 MQQSGD_Q_MMGR.

DYNAMICKÝ_SDÍLENÝ_ADRESÁŘ_MQQQ
Fronta je sdílená trvalá fronta vytvořená aplikací, která vydala volání MQOPEN, s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má hodnotu MQQDT_SHARED_DYNAMIC pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část “MQCLOSE-Zavřít objekt” na stránce 644.

Hodnota atributu **QSGDisp** pro sdílenou dynamickou frontu je MQQSGD_SHARED.

Tento atribut v definici modelové fronty neukazuje, jak byla modelovaná fronta definována, protože modelové fronty jsou vždy předdefinované. Místo toho se hodnota tohoto atributu ve frontě modelu používá k určení *DefinitionType* každé z dynamických front vytvořených z definice modelové fronty pomocí volání MQOPEN.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEFINITION_TYPE s voláním MQINQ.

DefInputOpenOption (MQLONG)
Jedná se o výchozí způsob, jak otevřít frontu pro vstup.

Tabulka 582. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Používá se v případě, že je při volání MQOPEN při otevření fronty zadána volba MQOO_INPUT_AS_Q_DEF. Hodnota je jedna z následujících možností:

MQO_INPUT_EXCLUSIVE
Chcete-li získat zprávy s výlučným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání se nezdaří s kódem příčiny MQRC_OBJECT_IN_USE, je-li fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (MQOO_INPUT_SHARED nebo MQOO_INPUT_EXCLUSIVE).

MQO_INPUT_SHARED
Chcete-li získat zprávy se sdíleným přístupem, otevřete frontu.

Fronta je otevřena pro použití s následnými voláními MQGET. Volání se může zdařit, pokud je fronta aktuálně otevřena touto nebo jinou aplikací s MQOO_INPUT_SHARED, ale selže s kódem příčiny MQRC_OBJECT_IN_USE, je-li 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 trvání zpráv ve frontě. Používá se, je-li vlastnost MQPER_PERSISTENCE_AS_Q_DEF zadána v deskriptoru zpráv, když je zpráva vložena.

Tabulka 583. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud v cestě rozpoznání názvu fronty existuje více než jedna definice, bude použita výchozí perzistence z hodnoty tohoto atributu v cestě *první* v cestě v době volání MQPUT nebo MQPUT1 . To může být:

- Fronta aliasů
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*)

Hodnota je jedna z následujících možností:

MQPER_PERSISTENT
Zpráva přežije selhání systému a správce front se restartuje. Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, které jsou mapovány na objekt CFSTRUCT na úrovni CFLEVEL (2) nebo nižší, nebo kde je objekt CFSTRUCT definován jako RECOVER (NO).

Trvalé zprávy lze umístit do trvalých dynamických front a předdefinovaných front.

MQPER_NOT_PERSISTENT
Zpráva obvykle nepřežije selhání systému nebo správce front se restartuje. To platí i v případě, že se během restartu správce front nachází neporušená kopie zprávy v pomocné paměti.

V případě sdílených front dočasné zprávy *do* přežijí restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility použitého k ukládání zpráv ve sdílených frontách.

Trvalé i přechodné zprávy mohou existovat ve stejné frontě.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_PERSISTENCE s voláním MQINQ.

DefPriority (MQLONG)
Jedná se o výchozí prioritu zpráv ve frontě. To platí, je-li vlastnost MQPRI_PRIORITY_AS_Q_DEF uvedena v deskriptoru zpráv, když je zpráva vložena do fronty.

Tabulka 584. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	X

Pokud je v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude z hodnoty tohoto atributu použita výchozí priorita z hodnoty atributu v *první* definici v cestě v čase operace vložení. To může být:

- Fronta aliasů
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*)

Způsob, jakým je zpráva umístěna ve frontě, závisí na hodnotě atributu **MsgDeliverySequence** fronty:

- Je-li atribut **MsgDeliverySequence** MQMDS_PRIORITY, logická pozice, při které je zpráva umístěna do fronty, závisí na hodnotě pole *Priority* v deskriptoru zpráv.
- Je-li atribut **MsgDeliverySequence** MQMDS_FIFO, jsou zprávy umístěny do fronty, jako by měly prioritu rovnající se *DefPriority* z vyřešené fronty, bez ohledu na hodnotu pole *Priority* v deskriptoru zprávy. Pole *Priority* si však zachovává hodnotu určenou aplikací, která vložila zprávu. Další informace najdete v tématu Atribut posloupnostiMsgDelivery .

Priority jsou v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší); viz atributMaxPriority.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_PRIORITY s voláním MQINQ.

DefReadAhead (MQLONG)

Určuje výchozí chování dopředného čtení pro netrvalé zprávy doručené klientovi.

Tabulka 585. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X		

Volba DefReadAhead může být nastavena na jednu z následujících hodnot:

MQREADA_NO
Netrvalé zprávy nejsou odeslány klientovi před tím, než je aplikace vyžádá. Pokud klient skončí abnormálně, dojde ke ztrátě maximálně jedné netrvalé zprávy.

MQREADA_YES
Netrvalé zprávy jsou odeslány před klientem před tím, než je aplikace požaduje. Netrvalé zprávy mohou být ztraceny, pokud klient skončí abnormálně, nebo pokud klient nespotřebuje všechny zprávy, které odeslal.

MQREADA_DISABLED
Čtení předem netrvalých zpráv pro tuto frontu není povoleno. Zprávy se do klienta neodesílají 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 výchozí typ vložení odezvy (DEFPRESP) definuje hodnotu používanou aplikacemi, když byl PutResponseType v rámci MQPMO nastaven na hodnotu MQPMO_RESPONSE_AS_Q_DEF. Tento atribut je platný pro všechny typy front.

Tabulka 586. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	X

Hodnota je jedna z následujících možností:

SYNC
Operace umístění je vydána synchronně po vrácení odezvy.

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 mohou být do fronty umístěny zprávy distribučního seznamu.

Tabulka 587. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Agent kanálu zpráv (MCA) nastaví atribut tak, aby informoval lokálního správce front o tom, zda správce front na druhém konci kanálu podporuje distribuční seznamy. Tento posledně uvedený správce front (nazývaný správce front *partnering*) je ten, který další obdrží zprávu poté, co byla odebrána z lokální přenosové fronty odesílajícím programem MCA.

Odesílající agent MCA nastaví atribut vždy, když ustanoví připojení k přijímajícímu agentovi MCA v partnerského správce front. Tímto způsobem odesílající agent MCA může způsobit, že lokální správce front bude umístěn v přenosové frontě pouze na zprávy, které může partnerský správce front zpracovat správně.

Tento atribut se primárně používá pro přenosové fronty, ale popsané zpracování se provede bez ohledu na využití definované pro frontu (viz Atribut Použití).

Hodnota je jedna z následujících možností:

PODPOROVANÁ MQDL_
Zprávy distribučního seznamu mohou být uloženy do fronty a přeneseny do správce front partnera v daném formuláři. Tím se snižuje objem zpracování potřebný k odeslání zprávy do více míst určení.

PODPOROVÁNO MQDL_NOT_SUPPORTED
Zprávy distribučního seznamu nelze uložit do fronty, protože partneringový správce front nepodporuje distribuční seznamy. Pokud aplikace umístí zprávu distribučního seznamu a tato zpráva má být umístěna do této fronty, správce front rozdělí zprávu distribučního seznamu a umístí jednotlivé zprávy do fronty místo ní. Tím se zvyšuje objem zpracování potřebného k odeslání zprávy do více míst určení, ale zajišťuje, že zprávy budou zpracovány správně správcem front partnering.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DIR_LISTS s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

Tento atribut není v produktu z/OSpodporován.

HardenGetBackout (MQLONG)

Pro každou zprávu je počet uchován z počtu případů, kdy je zpráva načtena voláním MQGET v rámci pracovní jednotky a tato jednotka práce byla následně vrácena zpět.

Tabulka 588. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Tento počet je k dispozici v poli *BackoutCount* v deskriptoru zpráv po dokončení volání MQGET.

Počet odvolání zpráv přežije restarty správce front. Chcete-li však zajistit, aby byl počet přesný, musí být informace o stavu *upřesněné* (zaznamenané na disku nebo jiné trvalé paměťové jednotce) pokaždé, když volání MQGET načte zprávu v rámci pracovní jednotky pro tuto frontu. Není-li tato operace provedena, správce front se nezdaří a volání MQGET se vrátí, počet může nebo nemusí být zvýšen.

Zahazování informací pro každé volání MQGET v rámci jednotky práce však uvaluje další náklady na zpracování, takže nastavte atribut **HardenGetBackout** na MQQA_BACKOUT_HARDENED pouze tehdy, je-li nezbytné, aby byl počet přesný.

V systému Multiplatformsje počet odvolání zpráv vždy tvrzený, bez ohledu na nastavení tohoto atributu.

Možné jsou následující hodnoty:

MACKY_BACKOUT_HARDENED
Zaměření se používá k ujištění, že počet vrácení pro zprávy v této frontě je přesný.

MQQA_BACKUT_NOT_HARDENED
Zahradničení se nepoužívá, aby se zajistilo, že počet vrácení pro zprávy v této frontě je přesný. Počet by proto mohl být nižší, než by měl být.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_HARDEN_GET_BACKUP spolu s voláním MQINQ.

IndexType (MQLONG)

Tato hodnota určuje typ indexu, který správce front uchovává pro zprávy ve frontě.

Tabulka 589. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Typ požadovaného indexu závisí na tom, jak aplikace načítá zprávy a zda je fronta sdílenou frontou nebo nesdílenou frontou (viz QSGDisp attribute). Pro *IndexType*jsou možné následující hodnoty:

MQIT_NONE
Správce front pro tuto frontu nespravuje žádný index. Tuto hodnotu použijte pro fronty, které jsou běžně zpracovávány postupně, tj. bez použití kritérií výběru na volání MQGET.

MQIT_MSG_ID
Správce front udržuje index, který používá identifikátory zpráv ve frontě zpráv. Použijte tyto fronty hodnot, kde aplikace obvykle načítá zprávy s použitím identifikátoru zprávy jako kritéria výběru na volání MQGET.

MQIT_CORREL_ID
Správce front udržuje index, který používá identifikátory korelace pro zprávy ve frontě. Tuto hodnotu použijte pro fronty, kde aplikace obvykle načítá zprávy s použitím identifikátoru korelace jako kritéria výběru na volání MQGET.

MQIT_MSG_TOKEN

Důležité: Tento typ indexu by měl být použit pouze pro fronty použité s produktem IBM MQ Workflow pro produkt z/OS .

Správce front udržuje index, který používá tokeny zpráv ve frontě zpráv ve frontě pro použití s funkcemi správce pracovní zátěže (WLM) produktu z/OS.

Tuto volbu *musíte* zadat pro fronty spravované WLM; neuvádějte ji pro žádný jiný typ fronty. Tuto hodnotu také nepoužívejte pro frontu, v níž aplikace nepoužívá funkce správce pracovní zátěže produktu z/OS , ale načítá zprávy s použitím tokenu zprávy jako kritéria výběru pro volání MQGET.

ID_SKUPINY_MQIT_GROUP_ID
Správce front udržuje index, který používá identifikátory skupin zpráv ve frontě. Tato hodnota musí být použita pro fronty, kde aplikace načítá zprávy pomocí volby MQGMO_LOGICAL_ORDER na volání MQGET.

Fronta s tímto typem indexu nemůže být přenosovou frontou. Sdílená fronta s tímto typem indexu musí být definována tak, aby mapovala 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 s použitím volby MQGMO_LOGICAL_ORDER na volání MQGET. To znamená, že fyzická objednávka zpráv není obvykle v pořadí, ve kterém se zprávy přicházely do fronty.
2. Pokud má fronta MQIT_GROUP_ID *MsgDeliverySequence* MQMDS_PRIORITY, správce front používá priority zpráv 0 a 1 k optimalizaci načítání zpráv v logickém pořadí. Výsledkem je, že 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 skupiny a segmentů v části MQGMO-Options field.

Typ indexu, který má být použit v různých případech, je zobrazen v části Tabulka 590 na stránce 843 a Tabulka 591 na stránce 844.

Tabulka 590. Navržené nebo vyžadované hodnoty typu indexu fronty, pokud není zadán parametr MQGMO_LOGICAL_ORDER

Kritéria výběru při volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Není	Libovolný	Libovolný
Výběr pomocí jednoho identifikátoru:		
Identifikátor zprávy	Navrženo MQIT_MSG_ID	Je vyžadován název MQIT_NONE nebo MQIT_MSG_ID; bylo navrženo MQIT_MSG_ID
Identifikátor korelace	Navrhl MQIT_CORREL_ID	Požaduje se MQIT_CORREL_ID
Identifikátor skupiny	Navrhl objekt MQIT_GROUP_ID	Požaduje se MQIT_GROUP_ID
Výběr pomocí dvou identifikátorů:		
Identifikátor zprávy a identifikátor korelace	Navrženo MQIT_MSG_ID nebo MQIT_CORREL_ID	Je vyžadována proměnná 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 odlišených klíčů).
Identifikátor zprávy plus identifikátor skupiny	Navržený identifikátor MQIT_MSG_ID nebo MQIT_GROUP_ID	Nepodporováno

Tabulka 590. Navržené nebo vyžadované hodnoty typu indexu fronty, pokud není zadán parametr MQGMO_LOGICAL_ORDER (pokračování)

Kritéria výběru při volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Identifikátor korelace plus identifikátor skupiny	Navržený objekt MQIT_CORREL_ID nebo MQIT_GROUP_ID	Nepodporováno
Výběr pomocí tří identifikátorů:		
Identifikátor zprávy a identifikátor korelace plus identifikátor skupiny	Navrženo MQIT_MSG_ID nebo MQIT_CORREL_ID nebo MQIT_GROUP_ID	Nepodporováno
Výběr pomocí kritérií souvisejících se skupinou:		
Identifikátor skupiny a pořadové číslo zprávy	Požaduje se MQIT_GROUP_ID	Požaduje se MQIT_GROUP_ID
Pořadové číslo zprávy (musí být 1)	Požaduje se MQIT_GROUP_ID	Požaduje se MQIT_GROUP_ID
Výběr pomocí tokenu zprávy:		
Token zpráv pro použití aplikací	Nepoužívejte MQIT_MSG_TOKEN	
Token zpráv pro použití WLM	MQIT_MSG_TOKEN povinné	Nepodporováno

Tabulka 591. Navržené nebo vyžadované hodnoty typu indexu fronty, je-li zadáno MQGMO_LOGICAL_ORDER

Kritéria výběru při volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
Není	Požaduje se MQIT_GROUP_ID	Požaduje se MQIT_GROUP_ID
Výběr pomocí jednoho identifikátoru:		
Identifikátor zprávy	Požaduje se MQIT_GROUP_ID	Nepodporováno
Identifikátor korelace	Požaduje se MQIT_GROUP_ID	Nepodporováno
Identifikátor skupiny	Požaduje se MQIT_GROUP_ID	Požaduje se MQIT_GROUP_ID
Výběr pomocí dvou identifikátorů:		
Identifikátor zprávy a identifikátor korelace	Požaduje se MQIT_GROUP_ID	Nepodporováno
Identifikátor zprávy plus identifikátor skupiny	Požaduje se MQIT_GROUP_ID	Nepodporováno
Identifikátor korelace plus identifikátor skupiny	Požaduje se MQIT_GROUP_ID	Nepodporováno
Výběr pomocí tří identifikátorů:		
Identifikátor zprávy a identifikátor korelace plus identifikátor skupiny	Požaduje se 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 v systému z/OS.

InhibitGet (MQLONG)

Tento ovládací prvek určuje, zda jsou povoleny operace get pro tuto frontu.

Tabulka 592. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Je-li fronta alias fronta, musí být operace získání povoleny pro alias i pro základní frontu v době operace get, aby bylo volání MQGET úspěšné. Hodnota je jedna z následujících možností:

MQQA_GET_INHIBED
Operace získání jsou blokovány.

Volání MQGET selhalo s kódem příčiny MQRC_GET_INHIBITED. To zahrnuje volání MQGET, která uvádí MQGMO_BROT FIRST nebo MQGMO_BRONEXT NEXT.

Poznámka: Je-li operace MQGET pracující v rámci transakce úspěšně dokončena, změna hodnoty atributu **InhibitGet** následně na MQQA_GET_INHIBITED nezabrání tomu, aby byla jednotka práce potvrzena.

MQQA_GET_ALLOWED
Operace získání jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INHIBIT_GET s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

InhibitPut (MQLONG)

Tento ovládací prvek určuje, zda jsou povoleny operace vložení pro tuto frontu.

Tabulka 593. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Pokud je v cestě rozpoznání názvu fronty uvedena více než jedna definice, musí být operace vložení povoleny pro *každou* definici v cestě (včetně všech definic alias správce front) v době operace vložení, aby bylo volání MQPUT nebo MQPUT1 úspěšné. Hodnota je jedna z následujících možností:

MQQA_PUT_BLOKOVÁNO
Operace vložení jsou blokovány.

Volání MQPUT a MQPUT1 se nezdařily s kódem příčiny MQRC_PUT_INHIBITED.

Poznámka: Je-li operace MQPUT pracující v rámci transakce úspěšně dokončena, změna hodnoty atributu **InhibitPut** následně na MQQA_PUT_INHIBITED nezabrání tomu, aby byla jednotka práce potvrzena.

MQQA_PUT_ALLOWED
Operace vložení jsou povoleny.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_INHIBIT_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 594. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Správce front odešle do inicializační fronty zprávu spouštěče, je-li v důsledku přijetí zprávy přicházející do fronty, do níž tento atribut náleží, vyžadováno spuštění aplikace. Inicializační fronta musí být monitorována aplikací monitoru spouštěčů, která spouští 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 595. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Vzhledem k tomu, že atribut fronty **MaxMsgLength** lze nastavit nezávisle na atributu správce front produktu **MaxMsgLength** , je menší z těchto dvou hodnot skutečný horní limit délky nejdelší fyzické zprávy, kterou lze umístit do fronty.

Pokud správce front podporuje segmentaci, je možné, aby aplikace umístila *logickou* zprávu, která je delší než menší než menší ze dvou atributů **MaxMsgLength** , ale pouze v případě, že aplikace určuje příznak MQMF_SEGMENTATION_ALLOWED v deskriptoru MQMD. Je-li tento parametr zadán, horní mez pro délku logické zprávy je 999 999 999 bajtů, ale obvykle omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, výsledkem je nižší mezní hodnota.

Pokus o umístění do fronty, která je příliš dlouhá, selže s jedním z následujících kódů příčiny:

- MQRC_MSG_TOO_BIG_FOR_Q, je-li zpráva příliš velká pro frontu
- MQRC_MSG_TOO_BIG_FOR_Q_MGR, je-li zpráva pro správce front příliš velká, avšak pro danou frontu není příliš velká

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)

Jedná se o definovaný horní limit počtu fyzických zpráv, které mohou být ve frontě v daném okamžiku vůbec existovat.

Tabulka 596. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Pokus o vložení zprávy do fronty, který již obsahuje zprávy produktu **MaxQDepth** , selže s kódem příčiny MQRC_Q_FULL.

Zpracování jednotek práce a segmentace zpráv může způsobit, že skutečný počet fyzických zpráv ve frontě překročí **MaxQDepth**. To však nemá vliv na možnosti načtení 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 nemůže hodnota překročit 999 999 999:

  - AIX
  - Linux
  - Windows
  - z/OS
- IBM i V systému IBM inesmí hodnota přesáhnout 640 000.

Poznámka: Úložný prostor dostupný pro frontu může být vyčerpán i v případě, že ve frontě je méně než **MaxQDepth** zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MAX_Q_DEPTH s voláním MQINQ.

Posloupnost MsgDelivery(MQLONG)

Tabulka 597. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tím určíte pořadí, ve kterém volání MQGET vrátí zprávy do aplikace:

MQSD_FIFO
Zprávy jsou vraceny ve FIFO pořadí (první dovnitř, první ven).

Volání MQGET vrátí zprávu *první* , která splňuje kritéria výběru zadaná ve volání, bez ohledu na prioritu zprávy.

PRIORITA MQMS_PRIORITY
Zprávy jsou vraceny v pořadí priority.

Volání MQGET vrací zprávu *highest-priority* , která odpovídá kritériím výběru zadaným ve volání. V rámci každé úrovně priority jsou zprávy vraceny ve FIFO pořadí (první dovnitř, první ven).

- On z/OS, if the queue has an *IndexType* of MQIT_GROUP_ID, the **MsgDeliverySequence** attribute specifies the order in which message groups are returned to the application. Konkrétní pořadí, ve kterém jsou skupiny vráceny, je určeno pozicí nebo prioritou první zprávy v každé skupině. Fyzické pořadí zpráv ve frontě není definováno, protože fronta je optimalizována pro efektivní načítání zpráv s použitím volby MQGMO_LOGICAL_ORDER na volání MQGET.
- Pokud je v produktu z/OShodnota *IndexType* MQIT_GROUP_ID a *MsgDeliverySequence* je MQMDS_PRIORITY, správce front používá priority zpráv nula a jeden pro optimalizaci načítání zpráv v logickém pořadí. Výsledkem je, že 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 se ve frontě nacházejí zprávy, je posloupnost doručení následující:

- Pořadí, ve kterém jsou zprávy vraceny voláním MQGET, jsou určovány hodnotami atributů **MsgDeliverySequence** a **DefPriority** platných pro frontu v době, kdy zpráva dorazí do fronty:
  - Je-li při doručení zprávy produkt *MsgDeliverySequence* MQMDS_FIFO, zpráva se umístí do fronty, jako by její priorita byla *DefPriority*. To nemá vliv na hodnotu pole *Priority* v deskriptoru zprávy této zprávy; v tomto poli je zachována hodnota, kterou měla při prvním vložení zprávy.
  - Pokud je při doručení zprávy *MsgDeliverySequence* MQMDS_PRIORITY, zpráva se umístí do fronty na místo odpovídající prioritě zadané argumentem *Priority* v deskriptoru zprávy.

  Pokud se změní hodnota atributu **MsgDeliverySequence** , zatímco se ve frontě nacházejí zprávy, pořadí zpráv ve frontě se nezmění.

  Pokud se změní hodnota atributu **DefPriority** , zatímco ve frontě jsou zprávy, zprávy nejsou nutně doručovány ve FIFO pořadí, i když je atribut **MsgDeliverySequence** nastaven na MQMDS_FIFO; ty, které byly umístěny do fronty na vyšší prioritě, jsou doručeny první.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_MSG_DELIVERY_SEQUENCE s voláním MQINQ.

NonPersistentMessageClass (MQLONG)
Cíl spolehlivosti pro přechodné zprávy.

Tabulka 598. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Uvádí okolnosti, za kterých jsou přechodné zprávy zařazené do této fronty vyřazeny:

MQNPM_CLASS_NORMAL
Netrvalé zprávy jsou omezeny na dobu trvání relace správce front; zprávy jsou zahozeny v případě restartování správce front. Tato hodnota je platná pouze pro nesdílené fronty a je výchozí hodnotou.

VYSOKÁ HODNOTA MQNPM_CLASS_HIGH
Správce front se pokusí zachovat přechodné zprávy po dobu životnosti fronty. Netrvalé zprávy mohou být v případě selhání ztraceny. Tato hodnota je vynucena pro sdílené fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_NPM_CLASS s voláním MQINQ.

Počet OpenInputCount (MQLONG)
Jedná se o počet popisovačů, které jsou aktuálně platné pro odebrání zpráv z fronty pomocí volání MQGET.

Tabulka 599. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Jedná se o celkový počet těchto popisovačů známých pro *lokálního* správce front. Je-li fronta sdílenou frontou, tento počet nezahrne otevření pro vstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do níž patří lokální správce front.

Počet zahrnuje manipulátory, ve kterých byla pro vstup otevřena fronta aliasů, která byla rozpoznána pro tuto frontu. Počet nezahrnuje manipulátory, ve kterých byla fronta otevřena pro akce, které neobsahovaly vstup (například, fronta otevřená pouze pro procházení).

Hodnota tohoto atributu kolísá, jak pracuje správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_OPEN_INPUT_COUNT s voláním MQINQ.

Počet OpenOutputCount (MQLONG)
Jedná se o počet popisovačů, které jsou momentálně platné pro přidání zpráv do fronty prostřednictvím volání MQPUT.

Tabulka 600. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Jedná se o celkový počet takových manipulátorů, které jsou známy správci front *local* ; nezahrne se otevření pro výstup, který byl proveden pro tuto frontu ve vzdálených správcích front. Je-li fronta sdílenou frontou, tento počet nezahrnuje otevření pro výstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do níž patří lokální správce front.

Počet zahrnuje manipulátory, ve kterých byla pro výstup otevřena fronta aliasů, která se překládá do této fronty. Počet nezahrnuje manipulátory, kde byla fronta otevřena pro akce, které neobsahovaly výstup (například, fronta byla otevřena pouze pro zjištění).

Hodnota tohoto atributu kolísá, jak pracuje 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 službu zařadit do fronty.

Tabulka 601. Typy front, na které se vztahuje tento atribut

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 pro zprávy, které jsou načítány z front s použitím volby MQGET s volbou MQGMO_PROPERTIES_AS_Q_DEF.

Tabulka 602. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Hodnota je jedna z následujících možností:

MQPROP_ALL
Všechny vlastnosti zprávy jsou zahrnuty ve zprávě, 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, bude chování vracet vlastnosti v popisovači zprávy.

KOMPATIBILITA MQPROP_COMPATIBILITY
Pokud zpráva obsahuje vlastnost s předponou mcd., jms., usr. 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 JMS související vlastnosti, v záhlaví MQRFH2 v datech zprávy pokračovat v práci beze změn. Je-li zadán popisovač zprávy, bude chování vracet 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 dodaný v poli MsgHandle struktury MQGMO na volání MQGET je ignorován. Vlastnosti zprávy nejsou pomocí popisovače zprávy přístupné.

MQPROP_NONE
Všechny vlastnosti zprávy, kromě vlastností v deskriptoru zprávy (nebo rozšíření), jsou před doručením zprávy do aplikace odebrány ze zprávy. Je-li zadán popisovač zprávy, bude chování vracet vlastnosti v popisovači zprávy.

Tento parametr lze použít pro fronty lokálního, aliasu a modelu. Chcete-li určit jeho hodnotu, použijte selektor MQIA_PROPERTY_CONTROL s voláním MQINQ.

Událost QDepthHigh(MQLONG)

Tento ovládací prvek určuje, zda jsou generovány události vysoké hloubky fronty.

Tabulka 603. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Příliš dlouhá fronta označuje, že aplikace založila zprávu do fronty a způsobila, že se počet zpráv ve frontě stal větší nebo roven horní prahové hodnotě hloubky fronty (viz atribut **QDepthHighLimit**).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPT_TH_HIGH_EVENT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení její hodnoty.

Limit QDepthHigh(MQLONG)

Jedná se o prahovou hodnotu, proti níž je porovnávána hloubka fronty pro generování události Příliš dlouhá fronta.

Tabulka 604. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Tato událost označuje, že aplikace vložila zprávu do fronty, a že to způsobilo, že se počet zpráv ve frontě stal větší nebo roven horní prahové hodnotě hloubky fronty. Viz QDepthHighAtribut události.

Hodnota je vyjádřena jako procentní část z maximální hloubky fronty (atribut**MaxQDepth**) a je větší než nebo rovna 0 a menší nebo rovna 100. Výchozí hodnota je 80.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPT_TH_HIGH_LIMIT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení její hodnoty.

Událost QDepthLow(MQLONG)

Tento ovládací prvek určuje, zda jsou generovány události nízké hloubky fronty.

Tabulka 605. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Příliš dlouhá fronta označuje, že aplikace načetla zprávu z fronty, a že to způsobilo, že se počet zpráv ve frontě stal méně nebo roven dolní prahové hodnotě hloubky fronty (viz QDepthLowLimit atributu).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte pro volání MQINQ selektor MQIA_Q_DEPTH_LOW_EVENT.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení její hodnoty.

Limit QDepthLow(MQLONG)
Jedná se o prahovou hodnotu, proti níž je porovnávána hloubka fronty, aby se vygenerovala událost Nízká hloubka fronty.

Tabulka 606. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Tato událost označuje, že aplikace načetla zprávu z fronty, a že to způsobilo, že se počet zpráv ve frontě stal méně než nebo roven dolní prahové hodnotě hloubky fronty. Viz QDepthLowAtribut události.

Hodnota je vyjádřena jako procentní část z maximální hloubky fronty (atribut**MaxQDepth**) a je větší než 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 její hodnoty.

Událost QDepthMax(MQLONG)
Tento ovládací prvek určuje, zda jsou generovány události zaplnění fronty. Událost Plná fronta indikuje, že vložení do fronty bylo zamítnuto, protože fronta je plná, to znamená, že hloubka fronty již dosáhla maximální hodnoty.

Tabulka 607. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících možností:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_DEPT_TH_MAX_EVENT s voláním MQINQ.

Tento atribut je podporován v systému z/OS, ale volání MQINQ nelze použít k určení její hodnoty.

QDesc (MQCHAR64)
Toto pole použijte pro popisný komentář.

Tabulka 608. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Obsah tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace 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.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_DESC s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_DESC_LENGTH.

QName (MQCHAR48)

Jedná se o název fronty definované v lokálním správci front.

Tabulka 609. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Všechny fronty definované ve správci front sdílejí stejný obor názvů fronty. Proto fronta MQQT_LOCAL a fronta MQQT_ALIAS 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)

Toto je interval služby použitý pro porovnání ke generování událostí Vysoká a servisní interval Interval služby OK.

Tabulka 610. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Viz QServiceIntervalAtribut události.

Hodnota je v milisekundách, a je větší než 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 její hodnoty.

Událost QServiceInterval(MQLONG)

Tento parametr řídí, zda jsou generovány události vysokého nebo servisního intervalu v rámci intervalu služby.

Tabulka 611. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

- Vysoká událost Interval služby se generuje, když kontrola označuje, že od fronty nebyly načteny žádné zprávy alespoň po dobu uvedenou atributem **QServiceInterval** .
- Událost Interval služby OK je generována, pokud kontrola indikuje, že zprávy byly získány z fronty v čase indikovaném atributem **QServiceInterval** .

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Hodnota je jedna z následujících možností:

MQQSIE_HIGH
Události vysoké intervalu služby fronty povoleny.

- Události vysoké intervalu služby fronty jsou **povoleny** a
- Události servisního intervalu fronty OK jsou **zakázány**.

MQQSIE_OK
Události OK intervalu služby fronty povoleny.

- Události vysoké intervalu služby fronty jsou **zakázány** a
- Události servisního intervalu fronty OK jsou **povoleny**.

MQQSIE_NONE
Nejsou povoleny žádné události intervalu služby fronty.

- Události vysoké intervalu služby fronty jsou **zakázány** a
- Události servisního intervalu fronty OK jsou také **zakázány**.

Pro sdílené fronty je hodnota tohoto atributu ignorována; předpokládá se hodnota MQQSIE_NONE.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_SERVICE_INTERVAL_EVENT s voláním MQINQ.

V systému z/OSnelze použít volání MQINQ k určení hodnoty tohoto atributu.

QSGDisp (MQLONG)
Určuje dispozice fronty.

Tabulka 612. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Hodnota je jedna z následujících možností:

MQQSGD_Q_MGR
Objekt má dispozice 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 tyto objekty jsou samostatné objekty a mezi nimi neexistuje žádná korelace. Jejich atributy nejsou omezeny na to, aby byly stejné jako ostatní.

MQQSD_KOPIE
Objekt je lokální kopií definice hlavního objektu, který existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít vlastní kopii daného objektu. Zpočátku 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.

SDÍLENÝ MQQSGD_SHARED
Objekt má sdílené odebrání. To znamená, že ve sdíleném úložišti existuje jediná instance objektu, která je známá všem správcům front ve skupině sdílení front. Přistupuje-li správce front v dané skupině k objektu, bude přistupovat k jedné sdílené instanci objektu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_QSG_DISP s voláním MQINQ.

z/OS Tento atribut je podporován pouze v systému z/OS.

QueueAccounting (MQLONG)

Tabulka 613. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	

Tato volba řídí kolekci dat evidence pro frontu. Pro data evidence, která mají být shromažďována pro tuto frontu, musí být také povolena data evidence pro toto připojení buď pomocí atributu QMGR ACCTQ, nebo pole Volby ve struktuře MQCNO v rámci volání MQCONNX.

Tento atribut může mít některou z následujících hodnot:

MQMON_Q_MGR
Účtovací data pro tuto frontu se shromažďují na základě nastavení atributu QMGR ACCTQ. Toto je výchozí nastavení.

MQMON_OFF
Neshromažďovat účtovací data pro tuto frontu.

MQMON_ON
Shromáždí účtovací data 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 614. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota je jedna z následujících možností:

MQMON_Q_MGR
Shromážděte data monitorování v závislosti na nastavení atributu správce front produktu **QueueMonitoring** . Toto je výchozí hodnota.

MQMON_OFF
Shromažďování online monitorovacích dat je pro tuto frontu vypnuto.

MQMON_LOW
Pokud hodnota atributu správce front produktu **QueueMonitoring** není MQMON_NONE, je zapnuto shromažďování online monitorování dat s nízkou rychlostí shromažďování dat pro tuto frontu.

MQMON_MEDIUM
Pokud hodnota atributu správce front produktu **QueueMonitoring** není MQMON_NONE, je zapnuto shromažďování online monitorování dat se střední rychlostí shromažďování dat pro tuto frontu.

MQMON_HIGH
Pokud hodnota atributu správce front produktu **QueueMonitoring** není MQMON_NONE, je zapnuto shromažďování online monitorování dat 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 615. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	

Tento ovládací prvek řídí shromažďování statistických dat pro frontu.

Tento atribut může mít některou z následujících hodnot:

MQMON_Q_MGR
Účtovací data pro tuto frontu jsou shromažďována na základě nastavení atributu QMGR STATQ. Toto je výchozí nastavení.

MQMON_OFF
Vypnout shromažďování statistických dat pro tuto frontu.

MQMON_ON
Povolit shromažďování statistických dat pro tuto frontu.

QType (MQLONG)

Tabulka 616. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	X

Jedná se o typ fronty; má jednu z následujících hodnot:

ALIAS MQQ_ALIAS
Definice alias fronty.

KLASTR MQQ_CLUSTER
Fronta klastru.

MQQ_LOCAL
Lokální fronta.

MQQT_REMOTE
Lokální definice vzdálené fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_Q_TYPE s voláním MQINQ.

Název RemoteQMgr(MQCHAR48)

Tabulka 617. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Jedná se o název vzdáleného správce front, na kterém je definována fronta **RemoteQName** . 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 prázdná a nesmí se jednat o název lokálního správce front. Je-li parametr **XmitQName** prázdný, 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 produktu **DefXmitQName** .

Je-li tato definice použita pro alias správce front, **RemoteQMgrName** je název správce front, pro který je alias vytvořen. Může se jednat o název lokálního správce front. Jinak, je-li **XmitQName** při otevření prázdné, musí existovat lokální fronta s názvem, který je stejný jako **RemoteQMgrName**; tato fronta se používá jako přenosová fronta.

Je-li tato definice použita pro alias odpovědi na alias, je tento název názvem správce front, který má být **ReplyToQMgr**.

Poznámka: Při vytváření nebo úpravě definice fronty není prováděno žádné ověřování pro hodnotu určenou pro tento atribut.

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 618. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Jedná se o název fronty, jak je znám ve vzdáleném správci front *RemoteQMgrName*.

Pokud aplikace otevře lokální definici vzdálené fronty, když se otevřená vyskytuje, *RemoteQName* nesmí být prázdné.

Je-li tato definice použita pro definici aliasu správce front, musí být při otevření prázdná hodnota *RemoteQName* .

Je-li definice použita pro alias odpovědi na alias, je tento název názvem fronty, která má být *ReplyToQ*.

Poznámka: Při vytváření nebo úpravě definice fronty není prováděno žádné ověřování pro hodnotu určenou pro tento atribut.

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 dobu, po kterou se má fronta uchovávat. Po uplynutí této doby je fronta vhodná k odstranění.

Tabulka 619. Typy front, na které se vztahuje tento atribut

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 aplikaci úklidu nebo operátorovi identifikovat a odstranit fronty, které již nejsou zapotřebí.

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 odpovědností uživatele provést požadovanou akci.

Použijte realistický interval uchování, abyste zabránili akumulaci trvalých dynamických front (viz DefinitionType attribute). Tento atribut lze však také použít s předdefinovanými frontami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_RETENTION_INTERVAL s voláním MQINQ.

Rozsah (MQLONG)

Tento příkaz určuje, zda položka pro tuto frontu existuje také v adresáři buňky.

Tabulka 620. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Adresář buňky je poskytován instalovatelnou službou názvů. Hodnota je jedna z následujících možností:

MQSCOM_Q_MGR
Definice fronty má obor správce front: definice fronty není rozšířena nad rámec správce front, který je vlastní. Chcete-li otevřít frontu pro výstup z jiného správce front, je třeba zadat buď název vlastnícího správce front, nebo musí mít jiný správce front lokální definici fronty.

BUŇKA MQSCO_CELL
Definice fronty má rozsah buňky: definice fronty je také umístěna v adresáři buňky, který je k dispozici všem správcům front v buňce. Frontu lze otevřít pro výstup z libovolného správce front v buňce zadáním názvu fronty. Název správce front, který tuto frontu vlastní, nemusí být zadán. Definice fronty však není k dispozici pro žádného správce front v buňce, která má také lokální definici fronty s tímto názvem, protože lokální definice má přednost.

Adresář buňky je poskytován instalovatelnou službou názvů.

Model 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.

Na podporu tohoto atributu se vztahují následující omezení:

- V systému IBM ije tento atribut podporován, je však platný pouze parametr MQSCO_Q_MGR.
- V systému z/OSnení atribut podporován.

Sdílitelnost (MQLONG)
Označuje, zda lze frontu otevřít pro vstup vícenásobně souběžně.

Tabulka 621. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Hodnota je jedna z následujících možností:

MQQA_SHAREABLE
Fronta je možné sdílet.

Volba vícenásobného otevření s volbou MQOO_INPUT_SHARED je povolena.

MQQA_NOT_SHAREABLE
Fronta není možné sdílet.

Volání MQOPEN s volbou MQOO_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 uživatelsky definovaný název, který definuje fyzickou paměť použitou k zadržení fronty. V praxi se zpráva zapisuje na disk pouze tehdy, je-li třeba, aby byla odstránkována z vyrovnávací paměti.

Tabulka 622. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Chcete-li určit hodnotu tohoto atributu, použijte selektor CLASS MQCA_STORAGE_CLASS s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_STORAGE_CLASS_LENGTH.

z/OS Tento atribut je podporován pouze v systému z/OS.

TriggerControl (MQLONG)

Tento příkaz určuje, zda se zprávy spouštěče zapisují do inicializační fronty ke spuštění aplikace pro obsluhu fronty.

Tabulka 623. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je jedna z následujících možností:

MQTC_OFF
Pro tuto frontu se nemají zapsat žádné zprávy spouštěče. Hodnota *TriggerType* je v tomto případě irelevantní.

MQTC_ON
Zprávy spouštěče se mají zapsat do této fronty, když se vyskytnou odpovídající události triggeru.

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, že zpráva spouštěče bude zapsána do inicializační fronty.

Tabulka 624. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Obsah těchto dat nemá význam pro správce front. Smysluje se buď do aplikace monitoru spouštěčů, která zpracovává inicializační frontu, nebo do aplikace, kterou spouští monitor spouštěčů.

Znakový řetězec nesmí obsahovat žádné hodnoty null. Je-li to nutné, doplní se 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 625. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší, které musí být ve frontě, než se vypíše zpráva spouštěče. To platí, je-li parametr *TriggerType* nastaven na hodnotu MQTT_DEPTH. Hodnota *TriggerDepth* je jedna nebo více. Tento atribut se nepoužívá jinak.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TRIGGER_DEPTH s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

TriggerMsgPriorita (MQLONG)

Jedná se o prioritu zprávy, pod níž zprávy nepřispívají ke generování zpráv spouštěče (to znamená, že správce front tyto zprávy ignoruje při rozhodování, zda má generovat zprávu spouštěče).

Tabulka 626. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

TriggerMsgPriority může být v rozsahu nula (nejnižší) až *MaxPriority* (highest; viz MaxPriority attribute); hodnota nula způsobí, že všechny zprávy přispívají k generaci 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)

Tím se řídí podmínky, za kterých jsou zprávy spouštěče zapisovány jako výsledek zpráv přicházejících do této fronty.

Tabulka 627. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Má jednu z následujících hodnot:

MQTTE_NONE
Žádné zprávy spouštěče se nezapisují jako výsledek zpráv v této frontě. To má stejný účinek jako nastavení *TriggerControl* na MQTC_OFF.

NEJPRVE MQTT_FIRST
Zpráva spouštěče se zapisuje vždy, když se počet zpráv priority *TriggerMsgPriority* nebo vyšší ve frontě změní z 0 na 1.

MQTT_EVERY
Zpráva spouštěče se zapisuje vždy, když se do fronty dostane zpráva o prioritě *TriggerMsgPriority* nebo vyšší.

MQTT_DEPTH
Zpráva spouštěče se zapisuje vždy, když se počet zpráv priority *TriggerMsgPriority* nebo vyšší na frontě rovná nebo překročí *TriggerDepth*. Po zápisu zprávy spouštěče je parametr *TriggerControl* nastaven na hodnotu MQTC_OFF, aby se zabránilo dalšímu spuštění, dokud nebude znovu explicitně zapnuto.

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, pro kterou frontu se používá fronta.

Tabulka 628. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Hodnota je jedna z následujících možností:

MQUS_NORMAL
Jedná se o frontu, kterou aplikace používají při vkládání a získávání zpráv; fronta není přenosová fronta.

PŘENOS MQUS_TRANSMISSION
Jedná se o frontu používanou k ukládání zpráv určených pro vzdálené správce front. Když aplikace odešle zprávu do vzdálené fronty, lokální správce front uloží tuto zprávu dočasně do příslušné přenosové fronty ve speciálním formátu. Agent kanálu zpráv poté přečte zprávu z přenosové fronty

a odešle zprávu do vzdáleného správce front. Další informace o konfiguraci vzdálené administrace najdete v tématu Konfigurace správců front pro vzdálenou administraci.

Pouze privilegované aplikace mohou otevřít přenosovou frontu pro MQOO_OUTPUT tak, aby na ni byly přímo vloženy zprávy. Obvykle to dělají pouze obslužné aplikace. Ujistěte se, že formát dat zprávy je správný (viz “MQXQH-záhlaví přenosové fronty” na stránce 613) nebo se mohou vyskytnout chyby během procesu přenosu. Kontext není předáván nebo nastaven, pokud není zadán jeden z kontextových voleb MQPMO_ * _CONTEXT.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_USAGE s voláním MQINQ.

XmitQName (MQCHAR48)

Jedná se o název přenosové fronty. Je-li tento atribut neprázdný, když se vyskytne otevření, buď pro vzdálenou frontu, nebo pro definici alias správce front, uvádí jméno lokální přenosové fronty, která má být použita pro předání zprávy.

Tabulka 629. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
			X	

Je-li parametr **XmitQName** prázdný, je jako přenosová fronta použita lokální fronta s názvem, který je stejný jako **RemoteQMgrName** . Pokud neexistuje žádná fronta s názvem **RemoteQMgrName**, použije se fronta určená atributem správce front produktu **DefXmitQName** .

Tento atribut je ignorován, je-li definice použita 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 ve všech systémech IBM MQ a IBM MQ MQI clients připojených k těmto systémům.

Poznámka: Názvy atributů zobrazené v této sekci jsou popisné názvy použité spolu s voláními MQINQ a MQSET. Názvy jsou stejné jako u příkazů PCF. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; další informace najdete v tématu Příkazy MQSC .

Tabulka 630. 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, doplněno 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)

Jedná se o 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:

POČET NÁZVŮ 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 vytvořena definičním procesem. Obsah tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace DBCS může toto pole obsahovat znaky DBCS (s výhradou maximální délky 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.

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 seznamu názvů naleznete v části Další názvy objektů .

Každý seznam názvů má název odlišný od názvů jiných seznamů názvů náležejících ke správci front, ale mohou duplikovat názvy jiných objektů správce 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)

Určuje charakter názvů v seznamu názvů a určuje, jak se seznam názvů používá. Je to jedna z následujících hodnot:

MQNT_NONE
Jedná se o seznam názvů bez přiřazeného typu.

MQNT_Q
Seznam názvů obsahující názvy front.

KLASTR 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 v systému z/OS.

Názvy (MQCHAR48xNameCount)

Jedná se o seznam názvů *NameCount* , kde každé jméno představuje název objektu, který je definován pro lokálního správce front. Další informace o názvech objektů najdete v tématu Pravidla pojmenování objektů IBM MQ.

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)

Určuje dispozice seznamu názvů. Hodnota je jedna z následujících možností:

MQQSGD_Q_MGR
Objekt má dispozice správce front: definice objektu je známá 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 tyto objekty jsou samostatné objekty a mezi nimi neexistuje žádná korelace. Jejich atributy nejsou omezeny na to, aby byly stejné jako ostatní.

MQQSD_KOPIE
Objekt je lokální kopií definice hlavního objektu, který existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít vlastní kopii daného objektu. Na počátku všechny kopie mají stejné atributy, ale můžete každou kopii změnit pomocí příkazů MQSC, takže se její atributy liší od svých atributů 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 v 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 použité spolu s voláními MQINQ a MQSET; názvy jsou stejné jako u příkazů PCF. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; další informace najdete v tématu Příkazy MQSC .

Tabulka 631. 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, doplněno 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)

Jedná se o 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 identifikující aplikaci, která má být spuštěna. Tyto informace používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě; informace se odesílají do inicializační fronty jako část zprávy spouštěče.

Význam *ApplId* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný serverem IBM MQ vyžaduje, aby byl *ApplId* název spustitelného programu. Níže uvedené poznámky se vztahují na uvedená prostředí:

- V systémech z/OSmusí produkt *ApplId* být:
  - Identifikátor transakce CICS , pro aplikace spuštěné pomocí transakce monitoru spouštěčů CICS CKTI
  - Identifikátor transakce IMS , pro aplikace spuštěné pomocí monitoru spouštěčů IMS CSQQTRMN
- V systému Windowsmůže být název programu uveden jako předpona cesty k disku a adresáři.
- V systému AIX and Linuxmůže být název programu uveden jako předpona cesty k adresáři.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se 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 používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě; informace se odesílají do inicializační fronty jako část zprávy spouštěče.

Význam *EnvData* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytnutý produktem IBM MQ připojuje *EnvData* k seznamu parametrů předanému do spuštěné aplikace. Seznam parametrů se skládá ze struktury MQTMC2 , za nímž následuje jedna mezera, následované *EnvData* s odstraněnými koncovými mezerami. Níže uvedené poznámky se vztahují na uvedená prostředí:

- V systému z/OS:
  - *EnvData* nepoužívá aplikace monitoru spouštěčů poskytované produktem IBM MQ.
  - Je-li ApplType MQAT_WLM, můžete zadat výchozí hodnoty do polí EnvData pro pole ServiceName a ServiceStep v záhlaví pracovních informací (MQWIH).
- V systému AIX and Linuxlze produkt *EnvData* nastavit na znak & a spustit tak spuštěnou aplikaci na pozadí.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se 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)

Toto pole použijte pro popisný komentář. Obsah tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace 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.

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 ke správci front. Název definice procesu by však mohl být stejný jako názvy jiných objektů správce front různých typů (například fronty).

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)

Určuje dispozice definice procesu. Hodnota je jedna z následujících možností:

MQQSGD_Q_MGR
Objekt má dispozice správce front: definice objektu je známá 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 tyto objekty jsou samostatné objekty a mezi nimi neexistuje žádná korelace. Jejich atributy nejsou omezeny na to, aby byly stejné jako ostatní.

MQQSD_KOPIE
Objekt je lokální kopií definice hlavního objektu, který existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít vlastní kopii daného objektu. Na počátku všechny kopie mají stejné atributy, ale můžete každou kopii změnit pomocí příkazů MQSC, takže se její atributy liší od svých atributů 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 v systému z/OS.

UserData (MQCHAR128)

UserData je řetězec znaků, který obsahuje informace o uživateli související s aplikací, která má být spuštěna. Tyto informace používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě, nebo aplikaci, která je spuštěna monitorem spouštěčů. Informace se odešlou do inicializační fronty jako část zprávy spouštěče.

Význam *UserData* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ předává *UserData* do spuštěné aplikace jako součást seznamu parametrů. Seznam parametrů se skládá ze struktury MQTMC2 (obsahující *UserData*), za nímž následuje jedna mezera, za kterou následuje *EnvData* s odebranými koncovými mezerami.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se vpravo mezerami. Pro produkt Microsoft Windowsnesmí znakový řetězec obsahovat uvozovky, pokud má být definice procesu předána produktu **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 MQI (IBM MQ Message Queue Interface) a MQAI (IBM MQ Administration Interface) jsou správci front nebo uživatelská procedura vraceny kód produktu **dokončení** a kód **důvod** , aby se označoval úspěch nebo selhání volání.

Aplikace nesmí záviset na chybách, které jsou kontrolovány ve specifickém pořadí, kromě případů, kdy je to výslovně uvedeno. Pokud by z volání mohlo dojít k více než jednomu kódu dokončení nebo kódu příčiny, závisí konkrétní hlášená chyba na implementaci.

Kontrola aplikací po úspěšném dokončení po volání rozhraní API produktu IBM MQ musí vždy zkontrolovat kód dokončení. Nepředpokládejte, že hodnota kódu dokončení je založena na hodnotě kódu příčiny.

Kódy dokončení

Parametr kódu dokončení (*CompCode*) umožňuje volajícímu rychle zjistit, zda bylo volání úspěšně dokončeno, dokončeno částečně, nebo selhalo.Následuje seznam kódů dokončení, s větším detailem, než je uvedeno v popisech volání:

MQCC_OK
Volání bylo dokončeno plně; všechny výstupní parametry byly nastaveny. Parametr **Reason** má v tomto případě vždy hodnotu MQRC_NONE.

VAROVÁNÍ MQCC_WARNING
Volání bylo dokončeno částečně. Některé výstupní parametry mohly být nastaveny spolu s výstupními parametry *CompCode* a *Reason* . Parametr **Reason** poskytuje další informace o částečném dokončení.

SELHÁNÍ MQCC_FAILED
Zpracování volání nebylo dokončeno. Stav správce front je nezměněn, není-li výslovně uvedeno jinak, je stav správce front nezměněn. Výstupní parametry *CompCode* a *Reason* byly nastaveny; ostatní parametry jsou nezměněny, kromě případů, kdy byly zaznamenány.

Příčinou může být chyba v aplikačním programu nebo může být výsledkem určité situace mimo program, například oprávnění uživatele mohlo být odvoláno. Parametr **Reason** udává další informace o chybě.

Kódy příčin

Parametr kódu příčiny (*Reason*) kvalifikuje parametr kódu dokončení (*CompCode*).

Není-li k dispozici žádný speciální důvod k vytvoření sestavy, je vrácen příkaz MQRC_NONE. Při úspěšném volání je vrácen objekt MQCC_OK a MQRC_NONE.

Je-li kód dokončení buď MQCC_WARNING, nebo MQCC_FAILED, správce front vždy nahlásí kvalifikovanou příčinu; podrobnosti jsou uvedeny pod každým popisem volání.

V případech, kdy rutiny uživatelské procedury nastavují kódy dokončení a důvody, musí tyto předpisy dodržovat. Dále platí, že všechny speciální hodnoty důvodu definované uživatelskými procedurami musí být menší než nula, aby se zajistilo, že se nekolidují s hodnotami nadefinovanými správcem front. Výstupy mohou nastavit důvody, které již správce front definuje, kde je to vhodné.

Kódy příčiny se také vyskytují v:

- Pole *Reason* struktury MQDLH
- Pole *Feedback* struktury MQMD

Úplný popis kódů příčiny naleznete v tématu Zprávy a kódy příčin.

Pravidla pro ověření platnosti voleb MQI

Tato sekce obsahuje seznam situací, které generují kód příčiny MQRC_OPTIONS_ERROR z volání MQOPEN, MQPUT, MQPUT1, MQGET, MQCLOSE nebo MQSUB.

Volání MQOPEN

Volby volání MQOPEN:

- Musí být zadán alespoň *jeden* z následujících:
  - MQOOK_BROWSE
  - MQOO_INPUT_EXCLUSIVE [1]
  - Hodnota MQOO_INPUT_SHARED [1]
  - MQOO_INPUT_AS_Q_DEF [1]
  - MQO_ DOTÁZAT SE
  - MQOOK_VÝSTUP
  - MQOOK_SADA
  - MQOO_BIND_ON_OPEN [2]
  - MQOO_BIND_NOT_FIXED [2]
  - MQOO_BIND_ON_GROUP [2]
  - MQOO_BIND_AS_Q_DEF [2]
- Povolen je pouze *jeden* z následujících možností:
  - MQOO_READ_AHEAD
  - MQOO_NO_READ_AHEAD
  - MQOO_READ_AHEAD_AS_Q_DEF

1. Povolen je pouze *jeden* z následujících možností:
   - MQO_INPUT_EXCLUSIVE
   - MQO_INPUT_SHARED
   - MQO_INPUT_AS_Q_DEF
2. Povolen je pouze *jeden* z následujících možností:
   - MQO_BIND_ON_OPEN
   - MQOO_BIND_NOT_FIXED
   - SKUPINA MQO_BIND_ON_GROUP
   - MQOO_BIND_AS_Q_DEF

   Poznámka: Volby, které jsou vypsané dříve, se navzájem vylučují. Protože však hodnota MQOO_BIND_AS_Q_DEF je nula, její určení s použitím některé z dalších dvou voleb vazby nevede k tomu, že kód příčiny MQRC_OPTIONS_ERROR je. Rozhraní MQOO_BIND_AS_Q_DEF je k dispozici pro dokumentaci programu podpory.

- Je-li zadána hodnota MQOO_SAVE_ALL_CONTEXT, musí být zadána také jedna z voleb MQOO_INPUT_ *.
- Je-li zadán jeden z voleb MQOO_SET_ * _CONTEXT nebo MQOO_PASS_ * _CONTEXT, musí být také zadán parametr MQOO_OUTPUT.
- Je-li zadán parametr MQOO_CO_OP, musí být zadán také MQOROWSROE.
- Je-li zadán MQOO_NO_MULTICAST, musí být také zadán parametr MQOO_OUTPUT.

Volání MQPUT

Pro volby put-message:

- Kombinace MQPMO_SYNCPOINT a MQPMO_NO_SYNCPOINT není povolena.
- Povolen je pouze *jeden* z následujících možností:
  - MQPMO_VÝCHOZÍ_KONTEXT
  - MQPMOTO_NE_KONTEXT

  - MQPMO_PASS_ALL_CONTEXT
  - KONTEXT MQPMO_PASS_IDENTITY_CONTEXT
  - MQPMO_SET_ALL_CONTEXT
  - KONTEXT MQPMO_SET_IDENTITY_CONTEXT
- Povolen je pouze *jeden* z následujících možností:
  - MQPMO_ASYNC_RESPONSE
  - MQPMO_SYNC_RESPONSE
  - MQPMO_RESPONSE_AS_TOPIC_DEF
  - MQPMO_ODEZVA_NA_DOBA_Q_DEF
- Funkce MQPMO_ALTERNATE_USER_AUTHORITY není povolena (je platná pouze na volání MQPUT1).

Volání MQPUT1

U voleb vložení zpráv jsou pravidla stejná jako pro volání MQPUT, s výjimkou následujících:

- Funkce MQPMO_ALTERNATE_USER_AUTHORITY je povolena.
- MQPMO_LOGICAL_ORDER není povoleno.

Volání MQGET

Pro volby get-message:

- Povolen je pouze *jeden* z následujících možností:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_SYNCPOINT
  - MQGMO_SYNCPOINT_IF_PERSISTENT
- Povolen je pouze *jeden* z následujících možností:
  - NEJPRVE MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - PŘÍŠTĚ MQGMO_BROWSE_NEXT
  - MQGMO_MSG_UNDER_CURSOR
- MQGMO_SYNCPOINT není povolen s některou z následujících položek:
  - NEJPRVE MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - PŘÍŠTĚ MQGMO_BROWSE_NEXT
  - MQGMOVÝ_ZÁMEK
  - MQGMO_ODEMKNOUT
- MQGMO_SYNCPINT_IF_PERSISTENT není povolen s některou z následujících položek:
  - NEJPRVE MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - PŘÍŠTĚ MQGMO_BROWSE_NEXT
  - ZPRÁVA MQGMO_COMPLETE_MSG
  - MQGMO_ODEMKNOUT
- MQGMO_MARK_SKIP_BACKOUT vyžaduje zadání MQGMO_SYNCPOINT.
- Kombinace MQGMO_WAIT a MQGMO_SET_SIGNAL není povolena.
- Je-li zadán parametr MQGMO_LOCK, musí být zadán také jeden z následujících:

  - NEJPRVE MQGMO_BROWSE_FIRST
  - MQGMO_BROWSE_MSG_UNDER_CURSOR
  - PŘÍŠTĚ MQGMO_BROWSE_NEXT
- Je-li zadán parametr MQGMO_UNLOCK, jsou povoleny pouze následující hodnoty:
  - MQGMO_NO_SYNCPOINT
  - MQGMO_NO_WAIT

Volání MQCLOSE

Pro volby volání MQCLOSE:

- Kombinace hodnot MQCO_DELETE a MQCO_DELETE_PURGE není povolena.
- Je povolena pouze jedna z následujících možností:
  - MQCO_KEEP_SUB
  - MQCO_REMOVE_SUB

Volání MQSUB

Volby volání MQSUB:

- Musí být uvedena alespoň jedna z následujících možností:
  - MQSO_ALTER
  - MQSO_RESUME
  - VYTVOŘENÉ MQSO_CREATE
- Je povolena pouze jedna z následujících možností:
  - MQSO_TRVALKA
  - MQSO_NON_DURABLE

  Poznámka: Volby, které jsou vypsané dříve, se navzájem vylučují. Protože však hodnota MQSO_NON_DURABLE je 0, zadání hodnoty MQSO_DURABLE nemá za následek chybu MQRC_OPTIONS_ERROR kódu příčiny. Rozhraní MQSO_NON_DURABLE je k dispozici pro dokumentaci programu podpory.
- Kombinace MQSO_GROUP_SUB a MQSO_MANAGED není povolena.
- Funkce MQSO_GROUP_SUB vyžaduje zadání hodnoty MQSO_SET_CORREL_ID.
- Je povolena pouze jedna z následujících možností:
  - MQSO_ANY_USERID
  - ID UŽIVATELE MQSO_FIXED_USERID
- Funkce MQSO_NEW_PUBLICATIONS_ONLY je povolena v kombinaci s:
  - VYTVOŘENÉ MQSO_CREATE
  - MQSO_ALTER, pokud byl nastaven parametr MQSO_NEW_PUBLICATIONS_ONLY v původním odběru
- Kombinace MQSO_PUBLICATIONS_ON_REQUEST a SubLevel větší než 1 není povolena.
- Je povolena pouze jedna z následujících možností:
  - MQSO_WILDCARD_CHAR
  - TÉMA MQSO_WILDCARD_TOPIC
- Objekt MQSO_NO_MULTICAST vyžaduje zadání hodnoty MQSO_MANAGED.

Zprávy příkazů publikování a odběru ve frontě

Aplikace může používat zprávy příkazu produktu MQRFH2 k řízení aplikace publikování/odběru ve frontě.

Aplikace, která používá MQRFH2 pro publikování/odběr, může odeslat následující zprávy příkazu do SYSTEM.BROKER.CONTROL.QUEUE:

- “Odstranit zprávu publikace” na stránce 870
- “Zrušit registraci zprávy odběratele” na stránce 871
- “Publikovat zprávu” na stránce 875
- “Registrovat zprávu odběratele” na stránce 877
- “Požadavek na aktualizaci zprávy” na stránce 882

Pokud zapisujete do fronty aplikace pro publikování/odběr, musíte těmto zprávám porozumět, zprávu odpovědi správce front a deskriptor zprávy (MQMD); prohlédněte si následující informace:

- “Zpráva s odpovědí správce front” na stránce 883
- “Nastavení MQMD pro publikování přeposlané správcem front” na stránce 889
- “Nastavení MQMD ve zprávách odezvy správce front” na stránce 890
- “Kódy příčiny publikování a odběru” na stránce 885

Tyto 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, jsou vlastnosti volitelné a mohou se vyskytnout 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 hlavičkovém souboru cmqpsc.h , který je dodáván se správcem front.

Odstranit zprávu publikace

Příkazová zpráva příkazu **Delete Publication** se odešle správci front z vydavatele nebo z jiného správce front, aby správci front sdělil, že má odstranit veškeré zachované publikace pro zadaná témata.

Tato zpráva se odešle do fronty monitorované rozhraním pro publikování/odběr ve frontě správce front.

Vstupní fronta by měla být fronta, do které byla odeslána původní publikace.

Pokud máte oprávnění pro některé, ale ne všechny, témata, která jsou uvedena ve zprávě příkazu **Delete Publication** , odstraní se pouze ta témata. Zpráva **Broker Response** indikuje, která témata nebudou odstraněna.

Podobně platí, že pokud příkaz **Publish** obsahuje více než jedno téma, příkaz **Delete Publication** odpovídá některým, ale ne všem, z těchto témat odstraňuje pouze publikace určené pro témata, která jsou zadána v příkazu **Delete Publication** .

Podrobnosti o parametrech deskriptoru zpráv (MQMD), které jsou zapotřebí při odesílání příkazových zpráv správci front, naleznete v příručce “Nastavení MQMD pro publikování přeposlané správcem front” na stránce 889 .

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je DeletePub (*MQPSC_DELETE_PUBLIKACE*).

Tato vlastnost musí být uvedena.

Téma > (*MQPSC_TOPIC*)
Hodnota je řetězec, který obsahuje téma, pro které mají být odstraněny zachované publikace. Zástupné znaky lze zahrnout do řetězce k odstranění publikování ve více než jednom tématu.

Tato vlastnost musí být uvedena; může být opakována podle potřeby jako mnoho témat.

DelOpt (*MQPSC_DELETE_OPTION*)
Vlastnost odstranění voleb může mít jednu z následujících hodnot:

`Lokální` (*MQPSC_LOCAL*)

Všechna zachovaná publikování pro určená témata jsou odstraněna v lokálním správci front (tj. správci front, do kterého je tato zpráva odeslána), ať už byla publikována s volbou `Lokální` nebo ne.

Publikování na jiných správcích front nejsou ovlivněny.

Žádné (*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný účinek jako vynechání vlastnosti `DelOpt` . Jsou-li současně zadány jiné volby, hodnota Žádná se ignoruje.

Pokud je tato vlastnost vynechána, budou všechny zachované publikace pro určená témata odstraněny ve všech správcích front v síti bez ohledu na to, zda byly publikovány s volbou `Lokální` .

Příklad

Zde je uveden příklad `NameValueData` pro zprávu příkazu **`Delete Publication`** . Tato akce je použita ukázkovou aplikací k odstranění v lokálním správci front zachované publikování, které obsahuje nejnovější skóre v rámci shody mezi Team1 a Team2.

```
<psc>
  <Command>DeletePub</Command>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
  <DelOpt>Local</DelOpt>
</psc>
```

Zrušit registraci zprávy odběratele

Příkazová zpráva příkazu **`Deregister Subscriber`** se odešle do správce front odběratelem nebo jinou aplikací jménem odběratele, aby označoval, že již nechce přijímat zprávy odpovídající daným parametrům.

Tato zpráva se odešle do systému SYSTEM.BROKER.CONTROL.QUEUE, řídicí fronta správce front. Uživatel musí mít potřebné oprávnění k 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 publikace předané správcem front .

Registrace jednotlivých odběrů může být zrušena zadáním odpovídajícího tématu, bodu odběru a hodnot filtru původního odběru. Pokud některé z hodnot nebyly zadány (tj. byly použity výchozí hodnoty) v původním odběru, měly by být při zrušení registrace odběru vynechány.

Všechny odběry pro odběratele nebo skupinu odběratelů lze zrušit registrací pomocí volby `DeregAll` . Je-li například zadán parametr `DeregAll` spolu s bodem odběru (ale bez tématu nebo filtru), dojde ke zrušení registrace všech odběrů pro odběratele v daném bodu odběru, a to bez ohledu na téma a filtr. Je povolena libovolná kombinace tématu, filtru a bodu odběru; pokud jsou všechny tři zadány pouze jeden odběr, volba `DeregAll` je ignorována.

Zpráva musí být odeslána odběratelem, který registroval odběr, což je potvrzeno kontrolou `ID` uživatele odběratele.

Registrace odběrů může také zrušit registraci administrátorem systému pomocí příkazů MQSC nebo PCF. Avšak odběry registrované s dočasnou dynamickou frontou jsou přidruženy k frontě, nikoli pouze s názvem fronty. Je-li fronta odstraněna, ať už explicitně, nebo aplikací odpojením od správce front, není již možné použít příkaz **`Deregister Subscriber`** k zrušení registrace odběrů pro tuto frontu. Odběru lze zrušit registraci pomocí pracovní plochy vývojářů a správce front jej automaticky odeberou, až se jeho

publikování stane platným přihlášením k odběru, nebo při příštím restartu správce front. Za normálních okolností by aplikace měly zrušit registraci odběrů před odstraněním fronty nebo odpojením od správce front.

Pokud odběratel odešle zprávu pro zrušení registrace odběru a obdrží zprávu s odpovědí, že tato zpráva byla úspěšně zpracována, mohou se některé publikace stále dostat do fronty odběratele, pokud byly zpracovávány správcem front ve stejnou dobu, kdy byla zrušena registrace odběru. Pokud se zprávy neodeberou z fronty, může dojít k nahromadění nezpracovaných zpráv ve frontě odběratele. Pokud aplikace provádí smyčku, která obsahuje volání MQGET s příslušným parametrem `CorrelId` po chvíli spánku, budou tyto zprávy vymazány z fronty.

Podobně platí, že pokud odběratel používá trvalou dynamickou frontu a zruší registraci a zavře frontu s volbou *MQCO_DELETE_PURGE* v rámci volání MQCLOSE, nemusí být fronta prázdná. Pokud nejsou některé publikace ze správce front potvrzeny při odstranění fronty, je návratový kód MQRC_Q_NOT_EMPTY vyvolán voláním funkce MQCLOSE. Aplikace se může tomuto problému vyhnout tak, že bude čas od času spát a znovu ji volat z volání MQCLOSE.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je `DeregSub` (*MQPSC_DEREGISTER_SUBSCRIBER*).

Tato vlastnost musí být uvedena.

Téma (*MQPSC_TOPIC*)
Hodnota je řetězec, který obsahuje téma, jehož registrace má být zrušena.

Tato vlastnost může být volitelně opakována, pokud má být zrušena registrace více témat. Je možné jej vynechat, pokud je parametr `DeregAll` zadán v souboru <RegOpt>.

Zadaná témata mohou být podmnožinou těch, která jsou registrována, pokud odběratel chce zachovat odběry pro jiná témata. Jsou povoleny zástupné znaky, ale řetězec tématu, který obsahuje zástupné znaky, se musí přesně shodovat s odpovídajícím řetězcem, který byl zadán ve zprávě příkazu **`Deregister Subscriber`** .

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Hodnota je řetězec, který uvádí bod odběru, ze kterého se má odběr odpojit.

Tato vlastnost se nesmí opakovat. Je možné jej vynechat, pokud je zadán parametr `< Topic>` nebo je-li parametr `DeregAll` zadán v souboru <RegOpt>. Vynecháte-li tuto vlastnost, dojde k následujícímu:

- Pokud **neurčíte** `DeregAll`, odběry odpovídající vlastnosti `< Topic>` (a vlastnost `< Filter >`, pokud jsou přítomny) se odhlašují z výchozího bodu odběru.
- Pokud uvedete `DeregAll`, zrušení registrace všech odběrů (odpovídajících vlastnostem `< Topic>` a `< Filter >`) se zruší ze všech bodů odběru.

Všimněte si, že výchozí bod odběru explicitně nelze určit. Proto neexistuje způsob, jak zrušit registraci všech odběrů pouze z tohoto bodu odběru; je třeba zadat témata.

SubIdentity (*MQPSC_SUBSCRIPTION_IDENTITY*)
Jedná se o řetězec proměnné délky s maximální délkou 64 znaků. Slouží k reprezentaci aplikace se zájmem o odběr. Správce front uchovává pro každý odběr sadu identit odběratele. Každý odběr může povolit, aby jeho identita byla nastavena pouze na jedinou identitu, nebo na neomezený počet identit.

Pokud je položka `SubIdentity` v sadě identit pro odběr, pak je odebrána ze sady. Je-li sada identit v důsledku tohoto stavu prázdná, bude odběr odebrán ze správce front, pokud není jako hodnota vlastnosti `RegOpt` zadána hodnota `LeaveOnly` . Pokud sada identit stále obsahuje další identity, nebude odběr odebrán ze správce front a tok publikování není přerušen.

Je-li zadána hodnota `SubIdentity` , ale `SubIdentity` není v sadě identit pro odběr, pak příkaz **`Deregister Subscriber`** selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

Filtr (*MQPSC_FILTER*)
Hodnota je řetězec určující filtr, který má být deregistrován. Musí přesně odpovídat, včetně případu a libovolných mezer, filtru odběru, který byl dříve registrován.

Tato vlastnost může být volitelně opakována, pokud má být odregistrován více než jeden filtr. Je možné jej vynechat, pokud je zadán parametr < Topic> nebo je-li parametr DeregAll zadán v souboru <RegOpt>.

Určené filtry mohou být podmnožinou těch, které jsou registrovány, pokud odběratel chce zachovat odběry pro jiné filtry.

RegOpt (*MQPSC_REGISTRATION_OPTION*)
Vlastnost voleb registrace může mít následující hodnoty:

DeregAll
(*MQPSC_DEREGISTER_ALL*)

Registrace všech odpovídajících odběrů registrovaných pro tohoto odběratele je zrušena.

Určíte-li volbu DeregAll, postupujte takto:

- < Topic>, <SubPoint>a < Filter > lze vynechat.
- < Topic> a < Filtr > lze v případě potřeby opakovat.
- <SubPoint> nesmí být opakován.

Pokud **neurčíte** DeregAll, postupujte takto:

- < Topic> musí být zadán a lze jej v případě potřeby opakovat.
- <SubPoint> a < Filter > lze vynechat.
- <SubPoint> nesmí být opakován.
- < Filtr > lze v případě potřeby opakovat.

Pokud se budou opakovat témata a filtry, budou odebrány všechny odběry odpovídající všem kombinacím těchto dvou kombinací. Například příkaz **Deregister Subscriber** , který uvádí tři témata a tři filtry, se pokusí odstranit devět odběrů.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

Identifikátor CorrelId v deskriptoru zpráv (MQMD), který nesmí být nula, se používá k identifikaci odběratele. Musí se shodovat s položkou CorrelId použitou v původním odběru.

FullResp
(*MQPSC_FULL_RESPONSE*)

Je-li zadán parametr FullResp , jsou ve zprávě odpovědi vráceny všechny atributy odběru, pokud příkaz selže.

Je-li zadán parametr FullResp , není v příkazu **Deregister Subscriber** povolena volba DeregAll . Také není možné zadat více témat. Příkaz selže s návratovým kódem *MQRCCF_REC_OPTIONS_ERROR*, v obou případech.

LeaveOnly
(*MQPSC_LEAVE_ONLY*)

Když tuto volbu uvedete s parametrem SubIdentity , který je v sadě identit pro odběr, SubIdentity je odebrán z identity sady pro odběr. Odběr není odebrán ze správce front, a to ani v případě, že výsledná sada identit je prázdná. Pokud se hodnota SubIdentity nenachází v sadě identity, příkaz selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

Je-li zadán parametr LeaveOnly bez prvku SubIdentity, příkaz selže s návratovým kódem *MQRCCF_REC_OPTIONS_ERROR*.

Pokud není zadán ani parametr LeaveOnly , ani 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. Jsou-li současně zadány jiné volby, hodnota Žádná se ignoruje.

IDVariableUser
(*MQPSC_VARIABLE_USER_ID*)

Pokud je zadána identita odběratele (fronta, správce front a correlid), není omezena pouze na jedno ID uživatele. Tento rozdíl se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabrání jakémukoli jinému uživateli, který tuto identitu používá. Pokud se nový odběratel pokusí použít stejnou identitu, vrátí se návratový kód *MQRCCF_DUPLICATE_SUBSCRIPTION* .

Kterýkoli uživatel může odběr upravit nebo zrušit jeho registraci, pokud má vhodné oprávnění, a vyhnout se tak existující kontrole, zda ID uživatele musí odpovídat původnímu odběrateli.

Chcete-li přidat tuto volbu k existujícímu odběru, musí příkaz pocházet ze stejného ID uživatele jako původní odběr.

Je-li odběr pro zrušení registrace nastaven na hodnotu `VariableUserId` , musí být tato hodnota nastavena při zrušení registrace, aby bylo možné určit, který odběr má být odregistrován. Jinak se použije ID uživatele příkazu **`Deregister Subscriber`** k identifikaci odběru. Tato hodnota je přepsána spolu s dalšími identifikátory odběratele, je-li zadán název odběru.

Předvolba, je-li tato vlastnost vynechána, je, že nejsou nastaveny žádné volby registrace.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je název správce front pro frontu odběratele. Musí se shodovat s názvem `QMgrName` použitým v původním odběru.

Je-li tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zpráv (MQMD). Je-li výsledný název prázdný, bude použit výchozí název správce front.

QName (*MQPSC_Q_NAME*)
Hodnota je název fronty odběratele. Musí odpovídat hodnotě `QName` použité v původním odběru.

Je-li tato vlastnost vynechána, výchozí hodnotou je název `ReplyToQ` v deskriptoru zpráv (MQMD), která nesmí být prázdná.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Pokud uvedete `SubName` na příkazu **`Deregister Subscriber`** , hodnota `SubName` má přednost před všemi ostatními poli identifikátoru kromě ID uživatele, pokud není parametr `VariableUserId` nastaven na odběru sám. Není-li parametr `VariableUserID` nastaven, příkaz **`Deregister Subscriber`** bude úspěšný pouze v případě, že se ID uživatele zprávy příkazu shoduje s ID odběru, pokud 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*. Je-li proveden pokus o deregistraci odběru, který má `SubName` pomocí zprávy příkazu, která odpovídá tradiční identitě, ale bez zadání `SubName` , je příkaz úspěšný.

Data SubUserData (*MQPSC_SUBSCRIPTION_USER_DATA*)
Jedná se o textový řetězec proměnné délky. Hodnota je uložena správcem front s odběrem, ale nemá žádný vliv na doručení publikování na odběratele. Hodnota může být změněna opětným registrací na stejném odběru s novou hodnotou. Tento atribut je určen pro použití aplikace.

SubUserData jsou vrácena v informacích o metatématu (MQCACF_REG_SUB_USER_DATA) pro odběr, je-li přítomna data SubUserData.

Příklad

Zde je uveden příklad NameValueData pro zprávu příkazu **Deregister Subscriber** . V tomto příkladu bude ukázková aplikace zrušena registrace svého odběru na témata, která obsahují nejnovější skóre pro všechny shody. Identita odběratele, včetně CorrelId, se převezme z předvoleb v MQMD.

```
<psc>
  <Command>DeregSub</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Publikovat zprávu

Příkazová zpráva příkazu **Publish** se umístí do fronty nebo ze správce front na odběratele, chcete-li publikovat informace na určeném tématu nebo tématech.

Oprávnění k vložení zprávy do fronty a oprávnění k publikování informací na určeném tématu nebo tématech je nezbytné.

Má-li uživatel oprávnění publikovat informace o některých, ale ne všech tématech, jsou k publikování použita pouze tato témata; odpověď s varováním označuje, která témata nebudou použita k publikování.

Pokud má odběratel nějaké odpovídající odběry, správce front předá zprávu produktu **Publish** do front odběratele definovaných v odpovídajících zprávách příkazu **Register Subscriber** .

Podrobnosti o parametrech deskriptoru zpráv (MQMD), které jsou potřebné při odesílání zpráv příkazů do správce front a které jsou použity v případě, že správce front předá publikování odběrateli fronty zpráv, naleznete v tématu Zpráva odpovědi správce front .

Správce front předá zprávu produktu **Publish** ostatním správcům front v síti, které mají odpovídající odběry, pokud se nejedná o lokální publikování.

Data zveřejňovaní, jsou-li nějaká, jsou zahrnuta do textu zprávy. Data mohou být popsána ve složce <mcd> v poli NameValueData záhlaví MQRFH2 .

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je Publikovat (*MQPSC_PUBLISH*).

Tato vlastnost musí být uvedena.

Téma (*MQPSC_TOPIC*)
Hodnota je řetězec, který obsahuje téma, které tuto publikaci kategorizuje. Nejsou povoleny žádné zástupné znaky.

Je třeba přidat téma do seznamu názvů SYSTEM.QPUBSUB.QUEUE.NAMELIST, viz Přidání proudu , kde naleznete pokyny, jak dokončit tuto úlohu.

Tato vlastnost musí být určena a volitelně může být opakována podle potřeby pro tolik témat, kolik je třeba.

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Bod odběru, na kterém je publikace publikována.

V produktu WebSphere Event Broker 6.0je hodnota vlastnosti <SubPoint> hodnotou atributu Bod odběru u uzlu publikování, který zpracovává publikování.

V produktu IBM WebSphere MQ 7.0.1se hodnota vlastnosti <SubPoint> musí shodovat s názvem bodu odběru. Viz Přidání bodu odběru.

PubOpt (*MQPSC_PUBLICATION_OPTION*)
Vlastnost voleb publikování může mít následující hodnoty:

RetainPub
(*MQPSC_RETAIN_PUB*)

Správce front má zachovat kopii publikování. Není-li tato volba nastavena, je publikace odstraněna ihned poté, co správce front odešle publikování všem aktuálním odběratelům.

IsRetainedPub
(*MQPSC_IS_RETAINED_PUB*)

(Může být nastaven pouze správcem front.) Tato publikace byla uchována správcem front. Správce front tuto volbu nastaví na upozornění odběratele, že tato publikace byla publikována dříve a byla zachována, za předpokladu, že byl odběr zaregistrován s volbou `InformIfZachovaná` . Je nastaven pouze v odezvě na zprávu příkazu `Registrovat odběratele` nebo `Požadovat aktualizaci` . Zachovaná publikování, která jsou odeslána přímo odběratelům, nemají tuto sadu voleb.

Lokální
(*MQPSC_LOCAL*)

Tato volba sděluje správci front, že tato publikace nesmí být odeslána jiným správcům front. Všichni odběratelé, kteří jsou registrováni u tohoto správce front, obdrží tuto publikaci, pokud mají odpovídající odběry.

OtherSubs
(*MQPSC_OTHER_SUBS_ONLY*)

Tato volba umožňuje jednodušší zpracování aplikací typu konference, kde vydavatel je také odběratelem stejného tématu. Říká správci front, aby neodeslal publikování do fronty odběratele vydavatele, a to i v případě, že má odpovídající odběr. Fronta odběratele odběratele se skládá z jeho `QMgrName`, `QNamea` volitelného `CorrelId`, jak je popsáno v následujícím seznamu.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

Pole `CorrelId` v MQMD (které nesmí být nula) je součástí fronty odběratele vydavatele v aplikacích, kde je vydavatel také odběratelem.

NONE
(*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný účinek jako vynechání vlastnosti voleb publikování. Jsou-li současně zadány jiné volby, hodnota Žádná se ignoruje.

Můžete mít více než jednu volbu publikace vložením dalších prvků <PubOpt> .

Předvolba, je-li tato vlastnost vynechána, je, že nejsou nastaveny žádné volby publikování.

PubTime (*MQPSC_PUBLISH_TIMESTAMP*)
Hodnota je volitelné časové razítko publikace nastavené vydavatelem. Je 16 znaků dlouhý s formátem:

```
YYYYMMDDHHMMSSTH
```

pomocí univerzálního času. Tyto informace nejsou správcem fronty zkontrolovány před tím, než je zasílána uživatelům.

SeqNum (*MQPSC_SEQUENCE_NUMBER*)
Hodnota je volitelné pořadové číslo nastavené vydavatelem.

Musí být zvýšena o jedničku při každé publikaci. Tento stav však není kontrolován správcem front, který pouze přenáší tyto informace na odběratele.

Pokud jsou publikace ve stejném tématu publikovány do různých propojených správců front, je odpovědností vydavatelů zajistit, aby byla pořadová čísla smysluplná, pokud jsou použita.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je řetězec obsahující název správce front pro frontu odběratele vydavatele, v aplikacích, kde vydavatel je také odběratelem (viz `OtherSubsOnly`).

Je-li tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zpráv (MQMD). Je-li výsledný název prázdný, bude použit výchozí název správce front.

QName (*MQPSC_Q_NAME*)
Hodnota je řetězec obsahující název fronty odběratele vydavatele, v aplikacích, kde vydavatel je 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 jsou některé příklady *NameValueData* pro příkazovou zprávu **Publish** .

První příklad je určen pro publikování odeslané simulátorem shody v ukázkové aplikaci, aby označilo, že došlo ke shodě.

```
<psc>
  <Command>Publish</Command>
  <Topic>Sport/Soccer/Event/MatchStarted</Topic>
</psc>
```

Druhý příklad je pro zachované publikování. Je publikován nejnovější skóre v porovnání mezi Team1 a Team2 .

```
<psc>
  <Command>Publish</Command>
  <PubOpt>RetainPub</PubOpt>
  <Topic>Sport/Soccer/State/LatestScore/Team1 Team2</Topic>
</psc>
```

Registrovat zprávu odběratele

Příkazová zpráva příkazu **Register Subscriber** je odeslána do správce front odběratelem nebo jinou aplikací jménem odběratele a označuje, že se chce přihlásit k odběru jednoho nebo více témat v bodu odběru. Lze také zadat filtr obsahu zpráv.

Vnořené závorky ve výrazech filtru publikování/odběru způsobují pokles výkonu exponenciálním způsobem. Vyvarujte se vnoření závorek do hloubky větší než okolo 6.

Zpráva se odešle na SYSTEM.BROKER.CONTROL.QUEUE, což je řídicí fronta správce front. Je vyžadováno oprávnění k vložení zprávy do této fronty spolu s oprávněním přístupu (nastavovaným administrátorem systému správce front) pro dané téma nebo témata v rámci odběru.

Pokud má uživatel oprávnění k některým, ale ne všem tématům, jsou registrována pouze ta, která jsou s oprávněním registrována; varovná odezva označuje ty, které nejsou zaregistrovány.

Podrobnosti o parametrech deskriptoru zpráv (MQMD), které jsou zapotřebí při odesílání příkazových zpráv správci front, naleznete v příručce “Nastavení MQMD v příkazových zprávách pro správce front” na stránce 888 .

Je-li odpověď na frontu dočasnou dynamickou frontou, správce front při zavření fronty automaticky zruší registraci odběru.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je RegSub (*MQPSC_REGISTER_SUBSCRIBER*). Tato vlastnost musí být uvedena.

Téma (*MQPSC_TOPIC*)
Téma, pro které chce odběratel přijímat publikování. Zástupné znaky lze zadat jako část tématu.

Použijete-li příkaz MQSC **display sub** k prozkoumání odběru vytvořeného tímto způsobem, hodnota značky < Topic> se zobrazí jako vlastnost TOPICSTR odběru.

Tato vlastnost je povinná a lze ji volitelně zopakovat pro tolik témat, kolik je třeba.

SubPoint (*MQPSC_SUBSCRIPTION_POINT*)
Hodnota je bod odběru, ke kterému je odběr připojen.

Je-li tato vlastnost vynechána, bude použit výchozí bod odběru.

V produktu WebSphere Event Broker 6.0se hodnota vlastnosti <SubPoint> musí shodovat s hodnotou atributu Bod odběru u odebíraných uzlů publikování.

V produktu IBM WebSphere MQ 7.0.1se hodnota vlastnosti <SubPoint> musí shodovat s názvem bodu odběru. Viz Přidání bodu odběru.

Filtr (*MQPSC_FILTER*)
Hodnota je výraz SQL, který se používá jako filtr na obsahu zpráv publikování. Pokud se publikování v uvedeném tématu shoduje s filtrem, odešle se odběrateli. Tato vlastnost odpovídá řetězci výběru, který se používá v voláních MQSUB a MQOPEN. Další informace naleznete v tématu Výběr obsahu zprávy .

Je-li tato vlastnost vynechána, nebude k dispozici 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 bez aktuální hodnoty SubName , je do odběru přidán SubName uvedený v tomto příkazu.

Je-li zadáno AddName , je povinné pole SubName , v opačném případě je vrácena chyba MQRCCF_OPTIONS_OPTIONS_ERROR.

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

Identifikátor CorrelId v deskriptoru zpráv (MQMD) se používá při odesílání odpovídajících publikací do fronty odběratele. Hodnota CorrelId nesmí být nula,

FullResp
(*MQPSC_FULL_RESPONSE*)

Pokud jsou uvedeny všechny atributy odběru, vrátí se ve zprávě odpovědi, pokud příkaz selže.

Volba FullResp je platná pouze v případě, že se zpráva příkazu odkazuje na jeden odběr. Proto je v příkazu povoleno pouze jedno téma; v opačném případě příkaz selže s návratovým kódem *MQRCCF_REC_OPTIONS_ERROR*.

InformIfRet
(*MQPSC_INFORM_IF_RETAILed*)

Správce front informuje odběratele v případě, že je publikování uchováno při odeslání zprávy Publikovat v odpovědi na zprávu příkazu **Register Subscriber** nebo **Request Update** . Tento správce front to provede zahrnutím volby publikování IsRetainedPub do zprávy.

JoinExcl
(*MQPSC_JOIN_EXCLUSIVE*)

Tato volba označuje, že zadané SubIdentity by mělo být přidáno jako výlučný člen pro sadu identit pro odběr a že do sady nelze přidat žádné další identity.

Pokud již byla identita sloučena se 'shared' a je jediným záznamem v sadě, sada se změní na výlučný zámek držený touto identitou. Jinak platí, že pokud má odběr aktuálně

jiné identity v sadě identity (se sdíleným přístupem), příkaz selže s návratovým kódem *MQRCCF_SUBSCRIPTION_IN_USE*.

JoinShared
(*MQPSC_JOIN_SHARED*)

Tato volba označuje, že zadaná `SubIdentity` by měla být přidána do sady identit pro odběr.

Je-li odběr momentálně uzamčen výhradně (pomocí volby `JoinExcl`), příkaz selže s návratovým kódem *MQRCCF_SUBSCRIPTION_LOCKED*, pokud identita, která má odběr zamknutou, je stejná identita jako v této zprávě příkazu. V tomto případě je zámek automaticky změněn na sdílený zámek.

Lokální
(*MQPSC_LOCAL*)

Odběr je lokální a není distribuován do jiných správců front v síti. Publikování provedené v jiných správcích front nejsou tomuto odběrateli doručeny, pokud také není k dispozici odpovídající globální odběr.

PouzeNewPubs
(*MQPSC_NEW_PUBS_ONLY*)

Zachovaná publikování, která existují v době, kdy je odběr zaregistrován, nejsou odběrateli odeslány; jsou odeslána pouze nová publikování.

Pokud odběratel znovu zaregistruje a změní tuto volbu tak, že již není nastaven, může být publikování, které již bylo odesláno, odesláno na něj znovu.

NoAlter
(*MQPSC_NO_ALTER*)

Atributy existujícího odpovídajícího odběru se nezmění.

Když se vytváří odběr, tato volba se ignoruje. Všechny ostatní uvedené volby se použijí pro nový odběr.

Má-li položka `SubIdentity` také jednu z voleb spojení (`JoinExcl` nebo `JoinShared`) je tato identita přidána do sady identit bez ohledu na to, zda je zadán parametr `NoAlter` .

NONE
(*MQPSC_NONE*)

Všechny volby registrace mají své výchozí hodnoty.

Je-li odběratel již registrován, jeho volby se resetují na výchozí hodnoty (povšimněte si, že to nemá stejný vliv jako vynechání vlastnosti voleb registrace) a vypršení platnosti odběru je aktualizováno z MQMD zprávy produktu **`Register Subscriber`** .

Jsou-li současně zadány další volby registrace, hodnota Žádná se ignoruje.

NonPers
(*MQPSC_NON_PERSISTENT*)

Publikování odpovídající tomuto odběru se doručí odběrateli jako přechodné zprávy.

Per
(*MQPSC_PERSISTENT*)

Publikování odpovídající tomuto odběru se doručí odběrateli jako trvalé zprávy.

PersAsPub
(*MQPSC_PERSISTENT_AS_PUBLISH*)

Publikování, které odpovídají tomuto odběru, jsou doručeny odběrateli s perzistencí specifikovanou vydavatelem. Toto chování je výchozí.

PersAsFronta
(*MQPSC_PERSISTENT_AS_Q*)

Publikování, které odpovídají tomuto odběru, jsou doručeny odběrateli s perzistencí specifikovanou ve frontě odběratele.

PubOnReqOnly
(*MQPSC_PUB_ON_REQUEST_ONLY*)

Správce front neodesílá publikace odběrateli, kromě odpovědi na zprávu příkazu **Request Update** .

IDVariableUser
(*MQPSC_VARIABLE_USER_ID*)

Pokud je zadána identita odběratele (fronta, správce front a correlid), není omezena pouze na jedno ID uživatele. Tento rozdíl se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabrání jakémukoli jinému uživateli, který tuto identitu používá. Pokud se nový odběratel pokusí použít stejnou identitu, vrátí se stejná identita *MQRCCF_DUPLICATE_SUBSCRIPTION* .

To umožňuje jakémukoli uživateli upravit nebo zrušit registraci odběru, pokud má uživatel vhodné oprávnění. Proto není třeba zkontrolovat, 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 původní odběr.

Je-li odběr příkazu **Request Update** nastaven na hodnotu `VariableUserId` , musí být tato hodnota nastavena při aktualizaci požadavku, aby označovala, na který odběr se bude odkazovat. Jinak se použije ID uživatele příkazu **Request Update** k identifikaci odběru. Tato hodnota je přepsána spolu s dalšími identifikátory odběratele, je-li zadán název odběru.

Pokud se příkazová zpráva **Register Subscriber** bez této sady voleb odkazuje na existující odběr, který má tuto sadu voleb, bude tato volba odebrána z tohoto odběru a ID uživatele odběru je nyní opraveno. Pokud již existuje odběratel, který má stejnou identitu (fronta, správce front a identifikátor korelace), ale s jiným ID uživatele, který je k němu přidružen, příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_IDENTITY* , protože k identitě odběratele může být přidružen pouze jeden ID uživatele.

Je-li vlastnost volby registrace vynechána a odběratel je již registrován, její volby registrace se nezmění a platnost odběru bude aktualizována z deskriptoru MQMD zprávy produktu **Register Subscriber** .

Není-li odběratel již registrován, vytvoří se nový odběr se všemi volbami registrace s použitím výchozích hodnot.

Výchozí hodnoty jsou `PersAsPub` a nejsou nastaveny žádné další volby.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je název správce front pro frontu odběratele, do kterého správce front odesílá odpovídající publikování.

Je-li tato vlastnost vynechána, je výchozí hodnotou název `ReplyToQMgr` v deskriptoru zpráv (MQMD). Je-li výsledné jméno prázdné, standardně se použije správce front `QMgrName`.

QName (*MQPSC_Q_NAME*)
Tato hodnota představuje název fronty odběratele, do níž správce front odesílá odpovídající publikování.

Je-li tato vlastnost vynechána, je výchozím nastavením název `ReplyToQ` v deskriptoru zpráv (MQMD), který nesmí být v tomto případě prázdný.

Je-li fronta dočasná dynamická fronta, dočasné doručení publikací (`NonPers`) musí být zadán ve vlastnosti `<RegOpt>` .

Je-li fronta dočasnou dynamickou frontou, správce front je při uzavření fronty automaticky deregistrován.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Jedná se o název přidělený konkrétnímu odběru. Můžete ji použít místo správce front, fronty a volitelného objektu correlId , abyste se odkazovaly na odběr.

Pokud odběr již existuje s touto hodnotou **SubName** , všechny ostatní atributy odběru (Topic, QMgrName, QName, CorrelId, UserId, RegOpts, UserSubData a Expiry) budou přepsány atributy, jsou-li zadány, které jsou předány nové zprávě příkazu `Registrovat odběratele` . Pokud je však zadán parametr **SubName** bez určeného pole QName a v záhlaví MQMD je určena položka ReplyToQ, fronta odběratele se změní na hodnotu Q ReplyTo.

Pokud odběr, který odpovídá tradiční identitě tohoto příkazu, již existuje, ale nemá žádný **SubName** , příkaz Registrace selže s návratovým kódem *MQRCCF_DUPLICATE_SUBSCRIPTION*, pokud není zadána volba **AddName** .

Pokud se pokusíte změnit existující pojmenovaný odběr pomocí jiného příkazu `Registrovat odběratele` , který určuje stejnou hodnotu **SubName** a hodnoty tématu, QMgrName, QName a CorrelId v novém příkazu odpovídají odlišnému stávajícímu odběru, s definovaným nebo bez definice SubName , příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_SUBSCRIPTION*. Tím se zabrání, aby se dva názvy odběrů odkazovaly na stejný odběr.

SubIdentity (*MQPSC_SUBSCRIPTION_IDENTITY*)
Tento řetězec se používá ke znázornění aplikace, která má zájem o odběr. Jedná se o znakový řetězec proměnné délky s maximální délkou 64 znaků a je volitelný. Správce front uchovává pro každý odběr sadu identit odběratele. Každý odběr může povolit, aby jeho sada identit obsahovala pouze jednu identitu nebo neomezený počet identit (viz volby **JoinShared** a **JoinExcl**).

Příkaz k odběru, který uvádí volbu **JoinShared** nebo **JoinExcl** přidá **SubIdentity** do sady identity odběru, pokud již neexistuje a pokud existující sada identit umožňuje takovou akci; to znamená, že žádný jiný odběratel se nepřipojil výhradně nebo sada identit je prázdná.

Jakákoli změna atributů odběru jako výsledku příkazu `Registrovat odběratele` , ve které je zadán parametr **SubIdentity** , je úspěšný pouze v případě, že by byl jediným členem sady identit pro tento odběr. Jinak dojde k selhání příkazu s návratovým kódem *MQRCCF_SUBSCRIPTION_IN_USE*. Tím zabráníte tomu, aby se atributy odběru změnily, aniž by byli o tom informováni další zainteresovaní odběratelé.

Pokud uvedete znakový řetězec, který je delší než 64 znaků, příkaz selže s návratovým kódem *MQRCCF_SUB_IDENTITY_ERROR*.

Data SubUserData (*MQPSC_SUBSCRIPTION_USER_DATA*)
Jedná se o textový řetězec proměnné délky. Hodnota je uložena správcem front s odběrem, ale nemá žádný vliv na doručení publikování na odběratele. Hodnota může být změněna opětným registrací na stejném odběru s novou hodnotou. Tento atribut je k dispozici pro použití aplikace.

Položka **SubUserData** se vrátí v informacích o metatématu (*MQCACF_REG_USER_DATA*) pro odběr, pokud je přítomen.

Pokud uvedete více než jednu z hodnot voleb registrace `NonPers, PersAsPub, PersAsQueue, and Pers`, použije se pouze poslední z nich. Tyto volby nelze kombinovat v jednotlivých odběrech.

Příklad

Zde je uveden příklad `NameValueData` pro zprávu příkazu **`Register Subscriber`** . V ukázkové aplikaci služba výsledků pomocí této zprávy zaregistruje odběr na témata obsahující nejnovější skóre ve všech shodách s nastavenou volbou 'Trvalý jako publikování'. Identita odběratele, včetně `CorrelId`, se převezme z předvoleb v MQMD.

```
<psc>
  <Command>RegSub</Command>
  <RegOpt>PersAsPub</RegOpt>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Požadavek na aktualizaci zprávy

Zpráva příkazu **Request Update** se odešle z odběratele do správce front a požádá o aktuální zachované publikace pro zadané téma a bod odběru, které odpovídají danému (volitelnému) filtru.

Tato zpráva je odeslána na adresu *SYSTEM.BROKER.CONTROL.QUEUE*, řídicí fronta správce front. Je vyžadováno oprávnění k vložení zprávy do této fronty, kromě oprávnění k přístupu pro dané téma v rámci aktualizace požadavku; tento stav je nastaven administrátorem systému správce front.

Tento příkaz se obvykle používá, pokud odběratel určil volbu Pub0nReq0nly , když je registrován. Má-li správce front nějaké odpovídající zachované publikace, jsou odeslány odběrateli. Pokud správce front nemá žádné odpovídající zachované publikace, požadavek selže s návratovým kódem *MQRCCF_NO_RETAINED_MSG*. Žadatel musí již dříve registrovaný odběr se stejným názvem tématu, SubPointa filtrem.

Vlastnosti

Příkaz (*MQPSC_COMMAND*)
Hodnota je ReqUpdate (*MQPSC_REQUEST_UPDATE*). Tato vlastnost musí být uvedena.

Téma (*MQPSC_TOPIC*)
Hodnota je téma, které odběratel požaduje; jsou povoleny zástupné znaky.

Tato vlastnost musí být uvedena, 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, bude použit výchozí bod odběru.

Filtr (*MQPSC_FILTER*)
Hodnota je výrazem ESQL, který se používá jako filtr na obsahu zpráv publikování. Pokud se publikování v uvedeném tématu shoduje s filtrem, odešle se odběrateli.

Vlastnost < Filter > by měla mít stejnou hodnotu jako ta, která byla zadána v původním odběru, pro který nyní žádáte o aktualizaci.

Je-li tato vlastnost vynechána, nebude k dispozici žádné filtrování obsahu.

RegOpt (*MQPSC_REGISTRATION_OPTION*)
Vlastnost voleb registrace může mít následující hodnotu:

CorrelAs
(*MQPSC_CORREL_ID_AS_IDENTITY*)

Hodnota CorrelId v deskriptoru zpráv (MQMD), která nesmí být nula, se používá při odesílání odpovídajících publikací do fronty odběratele.

NONE
(*MQPSC_NONE*)

Všechny volby mají své výchozí hodnoty. To má stejný účinek jako vynechání vlastnosti <RegOpt> . Jsou-li současně zadány jiné volby, hodnota Žádná se ignoruje.

IDVariableUser
(*MQPSC_VARIABLE_USER_ID*)

Pokud je zadána identita odběratele (fronta, správce front a correlid), není omezeno pouze na jedno ID uživatele. Tento rozdíl se liší od existujícího chování správce front, který přidružuje ID uživatele původní registrační zprávy k identitě odběratele, a od té doby zabrání 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 libovolnému uživateli upravit nebo zrušit registraci odběru v případě, že mají odpovídající oprávnění. Proto není třeba 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 původní odběr.

Je-li odběr příkazu **Request Update** nastaven na hodnotu VariableUserId , musí být tato hodnota nastavena při aktualizaci požadavku, aby označovala, na který odběr se bude odkazovat. Jinak se použije ID uživatele příkazu **Request Update** k identifikaci odběru. Tato hodnota je přepsána spolu s dalšími identifikátory odběratele, je-li zadán název odběru.

Předvolba, je-li tato vlastnost vynechána, je, že nejsou nastaveny žádné volby registrace.

QMgrName (*MQPSC_Q_MGR_NAME*)
Hodnota je název správce front pro frontu odběratele, do kterého správce front odesílá odpovídající zachované publikování.

Je-li tato vlastnost vynechána, je výchozí hodnotou název ReplyToQMgr v deskriptoru zpráv (MQMD). Je-li výsledné jméno prázdné, standardně se použije správce front QMgrName.

QName (*MQPSC_Q_NAME*)
Tato hodnota představuje název fronty odběratele, do níž správce front odesílá odpovídající zachované publikování.

Je-li tato vlastnost vynechána, je výchozím nastavením název ReplyToQ v deskriptoru zpráv (MQMD), který nesmí být v tomto případě prázdný.

SubName (*MQPSC_SUBSCRIPTION_NAME*)
Jedná se o název přidělený konkrétnímu odběru. Pokud je hodnota zadaná v příkazu **Request Update** , hodnota SubName má přednost před všemi ostatními poli identifikátoru kromě ID uživatele, pokud není parametr VariableUserId nastaven na samotný odběr. Není-li parametr VariableUserId nastaven, příkaz *Request Update* je úspěšný pouze v případě, že se ID uživatele zprávy příkazu shoduje s ID uživatele odběru. Pokud se ID uživatele zprávy příkazu neshoduje s identifikátorem uživatele odběru, příkaz selže s návratovým kódem *MQRCCF_DUPLICATE_IDENTITY*.

Je-li nastavena hodnota VariableUserId a ID uživatele se liší od ID odběru, bude příkaz úspěšný, pokud má ID uživatele nové zprávy příkazu oprávnění k procházení fronty proudu a vložení do fronty odběratele odběru. Jinak dojde k selhání příkazu s návratovým kódem *MQRCCF_NOT_AUTHORIZED*.

Existuje-li odběr, který odpovídá tradiční identitě tohoto příkazu, ale nemá žádné SubName, příkaz **Request Update** selže s návratovým kódem *MQRCCF_SUB_NAME_ERROR*.

Je-li učiněn pokus o aktualizaci pro odběr, který má SubName pomocí zprávy příkazu, která se shoduje s tradiční identitou, ale bez zadání SubName , příkaz uspěje.

Příklad

Zde je uveden příklad NameValueData pro zprávu příkazu **Request Update** . V ukázkové aplikaci služba výsledků použije tuto zprávu k vyžádání zachovaných publikování obsahujících nejnovější skóre pro všechny týmy. Identita odběratele, včetně CorrelId, se převezme z předvoleb v MQMD.

```
<psc>
  <Command>ReqUpdate</Command>
  <RegOpt>CorrelAsId</RegOpt>
  <Topic>Sport/Soccer/State/LatestScore/#</Topic>
</psc>
```

Zpráva s odpovědí správce front

Zpráva **Queue Manager Response** se odešle ze správce front do ReplyToQ vydavatele nebo odběratele, který označuje úspěch nebo selhání zprávy příkazu přijaté správcem front, pokud deskriptor zprávy příkazu určil, že je vyžadována odpověď.

Zpráva odpovědi je obsažena v poli DataNameValueData záhlaví MQRFH2 ve složce produktu <pscr> .

V případě varování nebo chyby obsahuje zpráva odezvy složku <psc> ze zprávy příkazu a také složku <pscr> . Data zprávy, pokud nějaká jsou, nejsou obsažena ve zprávě odezvy správce front. V případě

chyby nebyla zpracována žádná zpráva, která způsobila chybu; v případě varování mohla být některá zpráva zpracována úspěšně.

Pokud dojde k selhání při odesílání odezvy:

- U zpráv publikování se správce front pokouší odeslat odpověď do fronty nedoručených zpráv produktu IBM MQ , pokud dojde k selhání operace MQPUT. To umožňuje odeslání publikování odběratelům i v případě, že odpověď nelze odeslat zpět vydavateli.
- Pro ostatní zprávy, nebo pokud nelze odpověď publikování odeslat do fronty nedoručených zpráv, se zaprotokoluje chyba a zpráva příkazu se za normálních okolností vrátí zpět. To, zda k tomu dojde, závisí na tom, jak byl uzel MQInput konfigurován.

Vlastnosti

Dokončení (*MQPSCR_COMPLEION*)
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 příkazovou zprávu, pokud tento příkaz vytvořil kód dokončení `varování` nebo `chyba`. Obsahuje vlastnost < `Reason`> a může obsahovat další vlastnosti, které označují příčinu varování nebo chyby.

V případě jedné nebo více chyb existuje pouze jedna složka odezvy, která označuje pouze příčinu první chyby. V případě jednoho nebo více varování existuje složka odpovědí pro každé varování.

Příčina (*MQPSCR_REASON*)
Kód příčiny, který kvalifikují kód dokončení, je-li kód dokončení `varování` nebo `chyba`. Je nastaven na jeden z kódů chyby uvedených v následujícím příkladu. Vlastnost < `Reason`> je obsažena ve složce < `Response`> . Za kódem příčiny může následovat libovolná platná vlastnost ze složky <psc> (například název tématu) s uvedením příčiny chyby nebo varování. Pokud získáte kód příčiny? ???, zkontrolujte správnost dat, například odpovídající lomené závorky (< >).

Příklady

Zde je několik příkladů položek `NameValueData` ve zprávě **`Queue Manager Response`** . Úspěšná odezva může být následující:

```
<pscr>
  <Completion>ok</Completion>
</pscr>
```

Zde je příklad selhání odpovědi; selhání je chybou filtru. První řetězec `NameValueData` obsahuje odezvu; druhý příkaz 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 (v důsledku neautorizovaných témat). 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 a odběru

Tyto kódy příčiny mohou být vráceny v poli Příčina ve složce <pscr> odezvy publikování/odběru. Jsou zde také uvedeny konstanty, které lze použít ke znázornění těchto kódů v programovacích jazycích C nebo C + +.

Konstanty MQRC_ vyžadují hlavičkový soubor IBM MQ cmqc.h . Konstanty MQRCCF_ vyžadují soubor záhlaví IBM MQ cmqcfc.h (kromě souborů *MQRCCF_FILTER_ERROR* a *MQRCCF_WRONG_USER*, které vyžadují soubor záhlaví cmqpsc.h).

Kód příčiny a text	Vysvětlení	Vydal(a)
2336 CHYBA PŘÍKAZU MQRC_RFH_COMMAND_ERROR	Platné hodnoty pro pole < Command> ve složce <psc> jsou: RegSub, DeregSub, Publish, DeletePuba ReqUpdate. Všechny ostatní hodnoty mají za následek vydání tohoto kódu chyby.	Jakýkoli příkaz
2337 MQRC_RFH_ PARM_ERROR	Pro složky <psc> a <mcd> jsou nastaveny platné parametry, které lze v rámci těchto složek zadat. Zkontrolujte popisy těchto složek a ujistěte se, že jste neuvedli nesprávné parametry.	Jakýkoli příkaz
2338 MQRC_RFH_DUPLICATE_PARM	Některé parametry (například téma) ve složce <psc> mohou být opakovány, ale jiné (například příkaz) nelze opakovat. Zkontrolujte, zda jste nekopírovali neopakovatelný parametr.	Jakýkoli příkaz
2339 CHYBÍ MQRC_RFH_PARM_MISSING	Některé parametry ve složkách <psc> nebo <mcd> jsou volitelné a mohou být vynechány; některé jsou povinné a nesmí být vynechány. Zkontrolujte, zda jste zahrnuli všechny povinné parametry do složek <psc> a <mcd> .	Jakýkoli příkaz

Kód příčiny a text	Vysvětlení	Vydal(a)
2551 MQRC_SELECTION_NOT_AVAILABLE	Nebyl k dispozici žádný poskytovatel rozšířeného výběru zpráv, který určuje, kteří odběratelé s uvedeným filtrem by měli obdržet publikování.	Publikovat, registrovat odběratele a žádost o aktualizaci
	Nebyl k dispozici žádný poskytovatel rozšířeného výběru zpráv pro zpracování filtru určeného odběratele.	Registrovat odběratele a aktualizaci požadavku
2554 CHYBA MQRC_CONTENT_ERROR	Poskytovatel rozšířeného výběru zpráv zjistil chybu v aktuální nebo zachované publikaci.	Publikovat a aktualizovat požadavek
3008 PŘÍKAZ MQRCCF_COMMAND_FAILED	Došlo k vnitřní chybě, která zabránila správnému provedení příkazu. K chybě může dojít, pokud je příkaz znovu zadán. Systémový protokol událostí pro správce front obsahuje informace, které mají být použity při ohlašování problému pro produkt IBM.	Jakýkoli příkaz
3072 CHYBA MQRCCF_TOPIC_ERROR	Jedna nebo více hodnot zadaných pro parametr Topic je nesprávná. Zkontrolujte, zda se vaše hodnoty pro téma shodují s uvedenými omezeními.	Jakýkoli příkaz
3073 MQRCCF_NOT_REGISTERED	Kombinace SubPoint, tématu a filtru, kterou jste zadali ve svém příkazu DeregSub nebo ReqUpdate , nebyla buď kombinací, se kterou jste již byli registrováni, nebo u příkazu DeregSub , pokud byla zadána volba DeregAll , nebyla použita žádná z vlastností SubPoint, Topic nebo Filter, která nebyla použita pro zrušení registrace odběru.	Zrušit registraci příkazů odběratele a žádosti o aktualizaci
3074 CHYBA MQRCCF_Q_MGR_NAME_ERROR	Určený správce front byl neplatný nebo správce front nebyl k dispozici nebo neexistoval.	Zrušit registraci odběratele, publikovat, registrovat odběratele a příkazy pro aktualizaci požadavků
3076 CHYBA MQRCCF_Q_NAME_ERROR	Zadaný název fronty je neplatný nebo fronta v zadaném správci front neexistuje.	Zrušit registraci odběratele, publikovat, registrovat odběratele a příkazy pro aktualizaci požadavků
3077 ZPRÁVA MQRCCF_NO_RETAINED_MSG	Nebyly nalezeny žádné uchované zprávy pro určené téma. To může nebo nemusí být chyba, v závislosti na návrhu vašeho aplikačního programu.	Příkaz pro aktualizaci požadavku

Kód příčiny a text	Vysvětlení	Vydal(a)
3079 MQRCCF_INCORRECT_Q.	Příkazy RegSub, DeregSuba ReqUpdate se vždy odesílají do systému SYSTEM.BROKER.CONTROL.QUEUE fronty správce front, pro kterou jsou určeny. Příkazy publikování a odstranění publikování se odesílají 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.	Jakýkoli příkaz
3080 CHYBA MQRCCF_CORREL_ID_ERROR	Zadali jste ID CorrelAsjako jednu z vašich parametrů RegOpt . Pole CorrelId deskriptoru MQMD však neobsahuje platný korelační identifikátor (to znamená, že je nastaven na hodnotu MQCI_NONE).	Zrušit registraci odběratele a registrovat příkazy odběratele
3081 AUTORIZOVANÝ OBJEKT MQRCCF_NOT_AUTHORIZED	Nemáte autorizaci k provedení požadované akce. Nastavení autorizace pro správce front je řízeno administrátorem systému pomocí editoru Hierarchie témat.	Publikování a registrace příkazů odběratele
3083 CHYBA OBJEKTU MQRCCF_REG_OPTIONS_ERROR	Uvedli jste nerozpoznaný parametr RegOpt ve složce <psc> , který obsahuje příkaz RegSub nebo DeregSub .	Zrušit registraci odběratele a registrovat příkazy odběratele
3084 CHYBA MQRCCF_PUT_OPTIONS_ERROR	Zadali jste nerozpoznaný parametr PubOpt ve složce <psc> , která obsahuje příkaz Publikovat.	Příkaz publish
3087 CHYBA MQRCCF_DEL_OPTIONS_ERROR	Uvedli jste nerozpoznaný parametr DelOpt ve složce <psc> , která obsahuje váš příkaz DeletePub .	Příkaz Odstranit publikování
3150 CHYBA MQRCCF_FILTER_ERROR	Hodnota uvedená pro parametr filtru je neplatná. Zkontrolujte sekci, která popisuje platnou syntaxi výrazů filtru a ujistěte se, že váš výraz odpovídá.	Zrušit registraci odběratele, registrovat odběratele a příkazy pro aktualizaci požadavku
3151 UŽIVATEL MQRCCF_WRONG_USER	Odběr, který odpovídá zadanému odběru, již existuje; avšak byl registrován jiným uživatelem. Registrace odběru může být změněna nebo zrušena pouze uživatelem, který jej původně zaregistroval.	Zrušit registraci odběratele, registrovat odběratele a příkazy pro aktualizaci požadavku
3152 DUPLICITNÍ_ODBĚR MQRCCF_DUPLICATION	Odpovídající odběr již existuje s jiným názvem odběru.	

Kód příčiny a text	Vysvětlení	Vydal(a)
3153 CHYBA MQRCCF_SUB_NAME_ERROR	Formát názvu odběru buď není platný, nebo již existuje odpovídající odběr bez názvu odběru.	
3154 CHYBA OBJEKTU MQRCCF_SUB_IDENTITY_ERROR	Parametr identity odběru je chybný. Buď dodaná hodnota překračuje maximální povolenou délku, nebo identita odběru není momentálně členem sady identity odběru a nebyla uvedena volba registrace sloučení.	
3155 MQRCCF_SUBSCRIPTION_IN_USE	Byl proveden pokus o úpravu nebo zrušení registrace odběru u člena sady identit, když nebyl jediným členem této sady.	
3156 MQRCCF_SUBSCRIPTION_LOCKED	Odběr je aktuálně výlučně zamknut jinou identitou.	
3157 MQRCCF_ALREADY_JOINED	Byla zadána volba registrace sloučení, ale identita odběratele již byla členem sady identit odběru.	

Nastavení MQMD v příkazových zprávách 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 ji lze nastavit na jakoukoli 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*. Funkce *MQRC_MSG_TYPE_ERROR* bude vrácena, pokud položka `MsgType` není nastavena na jednu z těchto hodnot.

Parametr `MsgType` by měl být nastaven na *MQMT_REQUEST* pro příkazovou zprávu, pokud je odpověď vždy povinná. Parametry MQRO_PAN a MQRO_NAN v poli `Sestava` nejsou v tomto případě významné.

Je-li parametr `MsgType` nastaven na hodnotu *MQMT_DATAGRAM*, závisí odpovědi na nastavení parametrů MQRO_PAN a MQRO_NAN v poli `Sestava` :

- Funkce MQRO_PAN sama znamená, že správce front odešle odezvu pouze v případě, že je příkaz úspěšný.
- Funkce MQRO_NAN sama znamená, že správce front odešle odezvu pouze v případě, že došlo k selhání příkazu.
- Je-li příkaz dokončen s varováním, odešle se odpověď, pokud je nastavena hodnota MQRO_PAN nebo MQRO_NAN.
- MQRO_PAN + MQRO_NAN znamená, že správce front odešle odpověď bez ohledu na to, zda je příkaz úspěšný nebo neúspěšný. To má stejný efekt z perspektivy správce front jako nastavení `MsgType` na hodnotu MQMT_REQUEST.
- Pokud není nastaven ani MQRO_PAN ani MQRO_NAN, žádná odpověď se nikdy neodešle.

Formát
Nastavte na MQFMT_RF_HEADER_2

MsgId
Toto pole je obvykle nastaveno na hodnotu MQMI_NONE, takže správce front vygeneruje jedinečnou hodnotu.

CorrelId
Toto pole může být nastaveno na jakoukoli hodnotu. Pokud identita odesílatele obsahuje `CorrelId`, uveďte tuto hodnotu spolu s parametrem MQRO_PASS_CORREL_ID v poli `Sestava` , abyste se ujistili, že je nastavena ve všech zprávách odpovědi odeslaných správcem front odesílateli.

ReplyToQ
Toto pole definuje frontu, do níž mají být odesílány odpovědi, pokud nějaké existují. Může se jednat o frontu odesílatele; ta má tu výhodu, že parametr `QName` může být z této zprávy vynechán. 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 vlastní název.

Nastavení MQMD pro publikování přeposlané správcem front

Správce front používá tato nastavení polí v deskriptoru zpráv (MQMD), když odesílá publikování odběrateli. Všechna ostatní pole v deskriptoru MQMD jsou nastavena na jejich výchozí hodnoty.

Sestava
Volba `Sestava` je nastavena na hodnotu MQRO_NONE.

MsgType
Parametr `MsgType` je nastaven na hodnotu MQMT_DATAGRAM.

Vypršení
Volba `Vypršení platnosti` je nastavena na hodnotu ve zprávě `Publikovat` přijaté od vydavatele. V případě zachované zprávy se zbývající čas snižuje o přibližný čas, kdy byla zpráva ve správci front.

Formát
Volba `Formát` je nastavena na hodnotu MQFMT_RF_HEADER_2 .

MsgId
Položka `MsgId` je nastavena na jedinečnou hodnotu.

CorrelId
Je-li položka `CorrelId` součástí identity odběratele, jedná se o hodnotu specifikovanou odběratelem při registraci. Jinak se jedná o nenulovou hodnotu zvolenou správcem front.

Priorita
`Priorita` přebírá hodnotu nastavenou vydavatelem, nebo jako vyřešená, pokud vydavatel uvedl MQPRI_PRIORITY_AS_Q_DEF.

Trvání
`Perzistence` přebírá hodnotu nastavenou vydavatelem, nebo jako vyřešená, pokud vydavatel uvedl MQPER_PERSISTENCE_AS_Q_DEF, pokud není ve zprávě `Register Subscriber` pro odběratele, na který se tato publikace odesílá, jinak uvedeno jinak.

ReplyToQ
`ReplyToQ` je nastaveno na mezery.

ReplyToQMgr
Parametr `ReplyToQMgr` je nastaven na název správce front.

UserIdentifier
`UserIdentifier` je identifikátor uživatele odběratele, který je nastaven, když je odběratel registrován.

AccountingToken
`AccountingToken` je účetní token odběratele, jak je nastaven, když je odběratel poprvé registrován.

ApplIdentityData
`Data aplikaceApplIdentity` jsou data identity aplikace odběratele, která byla nastavena při první registraci odběratele.

PutApplType
Hodnota PutApplType je nastavena na hodnotu MQAT_BROKER.

PutApplName
Hodnota PutApplName je nastavena na prvních 28 znaků názvu správce front.

PutDate
PutDate je datum, kdy byla zpráva vložena.

PutTime
PutTime je čas, kdy byla zpráva vložena.

ApplOriginData
Hodnota ApplOriginData je nastavena na mezery.

Nastavení MQMD ve zprávách odezvy 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 jejich výchozí hodnoty.

Sestava
Sestava je nastavena na samé nuly.

MsgType
Parametr MsgType je nastaven na hodnotu MQMT_REPLY.

Formát
Volba Formát je nastavena na hodnotu MQFMT_RF_HEADER_2 .

MsgId
Nastavení hodnoty MsgId závisí na volbách Report v původní zprávě příkazu. Ve výchozím nastavení je hodnota nastavena na hodnotu MQMI_NONE, takže správce front vygeneruje jedinečnou hodnotu.

CorrelId
Nastavení parametru CorrelId závisí na volbách Report v původní zprávě příkazu. Standardně to znamená, že hodnota CorrelId je nastavena na stejnou hodnotu jako MsgId zprávy příkazu. Tento příkaz lze použít ke korelaci příkazů s jejich odezvami.

Priorita
Priorita je nastavena na stejnou hodnotu jako v původní zprávě příkazu.

Trvání
Perzistence je nastavena na hodnotu nastavenou v původní zprávě příkazu.

Vypršení
Volba Vypršení je nastavena na stejnou hodnotu jako v původní zprávě příkazu, kterou obdržel správce front.

PutApplType
Hodnota PutApplType je nastavena na hodnotu MQAT_BROKER.

PutApplName
Hodnota PutApplName je nastavena na prvních 28 znaků názvu správce front.

Ostatní pole kontextu jsou nastavena jako generovaná parametrem MQPMO_PASS_IDENTITY_CONTEXT.

Kódování počítače

Tento oddíl popisuje strukturu pole *Encoding* v deskriptoru zpráv.

Souhrn polí ve struktuře viz “MQMD-Deskriptor zpráv” na stránce 419 .

Pole *Encoding* je 32bitové celé číslo, které je rozděleno do čtyř samostatných podpolí; tato podpole identifikují:

- Kódování použité pro binární celá čísla
- Kódování použité pro packed-decimal celá čísla
- Kódování použité pro čísla s pohyblivou řádovou čárkou

- Vyhrazené bity

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Jsou definovány následující masky:

MQENC_INTEGER_MASK
Maska pro kódování binary-integer.

Toto podpole zabírá v poli *Encoding* bitové pozice 28 až 31.

MQENC_DECIMAL_MASK
Maska pro kódování packed-decimal-integer.

Toto podpole zabírá v poli *Encoding* bitové pozice 24 až 27.

MQENC_FLOAT_MASK.
Maska pro kódování s pohyblivou řádovou čárkou

Toto podpole zaujímá bitové pozice 20 až 23 v poli *Encoding* .

MAQ_REZERVOVANÁ_MASKA
Maska pro rezervované bity.

Toto podpole zabírá v poli *Encoding* bitové pozice 0 až 19.

Kódování Binary-integer

Pro kódování binary-integer jsou platné následující hodnoty:

MQENC_INTEGER_UNDEFINED
Binární celá čísla jsou znázorněna pomocí nedefinovaného kódování.

MQENC_INTEGER_NORMAL
Binární celá čísla jsou reprezentována konvenčním způsobem:

- Nejméně významný bajt v čísle má nejvyšší adresu kteréhokoli z bajtů v čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu je přilehlý k bajtu s další vyšší adresou; nejvíce významný bit v každém bajtu je přilehlý k bajtu s další nižší adresou

MQENC_INTEGER_REVERSED
Binární celá čísla jsou reprezentována stejným způsobem jako MQENC_INTEGRER_NORMAL, ale s bajty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako MQENC_INTEGRER_NORMAL.

Packed-decimal-integer kódování

Pro kódování packed-decimal-integer jsou platné následující hodnoty:

MQENC_DECIMAL_UNDEFINED
Pakovaný-desítková čísla jsou reprezentována pomocí nedefinovaného kódování.

MQENC_DECIMAL_NORMAL
Packed-decimal celá čísla jsou reprezentována v konvenčním způsobem:

- Každá desetinná číslice v tisknutelném tvaru čísla je vyjádřena v pakovaném desítkovém zápisu jedinou hexadecimální číslicí v rozsahu X' 0 'až X' 9 '. Každá hexadecimální číslice zabírá čtyři bity, a tak každý bajt v pakovaném dekadickém čísle představuje dvě desetinná místa v tisknutelném tvaru čísla.
- Nejméně významný bajt v packed-decimal number je bajt, který obsahuje nejméně významnou desetinnou číslici. V tomto bajtu obsahují nejvýznamnější čtyři bity nejméně významnou desítkovou číslici a nejméně významné čtyři bity obsahují znaménko. Znaménko je buď X'C '(kladné), X 'D' (negativní), nebo X'F ' (nepodepsaný).

- Nejméně významný bajt v čísle má nejvyšší adresu kteréhokoli z bajtů v daném čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu je přilehlý k bajtu s další vyšší adresou; nejvíce významný bit v každém bajtu je přilehlý k bajtu s další nižší adresou.

MQENC_DECIMAL_REVERSED
Packed-decimal integer are represented in the same way as MQENC_DECIMAL_NORMAL, but with the bytes arranged in reverse order. Bity v každém bajtu jsou uspořádány stejným způsobem jako MQENC_DECIMAL_NORMAL.

Kódování čísel s pohyblivou řádovou čárkou

Pro kódování s pohyblivou řádovou čárkou jsou platné následující hodnoty:

MQENC_FLOAT_UNDEFINED
Čísla s pohyblivou řádovou čárkou jsou reprezentována nedefinovaným kódováním.

MQENC_FLOAT_IEEEE_NORMAL
Čísla s pohyblivou řádovou čárkou jsou znázorněna pomocí standardního IEEE[4]formát pohyblivé řádové čárky, s bajty uspořádanými následujícím způsobem:

- Nejméně významný bajt v mantisy má nejvyšší adresu libovolného z bajtů v počtu; bajt obsahující exponent má nejnižší adresu.
- Nejméně významný bit v každém bajtu je přilehlý k bajtu s další vyšší adresou; nejvíce významný bit v každém bajtu je přilehlý k bajtu s další nižší adresou

Podrobnosti o kódování typu float se standardem IEEE lze nalézt ve standardu IEEE Standard 754.

MQENC_FLOAT_IEEE_OBRÁCENÝ
Čísla s pohyblivou řádovou čárkou jsou znázorněna stejným způsobem jako MQENC_FLOAT_IEEE_NORMAL, ale s bajty uspořádanými v opačném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako MQENC_FLOAT_IEEEE_NORMAL.

MQENC_FLOAT_S390
Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardního formátu s pohyblivou řádovou čárkou System/390 . Používá se také v systému System/370.

Konstruování kódování

Chcete-li vytvořit hodnotu pro pole *Encoding* v produktu MQMD, je možné přidat odpovídající konstanty, které popisují požadovaná kódování (nepřidávat stejnou konstantu vícekrát než jednou), nebo kombinovat pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Pokud je použita metoda Whichever, zkombinujte pouze jedno z kódování MQENC_INTEGRER_ * s jedním kódováním MQENC_DECIMAL_ * a jedním kódováním MQENC_FLOAT_ *.

Analyzování kódování

Pole *Encoding* obsahuje podpole; z toho důvodu musí aplikace, které mají prozkoumat celé celé číslo, pakované desetinné číslo nebo plovoucí kódování, použít jednu z popsaných technik.

Použití bitových operací

Pokud programovací jazyk podporuje bitové operace, proveďte následující kroky:

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - MQENC_INTEGRER_MASK pro kódování binárních celých čísel
   - MQENC_DECIMAL_MASK pro kódování dekadického celého čísla

[4] Institute of Electrical and Electronics Engineers

- MQENC_FLOAT_MASK pro kódování čísel s pohyblivou řádovou čárkou

Volejte hodnotu A.

2. Zkombinujte pole *Encoding* s A pomocí bitové AND operace; volejte výsledek B.
3. B je požadované kódování a lze jej testovat pro rovnost s každou z hodnot, které jsou platné pro daný typ kódování.

Použití aritmetiky

Pokud jazyk programovacího jazyka *nepodporuje* bitové operace, proveďte následující kroky pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - 1 pro kódování binárního celého čísla
   - 16 pro pakované dekadické celé kódování
   - 256 pro kódování čísel s pohyblivou řádovou čárkou

   Volejte hodnotu A.
2. Dělí hodnotu pole *Encoding* hodnotou A ; volejte výsledek B.
3. Dělí se B o 16; volejte výsledek C.
4. Multiply C od 16 a odečítat od B ; volejte výsledek D.
5. Násobení D podle A ; volejte výsledek E.
6. E je požadované kódování a lze jej testovat pro rovnost s každou z hodnot, které jsou platné pro daný typ kódování.

Souhrn kódování architektury počítače

Kódování pro počítačové architektury jsou zobrazeny v Tabulka 632 na stránce 893.

Tabulka 632. Souhrn kódování pro počítačové architektury

Architektura počítače	Kódování binárních celých čísel	Kódování komprimovaného dekadického celého čísla	Kódování čísel s pohyblivou řádovou čárkou
IBM i	normální	normální	Normální IEEE
Intel x86	Převrácené	Převrácené	IEEE převrácen
PowerPC	normální	normální	Normální IEEE
System/390	normální	normální	System/390

Volby sestav a příznaky zpráv

Tento oddíl popisuje pole *Report* a *MsgFlags* , která jsou součástí deskriptoru zpráv MQMD určeného na voláních MQGET, MQPUT a MQPUT1 .

Témata v této sekci popisují:

- Struktura pole sestavy a způsob, jakým je správce front zpracovává.
- Jak aplikace analyzuje pole sestavy
- Struktura pole s příznaky zprávy

Další informace o deskriptoru zpráv MQMD viz “MQMD-Deskriptor zpráv” na stránce 419.

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 zamítnuty, pokud je lokální správce front nerozpozná
- Volby sestavy, které jsou vždy akceptovány, i tehdy, když je lokální správce front nerozpozná
- Volby sestavy, které jsou akceptovány pouze v případě splnění určitých dalších podmínek

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Bity v podpoli nemusí být nutně sousedící. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Pro identifikaci podpolí jsou definovány následující masky:

MQRO_REJECT_UNSUP_MASK
Tato maska určuje bitové pozice v poli *Report* , kde volby sestavy, které nejsou podporovány lokálním správcem front, způsobí selhání volání MQPUT nebo MQPUT1 s kódem dokončení operace 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 určuje bitové pozice v poli *Report* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, nicméně přijímány na základě volání MQPUT nebo MQPUT1 . V tomto případě se vrací 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.

Do tohoto podpole jsou zahrnuty následující volby sestavy:

- AKTIVITA MQRO_ACTIVITY
- MQRO_COPY_MSG_ID_TO_CORREL_ID
- MQRO_DEAD_LETTER_Q
- MQRO_DISCARD_MSG
- VÝJIMKA MQRO_EXCEPTION
- MQRO_EXCEPTION_WIT_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
- ID_KOLEKCE_MQRO_PASS_RELACE_
- MQRO_PASS_MSG_ID

MQRO_ACCEPT_UNSUP_IF_XMIT_MASK
Tato maska určuje bitové pozice v poli *Report* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, nicméně přijímány na základě 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 (tedy fronta určená poli *ObjectQMgrName* a *ObjectName* v deskriptoru objektu uvedeném v volání MQOPEN nebo MQPUT1 není lokální přenosová fronta).

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.

Do tohoto podpole jsou zahrnuty následující volby sestavy:

- MQRO_COA
- MQRO_COA_WITH_DATA
- MQRO_COA_WITH_FULL_DATA
- MQRO_COD
- MQRO_CED_WITH_DATA
- MQRO_COD_WITH_FULL_DATA

Pokud jsou v poli *Report* zadány nějaké volby, které správce front nerozpozná, zkontroluje správce front každé dílčí pole postupně pomocí bitové operace AND, aby zkombinoval pole *Report* s maskou pro toto dílčí pole. Není-li výsledek této operace nula, vrátí se kód dokončení a kódy příčiny popsané dříve.

Je-li vrácen parametr MQCC_WARNING, není definován kód příčiny, který má být vrácen, pokud existují další varovné podmínky.

Možnost zadat a přijmout volby sestavy, které nejsou rozpoznány lokálním správcem front, je užitečné při odesílání zprávy s volbou sestavy, která je rozpoznána a zpracována *vzdáleným* správcem front.

Analýza pole sestavy

Pole *Report* obsahuje podpole; z toho důvodu aplikace, které potřebují zkontrolovat, zda odesilatel zprávy vyžádal určitou sestavu, musí použít některou z popsaných technik.

Použití bitových operací

Pokud programovací jazyk podporuje bitové operace, proveďte následující kroky:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která má být zkontrolována:
   - Sestava MQRO_COA_WITH_FULL_DATA pro sestavu COA
   - Sestava MQRO_COD_WITH_FULL_DATA pro sestavu COD
   - Sestava MQRO_EXCEPTION_WITH_FULL_DATA pro sestavu výjimek
   - Sestava MQRO_EXPIRATION_WITH_FULL_DATA pro vypršení platnosti sestavy

   Volejte hodnotu A.

   V systému z/OSpoužijte hodnoty MQRO_ * _WITH_DATA-namísto hodnot MQRO_ * _WITH_FULL_DATA.
2. Zkombinujte pole *Report* s A pomocí bitové AND operace; volejte výsledek B.
3. Testujte B for equality s každou hodnotou, která je možná pro daný typ sestavy.

   Je-li například A MQRO_EXCEPTION_WITH_FULL_DATA, test B pro rovnost s každou z následujících možností určuje, co bylo zadáno odesilatelem zprávy:

   - MQRO_NONE
   - VÝJIMKA MQRO_EXCEPTION
   - MQRO_EXCEPTION_WIT_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   Testy lze provádět v libovolném pořadí, které je nejvhodnější pro logiku aplikace.

Pro test voleb MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID použijte podobnou metodu; vyberte jako hodnotu A , podle toho, které z těchto dvou konstant je vhodné, a poté pokračujte podle postupu popsaného výše.

Použití aritmetiky

Pokud jazyk programovacího jazyka *nepodporuje* bitové operace, proveďte následující kroky pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která má být zkontrolována:
   - Sestava MQRO_COA pro sestavu COA
   - Sestava MQRO_COD pro sestavu COD
   - Sestava MQRO_EXCEPTION pro sestavu výjimek
   - Sestava MQRO_EXPIRATION pro sestavu vypršení platnosti

   Volejte hodnotu A.
2. Rozdělte pole *Report* hodnotou A ; volejte výsledek B.
3. Rozdělit B podle 8 ; volejte výsledek C.
4. Násobení C od 8 a odečtením od B ; volejte výsledek D.
5. Násobení D podle A ; volejte výsledek E.
6. Testujte E for equality s každou hodnotou, která je možná pro daný typ sestavy.

   Je-li například A MQRO_EXCEPTION, otestujte E pro rovnost s každou z následujících možností, abyste určili, co bylo zadáno odesilatelem zprávy:
   - MQRO_NONE
   - VÝJIMKA MQRO_EXCEPTION
   - MQRO_EXCEPTION_WIT_DATA
   - MQRO_EXCEPTION_WITH_FULL_DATA

   Testy lze provádět v libovolném pořadí, které je nejvhodnější pro logiku aplikace.

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
```

Použijte podobnou metodu testování pro volby MQRO_PASS_MSG_ID nebo MQRO_PASS_CORREL_ID; vyberte ji jako hodnotu A , podle toho, které z těchto dvou konstant je vhodné, a poté pokračujte podle postupu popsaného výše, ale hodnotu 8 nahraďte hodnotou 2.

Struktura pole s příznaky zprávy

Tyto informace popisují strukturu pole příznaků zpráv.

Pole *MsgFlags* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí. Tato podpole identifikují:

- Příznaky zpráv, které jsou zamítnuty v případě, že je lokální správce front nerozpozná
- Příznaky zpráv, které jsou vždy akceptovány, i když je lokální správce front nerozpozná
- Příznaky zpráv, které jsou akceptovány pouze v případě splnění určitých dalších podmínek

Poznámka: Všechna podpole v produktu *MsgFlags* jsou vyhrazena pro použití správcem front.

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Pro identifikaci podpolí jsou definovány následující masky:

MQMF_REJECT_UNSUP_MASK,
Tato maska určuje 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.

Do tohoto podpole jsou zahrnuty následující příznaky zpráv:

- MQM_LAST_MSG_IN_GROUP
- MQMF_LAST_SEGMENT
- MQMF_MSG_IN_GROUP
- SEGMENT MQMF_SEGMENT
- MQMF_SEGMENTATION_ALLOWED
- MQMF_SEGMENTATION_BLOKOVÁNO

MQMF_ACCEPT_UNSUP_MASK,
Tato maska určuje bitové pozice v poli *MsgFlags* , kde jsou nicméně přijímány příznaky zpráv, které nejsou podporovány lokálním správcem front, na volání MQPUT nebo MQPUT1 . Kód dokončení je MQCC_OK.

Toto podpole zabírá bitové pozice 0 až 11.

FUNKCE MQMF_ACCEPT_UNSUP_IF_XMIT_MASK
Tato maska identifikuje bitové pozice v poli *MsgFlags* , kde jsou příznaky zpráv, které nejsou podporovány lokálním správcem front, nicméně přijímány na základě 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 (tedy fronta určená poli *ObjectQMgrName* a *ObjectName* v deskriptoru objektu uvedeném v volání MQOPEN nebo MQPUT1 není lokální přenosová fronta).

Kód dokončení MQCC_OK je vrácen, pokud jsou tyto podmínky splněny, a MQCC_FAILED s kódem příčiny MQRC_MSG_FLAGS_ERROR, pokud ne.

Toto podpole zabírá bitové pozice 12 až 19.

Pokud jsou v poli *MsgFlags* zadány parametry, které správce front nerozpoznal, zkontroluje správce front každé dílčí pole postupně pomocí bitové operace AND, aby zkombinoval pole *MsgFlags* s maskou pro toto podpole. Není-li výsledek této operace nula, vrátí se kód dokončení a kódy příčiny popsané dříve.

Uživatelská procedura konverze dat

Tato kolekce témat popisuje rozhraní pro uživatelskou proceduru pro převod dat a zpracování prováděné správcem front při požadavku na převod dat.

Další informace o převodu dat naleznete v tématu *Převod dat v produktu IBM MQ* na adrese https://www.ibm.com/support/pages/node/317869.

Uživatelská procedura pro převod 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 na volání MQGET.

Jsou popsány následující subjekty:

- Zpracování prováděné správcem front v odezvě na volbu MQGMO_CONVERT; viz “Zpracování konverze” na stránce 898.

- Konvence zpracování použité správcem front při zpracování vestavěného formátu; tyto konvence se doporučují také pro uživatelské procedury zápisu. Viz "Konvence zpracování" na stránce 899.
- Speciální pokyny pro převod zpráv sestav viz "Převod zpráv sestav" na stránce 903.
- Parametry předané uživatelské proceduře pro převod dat; viz "MQ_DATA_CONV_EXIT-Ukončení převodu dat" na stránce 916.
- Volání, které lze použít z uživatelské procedury pro převod znakových dat mezi různými reprezentacemi; viz "MQXCNVC-Převod znaků" na stránce 909.
- Parametr datové struktury, který je specifický pro uživatelskou proceduru, viz "MQDXP-Data-konverze-výstupní parametr" na stránce 904.

Zpracování konverze

Tyto informace popisují zpracování prováděné správcem front v odezvě na volbu MQGMO_CONVERT.

Správce front provede následující akce, je-li v rámci volání MQGET zadána volba MQGMO_CONVERT a je vrácena zpráva, která má být vrácena aplikaci:

1. Je-li splněna jedna nebo více z následujících podmínek, není převod nutný:
   - Data zprávy jsou již ve znakové sadě a kódování požadované aplikací, která vydala volání MQGET. Aplikace musí nastavit pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** v rámci volání MQGET na vyžadované hodnoty před zadáním volání.
   - Délka dat zprávy je nula.
   - Délka parametru **Buffer** volání MQGET je nulová.

   V těchto případech je zpráva vrácena bez převodu na aplikaci, která vydala volání MQGET; hodnoty *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastaveny na hodnoty v řídicích informacích ve zprávě a volání je dokončeno s jednou z následujících kombinací kódu dokončení a kódu příčiny:

Tabulka 633. Kód dokončení a kombinace kódu příčiny

Kód dokončení	Kód příčiny
MQCC_OK	MQRC_NONE
VAROVÁNÍ MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED
VAROVÁNÍ MQCC_WARNING	OPERACE MQRC_TRUNCATED_MSG_FAILED

   Následující kroky se provádějí pouze v případě, že znaková sada nebo kódování dat zprávy se liší od odpovídající hodnoty v parametru **MsgDesc** a že jsou data k převedení:
2. Pokud má pole *Format* v informacích o ovládacím prvku ve zprávě hodnotu MQFMT_NONE, bude vrácena nekonvertovaný kód dokončení s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

   Ve všech ostatních případech zpracování konverze pokračuje.
3. Zpráva se odstraní z fronty a umístí se do dočasné vyrovnávací paměti, která má stejnou velikost jako parametr **Buffer** . Pro operace procházení je zpráva kopírována do dočasné vyrovnávací paměti místo toho, aby byla odebrána z fronty.
4. Pokud má být zpráva oseknuta tak, aby se vešla do vyrovnávací paměti, provede se následující:
   - Pokud nebyla zadána volba MQGMO_ACCEPT_TRUNCATED_REMSG, bude vrácena nekonvertovaný kód dokončení MQCC_WARNING a kód příčiny MQRC_TRUNCATED_MSG_FAILED.
   - Je-li uvedena volba MQGMO_ACCEPT_TRUNCATED_MSG *bylo* , kód dokončení je nastaven na hodnotu MQCC_WARNING, kód příčiny je nastaven na hodnotu MQRC_TRUNCATED_MSG_ACCEPTED a zpracování konverze pokračuje.
5. Pokud lze zprávu umístit do vyrovnávací paměti bez zkrácení nebo byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG, bude provedeno následující:

- Je-li formát vestavěným formátem, vyrovnávací paměť se předá do služby pro převod dat správce front.
- Pokud formát není vestavěný formát, bude vyrovnávací paměť předána uživatelské proceduře s týmž názvem jako má formát. Nelze-li uživatelskou proceduru najít, vrátí se nekonvertovaný kód s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_FORMAT_ERROR.

Pokud se nevyskytne žádná chyba, výstup ze služby pro převod dat nebo z uživatelem napsaného ukončení je převedená zpráva a kód dokončení a kód příčiny, které se vrátí do aplikace, která volala příkaz MQGET.

6. Je-li konverze úspěšná, správce front vrátí převedenou zprávu na aplikaci. V tomto případě je kód dokončení a kód příčiny vrácený voláním MQGET jednou z následujících kombinací:

Tabulka 634. Kód dokončení a kombinace kódu příčiny

Kód dokončení	Kód příčiny
MQCC_OK	MQRC_NONE
VAROVÁNÍ MQCC_WARNING	MQRC_TRUNCATED_MSG_ACCEPTED

Je-li však konverze provedena uživatelskou procedurou, mohou být vráceny jiné kódy příčiny, i když je konverze úspěšná.

Pokud převod selže, správce front vrátí nekonvertovaný zprávu do aplikace s poli *CodedCharSetId* a *Encoding* v parametru **MsgDesc** nastaveným na hodnoty v řídicích informacích ve zprávě a s kódem dokončení MQCC_WARNING.

Konvence zpracování

Při převádění vestavěného formátu postupuje správce front podle popisovaných konvencích zpracování.

Uživatelské procedury by měly být také následovány těmito konvencemi, ačkoli správce front toto oprávnění nevynucuje. Vestavěné formáty převedené správcem front jsou:

- MQFMT_ADMIN
- MQFMT_CICS (pouze z/OS)
- MQFMT_COMMAND_1
- MQFMT_COMMAND_2
- HLAVIČKA MQFMT_DEAD_LETTER_HEADER
- ZÁHLAVÍ MQFMT_DICT_HEADER
- MQFMT_EVENT verze 1
- MQFMT_EVENT, verze 2
- MQFMT_IMS
- MQFMT_IMS_VAR_STRING
- ROZŠÍŘENÍ MQFMT_MD_EXTENSION
- MQFMT_PCF
- MQFMT_REF_MSG_HEADER
- ZÁHLAVÍ MQFMT_RF_HEADER
- MQFMT_RF_HEADER_2
- ŘETĚZEC MQFMT_STRING
- SPOUŠTĚČ MQFMT_TRIGGER
- MQFMT_WORK_INFO_HEADER (pouze z/OS)
- ZÁHLAVÍ MQFMT_XMIT_Q_HEADER

1. Pokud se zpráva během převodu rozbalí a překročí velikost parametru **Buffer** , provede se následující akce:
   - Pokud nebyla zadána volba MQGMO_ACCEPT_TRUNCATED_REMSG, bude vrácena nekonvertovaný kód dokončení MQCC_WARNING a kód příčiny MQRC_CONVERTED_MSG_TOO_BIG.
   - Pokud byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG *byla* , zpráva je zkrácena, kód dokončení je nastaven na hodnotu MQCC_WARNING, kód příčiny je nastaven na hodnotu MQRC_TRUNCATED_MSG_ACCEPTED a zpracování konverze pokračuje.
2. Dojde-li k oříznutí (buď před převodem nebo během převodu), může být počet platných bajtů vrácených v parametru **Buffer** menší než délka vyrovnávací paměti.

   K tomu může dojít například v případě, že se jedná o 4bajtové celé číslo nebo o znak DBCS, který je strdles na konec vyrovnávací paměti. Neúplný prvek informací není převeden a ty bajty ve vrácené zprávě neobsahují platné informace. K tomu může dojít také v případě, že byla během převodu oříznuta zpráva, která byla oříznuta před konverzí převodu.

   Pokud je počet vrácených platných bajtů menší než délka vyrovnávací paměti, nepoužité bajty na konci vyrovnávací paměti jsou nastaveny na hodnotu null.
3. Pokud pole nebo řetězec obsahuje konec vyrovnávací paměti, je konvertováno tolik dat, kolik je možné; pouze daný prvek pole nebo znak DBCS, který je neúplný, se nekonvertuje; jsou převedeny předchozí prvky nebo znaky pole.
4. Dojde-li k oříznutí (buď před nebo během převodu), délka vrácená pro parametr **DataLength** je délka nepřevedené zprávy před oseknutím.
5. Pokud se řetězce převádějí mezi jednobajtovými znakovými sadami (SBCS), dvoubajtovými znakovými sadami (DBCS) nebo vícebajtovými znakovými sadami (MBCS), řetězce se mohou rozšiřovat nebo uzavírat smlouvy.
   - V PCF formátuje MQFMT_ADMIN, MQFMT_EVENT a MQFMT_PCF řetězce v strukturách MQCFST a MQCFSL rozbalením nebo kontraktem podle potřeby pro umístění řetězce po převodu.

     Pro strukturu seznamu řetězců MQCFSL se mohou řetězce v seznamu rozšiřovat nebo uzavírat podle různých částek. Pokud k tomu dojde, správce front vycpává kratší řetězce mezerami tak, aby jejich délka byla stejná jako nejdelší řetězec po převodu.
   - Ve formátu MQFMT_REF_MSG_HEADER jsou řetězce adresované poli `SrcEnvOffset`, `SrcNameOffset`, `DestEnvOffseta` `DestNameOffset` rozšiřovány nebo podle potřeby podle potřeby akceptovány pro umístění řetězců po převodu.
   - Pole `NameValueString` ve formátu MQFMT_RF_HEADER rozšiřuje pole nebo smlouvy podle potřeby tak, aby bylo možné přizpůsobit dvojice názvu a hodnoty po převodu.
   - Ve strukturách s pevnými velikostmi polí umožňuje správce front rozšiřovat nebo uzavírat smlouvy v rámci svých pevných polí za předpokladu, že žádné významné informace nejsou ztraceny. V tomto ohledu jsou koncové mezery a znaky následující za prvním znakem null v poli považovány za nevýznamné.
     - Pokud se řetězec rozvine, ale pouze nevýznamné znaky je třeba vyřadit, aby bylo možné do pole umístit převedený řetězec, konverze uspěje a volání je dokončeno s operací MQCC_OK a s kódem příčiny MQRC_NONE (za předpokladu, že nejsou žádné jiné chyby).
     - Pokud se řetězec rozvine, ale převedený řetězec vyžaduje, aby byly do pole vhozeny velké znaky, aby se do pole vešly, byla vrácena nekonvertovaný zpráva a volání je dokončeno s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_STRING_TOO_BIG.

       Poznámka: Kód příčiny MQRC_CONVERTED_STRING_TOO_BIG má v tomto případě za následek určení, zda byla zadána volba MQGMO_ACCEPT_TRUNCATED_MSG.
     - Pokud jsou řetězce smlouvy, správce front vycpávky z řetězce s mezerami do délky pole.
6. Pro zprávy sestávající z jedné nebo více struktur záhlaví MQ , za nimiž následují uživatelská data, může být převedena jedna nebo více struktur záhlaví, zatímco zbytek zprávy nikoli. Nicméně (se dvěma výjimkami) vždy pole *CodedCharSetId* a *Encoding* v každé struktuře záhlaví vždy správně označují znakovou sadu a kódování dat, která se řídí strukturou záhlaví.

Tyto 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 argumentu **MsgDesc** určují hodnoty, které nejsou definovány nebo nejsou podporovány, může správce front chybu ignorovat, pokud nedefinovaná nebo nepodporovaná hodnota nemusí být použita při převodu zprávy.

   Pokud například pole *Encoding* ve zprávě určuje nepodporované kódování s plovoucí desetinnou čárkou, ale zpráva obsahuje pouze celočíselné údaje nebo obsahuje data s pohyblivou řádovou čárkou, která nevyžadují převod (protože zdrojový a cílový typ float jsou identické), nelze chybu diagnostikovat.

   Je-li diagnostikována chyba, je vrácena nekonvertovaný zpráva s kódem dokončení MQCC_WARNING a jedním z kódů příčiny MQRC_SOURCE_ * _ERROR nebo MQRC_TARGET_ * _ERROR (je-li to vhodné); pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastaveny na hodnoty v řídicích informacích ve zprávě.

   Není-li chyba diagnostikována a konverze se úspěšně dokončí, hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v argumentu **MsgDesc** jsou hodnoty zadané aplikací zadávající volání MQGET.

8. Ve všech případech je zpráva vrácena do aplikace bez převedení kódu dokončení je nastavena na hodnotu MQCC_WARNING a pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** jsou nastavena na hodnoty odpovídající nekonverzeným datům. To se provádí také pro MQFMT_NONE.

   Argument **Reason** je nastaven na kód, který udává, proč se konverze nepodařilo provést, pokud zpráva také nebyla oříznuta; kódy příčiny související s oseknutím mají přednost před kódy příčiny souvisejícími s převodem. (Chcete-li určit, zda byla zkrácená zpráva převedena, zkontrolujte hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v parametru **MsgDesc** .)

   Je-li diagnostikována chyba, je vrácen specifický kód příčiny nebo obecný kód příčiny MQRC_NOT_CONVERTED. Vrácený kód příčiny závisí na schopnostech diagnostiky základní služby pro převod dat.

9. Je-li vrácen kód dokončení MQCC_WARNING a je relevantní více než jeden kód příčiny, bude mít pořadí přednosti následující pořadí:

   a. Následující důvody mají přednost před všemi ostatními; pouze jeden z důvodů v této skupině může vzniknout:

      - MQRC_SIGNAL_REQUEST_ACCEPTED
      - MQRC_TRUNCATED_MSG_ACCEPTED

   b. Pořadí priorit v rámci zbývajících kódů příčiny není definováno.

10. Po dokončení volání MQGET:

    - Následující kód příčiny indikuje, že zpráva byla úspěšně převedena:
      - MQRC_NONE
    - Následující kódy příčiny označují, že zpráva *mohla* byla úspěšně převedena (zkontrolujte pole *CodedCharSetId* a *Encoding* v parametru **MsgDesc** , abyste zjistili aktuální informace):
      - MQRC_MSG_MARKED_BROWSE_CO_OP
      - MQRC_TRUNCATED_MSG_ACCEPTED
    - Všechny ostatní kódy příčiny indikují, že zpráva nebyla převedena.

    Následující zpracování je specifické pro vestavěné formáty. Nevztahuje se na uživatelem definované formáty:

11. S výjimkou následujících formátů:

    - MQFMT_ADMIN
    - MQFMT_COMMAND_1

- MQFMT_COMMAND_2
- UDÁLOST MQFMT_EVENT
- MQFMT_IMS_VAR_STRING
- MQFMT_PCF
- ŘETĚZEC MQFMT_STRING

žádný z vestavěných formátů nelze převést ze znakových sad nebo do znakových sad, které nemají znaky SBCS pro znaky, které jsou platné ve názvech front. Je-li proveden pokus o provedení takové konverze, je vrácena nekonvertovaný zpráva s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_SOURCE_CCSID_ERROR nebo MQRC_TARGET_CCSID_ERROR.

Znaková sada Unicode UTF-16 je příklad znakové sady, která nemá znaky SBCS pro znaky, které jsou platné ve jménech front.

12. Jsou-li data zprávy pro vestavěný formát zkrácena, pole ve zprávě obsahující délky řetězců nebo počty prvků nebo struktur nejsou upravena tak, aby odrážela délku dat, která se skutečně vrací do aplikace; hodnoty vrácené pro taková pole v datech zprávy jsou hodnoty použitelné pro zprávu *před oříznutím*.

    Při zpracování zpráv, jako je zkrácená zpráva MQFMT_ADMIN, se ujistěte, že se aplikace nepokusí o přístup k datům za koncem vrácených dat.

13. Je-li název formátu MQFMT_DEAD_LETTER_HEADER, data zprávy začínají strukturou MQDLH, pravděpodobně za ním následují nula nebo více bajtů dat zprávy aplikace. Formát, znaková sada a kódování dat zprávy aplikace jsou definovány v polích `Format`, `CodedCharSetId`a `Encoding` ve struktuře MQDLH na začátku zprávy. Protože struktura MQDLH a data zprávy aplikace mohou mít různé znakové sady a kódování, jedna, druhá nebo obě hodnoty struktury MQDLH a data zprávy aplikace mohou vyžadovat konverzi.

    Správce front převede nejprve strukturu MQDLH podle potřeby. Pokud je převod úspěšný, nebo struktura MQDLH nevyžaduje převod, správce front zkontroluje pole `CodedCharSetId` a `Encoding` ve struktuře MQDLH, aby zjistil, zda je vyžadována konverze dat zprávy aplikace. Je-li požadována konverze, vyvolá správce front uživatelskou proceduru s názvem zadaným polem `Format` ve struktuře MQDLH nebo provede vlastní převod (pokud `Format` je název vestavěného formátu).

    Pokud volání MQGET vrátí kód dokončení MQCC_WARNING a kód příčiny je jeden z těch, které označují, že konverze nebyla úspěšná, použije se jedna z následujících možností:

    - Strukturu MQDLH nelze převést. V tomto případě data zprávy aplikace nebudou konvertována.
    - 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 bylo možné určit, která z výše uvedených hodnot se má použít.

14. Je-li název formátu MQFMT_XMIT_Q_HEADER, data zprávy začínají strukturou MQXQH, případně mohou následovat nula nebo více bajtů dalších dat. Tato přídavná 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 MQ .

    Struktura MQXQH musí být ve znakové sadě a kódování správce front. Formát, znaková sada a kódování dat následující strukturu MQXQH jsou dány poli `Format`, `CodedCharSetId`a `Encoding` ve struktuře MQMD obsažené v MQXQH. Pro každou následující strukturu záhlaví MQ jsou pole `Format`, `CodedCharSetId`a `Encoding` ve struktuře popisovat data, která následují za touto strukturou; tato data jsou buď jiná struktura záhlaví MQ , nebo data zprávy aplikace.

    Je-li pro zprávu MQFMT_XMIT_Q_HEADER zadána volba MQGMO_CONVERT, budou převedena data zprávy aplikace a některé struktury záhlaví produktu MQ , *ale data ve struktuře MQXQH nejsou*. Při návratu z volání MQGET proto postupujte takto:

    - Hodnoty polí `Format`, `CodedCharSetId`a `Encoding` v parametru **MsgDesc** popisují data ve struktuře MQXQH a nikoli data zprávy aplikace; tyto hodnoty tedy nejsou stejné jako hodnoty určené aplikací, která vydala volání MQGET.

Výsledkem je, že aplikace, která opakovaně získává zprávy z přenosové fronty s určenou volbou MQGMO_CONVERT, musí před každým voláním MQGET resetovat pole `CodedCharSetId` a `Encoding` v parametru **MsgDesc** na hodnoty vyžadované pro data zprávy aplikace.

- Hodnoty polí `Format`, `CodedCharSetId`a `Encoding` v poslední struktuře záhlaví MQ popisují data zprávy aplikace. Pokud zde nejsou žádné další struktury záhlaví MQ , data zprávy aplikací jsou popsána těmito poli ve struktuře MQMD v rámci struktury MQXQH. Je-li konverze úspěšná, budou hodnoty 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 jeho pole záznamů MQOR a MQPMR), které mohou být následně následovány nulou nebo více dalšími strukturami záhlaví MQ a s nulovým nebo více bajty dat zprávy aplikace. Podobně jako struktura MQXQH se struktura MQDH musí nacházet ve znakové sadě a kódování správce front a nebude převedena na volání MQGET, a to ani v případě, že je zadána volba MQGMO_CONVERT.

Zpracování dříve popisovaných struktur MQXQH a MQDH je primárně určeno pro použití v agentech zpráv při získávání zpráv z přenosových front.

Převod zpráv sestav

Obecně může zpráva hlášení obsahovat různé množství dat aplikační zprávy, v závislosti na volbách sestavy uvedených odesílatelem původní zprávy. Avšak, sestava aktivit může obsahovat data, ale bez volby sestavy, která uvádí * _WITH_DATA v konstantě.

Zpráva sestavy může obsahovat zejména:

1. Žádná data zprávy aplikace
2. Některá data zprávy aplikace z původní zprávy

   K tomu dojde, 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 dojde, když odesilatel 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 s hlášením, zkopíruje název formátu z původní zprávy do pole *`Format`* v řídicí informaci ve zprávě sestavy. Název formátu ve zprávě sestavy může proto znamenat délku dat, která se liší od délky skutečně obsažené ve zprávě sestavy (případy 1 a 2 dříve).

Je-li při načítání zprávy sestavy uvedena volba MQGMO_CONVERT, postupujte takto:

- V případě předchozího případu se uživatelská procedura konverze dat nevyvolá (protože zpráva hlášení neobsahuje žádná data).
- Pro případ 3 dříve název formátu správně implikuje délku dat zprávy.
- Avšak pro případ 2 dříve byla uživatelská procedura převodu dat vyvolána pro převod zprávy, která je *kratší* než délka, která je odvozena z názvu formátu.

  Kromě toho je kód příčiny předaný uživatelské proceduře obvykle MQRC_NONE (to znamená, že kód příčiny neoznačuje, že zpráva byla zkrácena). Důvodem je skutečnost, že data zprávy byla zkrácena *odesilatelem* zprávy sestavy a nikoli správcem front příjemce v odezvě na volání MQGET.

Vzhledem k těmto možnostem nesmí uživatelská procedura pro převod dat použít název formátu k odvození délky dat, která mu byla předána; místo toho musí uživatelská procedura zkontrolovat délku poskytnutých dat a musí být připravena převést menší množství dat, než je délka, která je odvozena z názvu formátu. Pokud lze data úspěšně převést, kód dokončení MQCC_OK a kód příčiny MQRC_NONE musí být vráceni ukončením. Délka dat zprávy, která má být konvertována, je předána ukončení jako parametr **InBufferLength** .

Rozhraní pro programování závislé na produktu

MQDXP-Data-konverze-výstupní parametr

Struktura MQDXP je parametr, který správce front předá výstupnímu programu pro převod dat při vyvolání uživatelské procedury pro převod dat zprávy v rámci zpracování volání MQGET. Podrobné informace o ukončení převodu dat naleznete v popisu volání MQ_DATA_CONV_EXIT.

Znaková data v aplikaci MQDXP jsou ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front produktu **CodedCharSetId** . Numerická data v MQDXP jsou v nativním kódování počítače; to je dáno MQENC_NATIVE.

Ukončovací program může změnit pouze pole *DataLength*, *CompCode*, *Reason*a *ExitResponse* v MQDXP; změny v ostatních polích budou ignorovány. Pole *DataLength* však nelze změnit, jestliže převáděná zpráva je segment, který obsahuje pouze část logické zprávy.

Když se řízení vrátí ke správci front z uživatelské procedury, správce front zkontroluje hodnoty vrácené MQDXP. Pokud vrácené hodnoty nejsou platné, správce front bude pokračovat ve zpracování stejně, jako by funkce MQXDR_CONVERSION_FAILED v produktu *ExitResponse* vrátila chybu MQXDR_CONVERSION_FAILED. Správce front však ignoruje hodnoty polí *CompCode* a *Reason* vrácených uživatelskou procedurou v tomto případě a použije místo toho hodnoty, které měla tato pole na hodnotě *vstup* pro ukončení. Následující hodnoty v MQDXP způsobí, že se toto zpracování bude provádět:

- Pole *ExitResponse* není MQXDR_OK a ne MQXDR_CONVERSION_FAILED
- Pole *CompCode* není MQCC_OK a ne MQCC_WARNING
- Pole *DataLength* menší než nula nebo *DataLength* se změnilo, když převáděná zpráva je segment, který obsahuje pouze část logické zprávy.

Následující tabulka shrnuje pole ve struktuře.

Tabulka 635. Pole v MQDXP

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
AppOptions	Volby aplikace	AppOptions
Encoding	Aplikace-numerické kódování požadované aplikací	Kódování
CodedCharSetId	Znaková sada vyžadovaná aplikací	CodedCharSetId
DataLength	Délka dat zprávy v bajtech	DataLength
CompCode	Kód dokončení	CompCode
Reason	Kód příčiny kvalifikující *CompCode*	Příčina
ExitResponse	Odezva z uživatelské procedury	ExitResponse
Hconn	Manipulátor připojení	Hconn
pEntryPoints	Adresa struktury MQIEP	pEntrybodů

Pole

Struktura MQDXP obsahuje následující pole; pole jsou popsána v abecedním pořadí.

AppOptions
Typ: MQLONG

Jedná se o kopii pole *Options* struktury MQGMO určeného aplikací, která vydala volání MQGET. Ukončení může být nutné prozkoumat, aby bylo možné zjistit, 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 vyžadované aplikací, která vydala volání MQGET; viz pole *CodedCharSetId* ve struktuře MQMD pro více podrobností. Pokud aplikace určuje speciální hodnotu MQCCSI_Q_MGR při volání MQGET, změní ji správce front na skutečný identifikátor znakové sady znakové sady používané správcem front před vyvoláním uživatelské procedury.

Je-li konverze úspěšná, uživatelská procedura musí kopírovat toto pole do pole *CodedCharSetId* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

CompCode
Typ: MQLONG

Když je vyvoláno ukončení, obsahuje kód dokončení, který je vrácen do aplikace, která vydala volání MQGET, pokud tento výstup nic neudělá. Vždy je to MQCC_WARNING, protože buď byla zpráva zkrácena, nebo zpráva vyžaduje konverzi, a to ještě nebylo provedeno.

Na výstupu z uživatelské procedury toto pole obsahuje kód dokončení, který má být vrácen do aplikace v parametru **CompCode** volání MQGET; jsou platné pouze hodnoty MQCC_OK a MQCC_WARNING. Prohlédněte si popis pole *Reason* pro návrhy na to, jak může procedura nastavit toto pole na výstupu.

Jedná se o vstupní/výstupní pole pro ukončení.

DataLength
Typ: MQLONG

Když je vyvoláno ukončení, toto pole obsahuje původní délku dat zprávy aplikace. Pokud byla zpráva zkrácena, aby se vešla do vyrovnávací paměti poskytnuté aplikací, velikost zprávy zadané pro uživatelskou proceduru je *menší* než hodnota parametru *DataLength*. Velikost zprávy poskytované při ukončení je vždy dána parametrem **InBufferLength** ukončení, bez ohledu na jakékoli oseknutí, které se vyskytlo.

Oseknutí je indikováno polem *Reason* s hodnotou MQRC_TRUNCATED_MSG_ACCEPTED na vstupu do uživatelské procedury.

Většina přepočtů nemusí tuto délku měnit, ale procedura může v případě potřeby provést; hodnota nastavená uživatelskou procedurou se vrátí do aplikace v parametru **DataLength** volání MQGET. Tato délka však nemůže být změněna, pokud převáděná zpráva je segment, který obsahuje pouze část logické zprávy. Důvodem je to, že změna délky by způsobila, že odchylky dalších segmentů v logické zprávě budou nesprávné.

Všimněte si, že pokud chce uživatelská procedura změnit délku dat, uvědomte si, že správce front již rozhodl, zda data zprávy zapadá do vyrovnávací paměti aplikace, a to na základě délky *nepřevedených* dat. Toto rozhodnutí určuje, zda je zpráva odebrána z fronty (nebo se přemístil kurzor procházení pro požadavek na procházení) a není ovlivněn žádnou změnou délky dat způsobené převodem. Z tohoto důvodu se doporučuje, aby převodní procedura nezpůsobila změnu v délce dat zprávy aplikace.

Pokud převod znaků implikuje změnu délky, lze řetězec převést na jiný řetězec se stejnou délkou v bajtech, zkracovat koncové mezery nebo vyplňovat mezerami podle potřeby.

Uživatelská procedura se nevyvolá, pokud zpráva neobsahuje žádná data zprávy aplikace; proto je *DataLength* vždy větší než nula.

Jedná se o vstupní/výstupní pole pro ukončení.

Encoding
Typ: MQLONG

Numerické kódování požadované aplikací.

Jedná se o číselné kódování požadované aplikací, která vydala volání MQGET. Další podrobnosti naleznete v poli *Encoding* ve struktuře MQMD.

Je-li konverze úspěšná, uživatelská procedura zkopíruje toto pole do pole *Encoding* v deskriptoru zpráv.

Toto je vstupní pole pro ukončení.

ExitOptions
Typ: MQLONG

Jedná se o vyhrazené pole; jeho hodnota je 0.

ExitResponse
Typ: MQLONG

Odezva z ukončení.Toto nastavení je nastaveno na základě ukončení, aby se označilo úspěch nebo jinak konverze. Musí se jednat o jeden z následujících:

MQXDR_OK
Převod byl úspěšný.

Pokud tato hodnota určuje tuto hodnotu, vrátí správce front následující informace o aplikaci, která vydala volání MQGET:

- Hodnota pole *CompCode* na výstupu z uživatelské procedury
- Hodnota pole *Reason* na výstupu z uživatelské procedury
- Hodnota pole *DataLength* na výstupu z uživatelské procedury
- Obsah výstupní vyrovnávací paměti výstupu *OutBuffer*. Počet vrácených bajtů je menší z hodnot parametru **OutBufferLength** exit a hodnota pole *DataLength* na výstupu z ukončení.

  Pokud jsou pole *Encoding* a *CodedCharSetId* v parametru deskriptoru zprávy uživatelské procedury *both* nezměněná, vrátí správce front následující zprávy:
- Hodnota polí *Encoding* a *CodedCharSetId* ve struktuře MQDXP na *vstupu* k ukončení.

  Pokud byla změněna jedna nebo obě pole *Encoding* a *CodedCharSetId* v parametru deskriptoru zpráv uživatelské procedury, vrátí správce front následující zprávy:
- Hodnota polí *Encoding* a *CodedCharSetId* v parametru deskriptoru zprávy uživatelské procedury na výstupu z uživatelské procedury.

MQXDR_CONVERSION_FAILED
Převod byl neúspěšný.

Pokud tato hodnota určuje tuto hodnotu, vrátí správce front následující informace o aplikaci, která vydala volání MQGET:

- Hodnota pole *CompCode* na výstupu z uživatelské procedury
- Hodnota pole *Reason* na výstupu z uživatelské procedury
- Hodnota pole *DataLength* na *vstupu* pro ukončení
- Obsah vstupní vyrovnávací paměti uživatelské procedury *InBuffer*. Počet vrácených bajtů je zadán parametrem **InBufferLength** .

Pokud byla ukončena uživatelská procedura *InBuffer*, výsledky nejsou definovány.

ExitResponse je výstupní pole z uživatelské procedury.

Hconn
Typ: MQHCONN

Jedná se o manipulátor připojení, který lze použít při volání MQXCNVC. Tento manipulátor nemusí být nutně stejný jako popisovač určený aplikací, která vydala volání MQGET.

pEntryPoints
Typ: PMQIEP

Adresa struktury MQIEP, pomocí které lze provádět volání MQI a DCI.

Reason
Typ: MQLONG

Kód příčiny kvalifikující *CompCode*.

Když je vyvolána uživatelská procedura, obsahuje kód příčiny vrácený do aplikace, která vydala volání MQGET, pokud se uživatelská procedura rozhodne nedělat nic. Mezi možné hodnoty patří MQRC_TRUNCATED_MSG_ACCEPTED, což znamená, že zpráva byla zkrácena, aby se vešla do vyrovnávací paměti poskytnuté aplikací, a MQRC_NOT_CONVERTED, což znamená, že zpráva vyžaduje převod, ale že tato zpráva nebyla ještě hotova.

Na výstupu z uživatelské procedury je toto pole obsahovat důvod vrátit aplikaci do parametru **Reason** volání MQGET; doporučuje se následující:

- Pokud má parametr *Reason* hodnotu MQRC_TRUNCATED_MSG_ACCEPTED na vstupu do uživatelské procedury, nesmí být pole *Reason* a *CompCode* změněna bez ohledu na to, zda je převod úspěšný nebo neúspěšný.

  (Pokud pole *CompCode* není MQCC_OK, aplikace, která načte zprávu, může identifikovat selhání převodu porovnáním vrácených hodnot *Encoding* a *CodedCharSetId* v deskriptoru zpráv s požadovanými hodnotami; naopak, aplikace nemůže rozlišit oříznutou zprávu od zprávy, která vyrovnávací paměť nabyla. Z tohoto důvodu musí být MQRC_TRUNCATED_MSG_ACCEPTED vrácen v preferencích k libovolným z důvodů, které indikují selhání převodu.)
- Pokud má *Reason* jakoukoli jinou hodnotu na vstupu do výstupu:
  - Pokud je konverze úspěšná, *CompCode* musí být nastavena na MQCC_OK a *Reason* nastaveno na MQRC_NONE.
  - Pokud převod selže nebo se zpráva rozbalí a musí být oseknuta tak, aby se vešla do vyrovnávací paměti, *CompCode* musí být nastavena na hodnotu MQCC_WARNING (nebo ponechána nezměněná) a hodnota *Reason* nastavena na jednu z uvedených hodnot, aby byla uvedena povaha selhání.

    Všimněte si, že je-li zpráva po převodu příliš velká pro vyrovnávací paměť, musí být zkrácena pouze v případě, že aplikace, která vydala volání MQGET, byla uvedena v rámci volby MQGMO_ACCEPT_TRUNCATED_MSG:

    - Pokud byla zadána tato volba, bude vrácena příčina MQRC_TRUNCATED_MSG_ACCEPTED.
    - Pokud tuto volbu nezadáte, bude vrácena nekonvertovaný zpráva s kódem příčiny MQRC_CONVERTED_MSG_TOO_BIG.

Uvedené kódy příčiny jsou doporučeny pro použití uživatelskou procedurou k určení důvodu selhání převodu, ale tato procedura může vracet jiné hodnoty ze sady kódů MQRC_ *, pokud to bude považováno za vhodné. Kromě toho je rozsah hodnot MQRC_APPL_FIRST přes MQRC_APPL_LAST alokován pro použití uživatelskou procedurou k označení podmínek, které chce uživatelská procedura komunikovat s aplikací, která vydala volání MQGET.

Poznámka: Pokud zprávu nelze úspěšně převést, při ukončení se musí v poli *ExitResponse* vrátit MQXDR_CONVERSION_FAILED, aby mohl správce front vrátit nepřevedenou zprávu. To je pravda bez ohledu na kód příčiny vrácený v poli *Reason* .

NEJPRVE 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 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

MQRC_NOT_CONVERTED
(2119, X'847 ') Data zprávy nejsou převedena.

CHYBA 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.

CHYBA MQRC_SOURCE_FLOAT_ENC_ERROR
(2114, X'842 ') Kódování čísel s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') Kódování celého čísla zdroje nebylo rozpoznáno.

CHYBA MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_DECIMAL_ENC_ERROR
(2117, X'845 ') Packed-decimal encoding specified by receiver not recognized.

MQRC_TARGET_FLOAT_ENC_ERROR
(2118, X'846 ') Kódování čísel s pohyblivou řádovou čárkou určené příjemcem není rozpoznáno.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') Cílové celé číslo kódování nebylo rozpoznáno.

MQRC_TRUNCATED_MSG_ACCEPTED
(2079, X'81F') Byla vrácena oříznutá zpráva (zpracování dokončeno).

Jedná se o vstupní/výstupní pole pro ukončení.

`StrucId`
Typ: MQCHAR4

Identifikátor struktury.Hodnota musí být:

ID_STRUKTURY MQDXP_STRUCTION_ID
Identifikátor pro strukturu výstupního parametru konverze dat.

Pro programovací jazyk C je také definována konstanta MQDXP_STRUC_ID_ARRAY; hodnota 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 parametru výstupního bodu převodu dat.

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQDXP_CURRENT_VERSION
Aktuální verze struktury parametru ukončení konverze dat.

Poznámka: Když je představena nová verze této struktury, rozvržení existující součásti se nezmění. Ukončení musí proto zkontrolovat, zda je pole *`Version`* rovno nebo větší než nejnižší verze, která obsahuje pole, která musí uživatelská procedura použít.

Toto je vstupní pole pro ukončení.

Deklarace C

```
typedef struct tagMQDXP MQDXP;
struct tagMQDXP {
  MQCHAR4  StrucId;         /* Structure identifier */
  MQLONG   Version;         /* Structure version number */
  MQLONG   ExitOptions;     /* Reserved */
```

```
  MQLONG   AppOptions;      /* Application options */
  MQLONG   Encoding;        /* Numeric encoding required by
                               application */
  MQLONG   CodedCharSetId;  /* Character set required by application */
  MQLONG   DataLength;      /* Length in bytes of message data */
  MQLONG   CompCode;        /* Completion code */
  MQLONG   Reason;          /* Reason code qualifying CompCode */
  MQLONG   ExitResponse;    /* Response from exit */
  MQHCONN  Hconn;           /* Connection handle */
  PMQIEP   pEntryPoints;    /* Address of the MQIEP structure */
};
```

Deklarace 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.
```

Deklarace asembleru System/390

```
MQDXP                 DSECT
MQDXP_STRUCID         DS   CL4  Structure identifier
MQDXP_VERSION         DS   F    Structure version number
MQDXP_EXITOPTIONS     DS   F    Reserved
MQDXP_APPOPTIONS      DS   F    Application options
MQDXP_ENCODING        DS   F    Numeric encoding required by application
MQDXP_CODEDCHARSETID  DS   F    Character set required by application
MQDXP_DATALENGTH      DS   F    Length in bytes of message data
MQDXP_COMPCODE        DS   F    Completion code
MQDXP_REASON          DS   F    Reason code qualifying COMPCODE
MQDXP_EXITRESPONSE    DS   F    Response from exit
MQDXP_HCONN           DS   F    Connection handle
*
MQDXP_LENGTH          EQU  *-MQDXP
                      ORG  MQDXP
MQDXP_AREA            DS   CL(MQDXP_LENGTH)
```

MQXCNVC-Převod znaků

Volání MQXCNVC převádí znaky z jedné znakové sady do jiné pomocí programovacího jazyka C.

Toto volání je součástí produktu IBM MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce produktu IBM MQ .

Pozn.: Volání lze použít jak z aplikace, tak z výstupních prostředí pro převod dat.

Syntaxe

MQXCNVC (*Hconn*, *Options*, *SourceCCSID*, *SourceLength*, *SourceBuffer*, *TargetCCSID*, *TargetLength*, *TargetBuffer*, *DataLength*, *CompCode*, *Reason*)

Parametry

Hconn

Typ: MQHCONN-vstup

Tento manipulátor představuje připojení ke správci front.

Při ukončení převodu dat je Hconn obvykle manipulační prostředek, který je předán uživatelské proceduře pro převod dat v poli Hconn struktury MQDXP. Tento manipulátor nemusí být nutně stejný jako popisovač určený aplikací, která vydala volání MQGET.

V systému IBM ilze pro produkt Hconnzadat následující speciální hodnotu:

MQC_DEF_HCONN
Výchozí popisovač připojení.

Pokud spustíte aplikaci CICS TS 3.2 nebo vyšší, ujistěte se, že ukončovací program pro převod znaků, který vyvolá volání MQXCNVC, je definován jako OPENAPI. Tato definice zabraňuje chybě MQRC_HCONN_ERROR v roce 2018 způsobená nesprávným připojením a umožňuje dokončení příkazu MQGET.

Volby

Typ: MQLONG-vstup

Volby, které řídí akci MQXCNVC.

Může být uvedena nula nebo více popsaných voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Volba Výchozí převod: Tato volba určuje použití výchozí konverze znaků:

PŘEVOD MQDCC_DEFAULT_VERSION
Výchozí převod.

Tato volba uvádí, že lze použít výchozí převod znaků, pokud jedna nebo obě znakové sady určené ve volání nejsou podporovány. To umožňuje správci front použít výchozí znakovou sadu určenou pro instalaci, která se bude při převodu řetězce přibližovat zadané znakové sadě.

Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězce je to, že některé znaky mohou být nesprávně převedeny. Tomu lze zabránit tak, že použijete v řetězci pouze znaky, které jsou společné jak pro uvedenou znakovou sadu, tak pro výchozí znakovou sadu.

Výchozí znakové sady jsou definovány volbou konfigurace, když je správce front instalován nebo restartován.

Není-li parametr MQDCC_DEFAULT_CONVERSION zadán, bude správce front používat k převodu řetězce pouze určené znakové sady a volání selže, pokud není 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 vyplnění převedeného řetězce s mezerami nebo zahodit nevýznamné koncové znaky za účelem převedení převedeného řetězce na cílovou vyrovnávací paměť:

MQDCC_FILL_TARGET_BUFFER.
Vyplnit cílovou vyrovnávací paměť.

Tato volba vyžaduje provedení převodu takovým způsobem, aby byla cílová vyrovnávací paměť zcela vyplněna:

- Jsou-li při převodu zadány koncové mezery, jsou za účelem vyplnění cílové vyrovnávací paměti přidány koncové mezery.
- Pokud se řetězec při převodu rozvine, koncové znaky, které nejsou významné, budou vyřazeny, aby převedený řetězec vešel do cílové vyrovnávací paměti. Je-li to možné provést úspěšně, volání skončí s MQCC_OK a s kódem příčiny MQRC_NONE.

  Pokud existuje příliš málo nevýznamných koncových znaků, je velká část řetězce tak, jak se vejde do cílové vyrovnávací paměti, a volání je dokončeno s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_TOPO_BIG.

  Nevýznamné znaky jsou:

  – Koncové mezery

  – Znaky následující za prvním znakem null v řetězci (ale kromě prvního znaku null samotného)
- Pokud je řetězec, `TargetCCSID`a `TargetLength` takový, že cílová vyrovnávací paměť nemůže být plně nastavena s platnými znaky, volání selže s chybou MQCC_FAILED a s kódem příčiny MQRC_TARGET_LENGTH_ERROR. K tomu může dojít, když `TargetCCSID` je čistá DBCS znaková sada (jako je UTF-16), ale `TargetLength` uvádí délku, která je lichým počtem bajtů.
- `TargetLength` může být menší než nebo větší než `SourceLength`. Při návratu z MQXCNVC má `DataLength` stejnou hodnotu jako `TargetLength`.

Není-li tato volba zadána, postupujte takto:

- Řetězec se může podle potřeby ve vyrovnávací paměti podle potřeby uzavírat nebo rozšiřovat v rámci cílové vyrovnávací paměti. Nevýznamné koncové znaky nejsou přidány nebo zrušeny.

  Pokud se převedený řetězec vejde do cílové vyrovnávací paměti, volání skončí s MQCC_OK a s kódem příčiny MQRC_NONE.

  Je-li převedený řetězec pro cílovou vyrovnávací paměť příliš velký, je velká část řetězce jako uložení umístěna do cílové vyrovnávací paměti a volání je dokončeno s MQCC_WARNING a kódem příčiny MQRC_CONVERTED_TOPO_BIG. Povšimněte si, že v tomto případě může být vrácen méně než `TargetLength` bajtů.
- `TargetLength` může být menší než nebo větší než `SourceLength`. Při návratu z MQXCNVC je `DataLength` menší než nebo rovno `TargetLength`.

Tato volba je podporována v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows

Volby kódování: Popsané volby lze použít k uvedení celočíselných kódování zdrojového a cílového řetězce. Relevantní kódování je použito pouze v případě, že odpovídající identifikátor znakové sady označuje, že znázornění znakové sady v hlavní paměti je závislé na kódování použité pro binární celá čísla. Toto se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Kódování je ignorováno, pokud znaková sada je jednobajtová znaková sada (SBCS), nebo vícebajtová znaková sada s reprezentací v hlavní paměti, která není závislá na celočíselném kódování.

Je třeba zadat pouze jednu 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` . Je-li však zdrojové nebo cílové kódování získáno z pole `Encoding` v produktu MQMD nebo v jiné struktuře, je třeba provést následující zpracování:

1. Celočíselné kódování musí být extrahováno z pole `Encoding` odstraněním plovoucího a packed-decimálního kódování; podrobnosti o tom, jak to provést, viz “Analyzování kódování” na stránce 892 .
2. Celočíselné kódování, které je výsledkem kroku 1, musí být vynásobeno příslušným faktorem, než bude přidáno do pole `Options` . Jedná se o následující faktory:
   - MQDCC_SOURCE_ENC_FACTOR pro zdrojové kódování
   - Objekt MQDCC_TARGET_ENC_FACTOR pro cílové kódování

Následující příklad kódu 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 tato volba zadána, bude výchozí hodnota kódování nastavena na nedefinované (MQDCC_ * _ENC_UNDEFINED). Ve většině případů to nemá vliv na úspěšné dokončení volání MQXCNVC. Je-li však odpovídající znaková sada vícebajtová znaková sada se znázorněním, která je závislá na kódování (například znaková sada UTF-16), volání selže s kódem příčiny MQRC_SOURCE_INTEGRER_ENC_ERROR nebo MQRC_TARGET_INTEGRER_ENC_ERROR.

Volby kódování jsou podporovány v následujících prostředích:

- AIX
- IBM i
- Linux
- Windows
- z/OS

Výchozí volba: Pokud žádná z výše popsaných voleb není uvedena, lze použít následující volbu:

MQDCC_NONE
Nejsou uvedeny žádné volby.

MQDCC_NONE je definována pro dokumentaci programu podpory. Není určeno, že tato volba je použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze zjistit.

SourceCCSID
Typ: MQLONG-vstup

Jedná se o identifikátor kódované znakové sady vstupního řetězce v `SourceBuffer`.

SourceLength
Typ: MQLONG-vstup

Délka vstupního řetězce v `SourceBuffer` je délka (v bajtech); 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, do níž má být produkt `SourceBuffer` převeden.

TargetLength
Typ: MQLONG-vstup

Toto je délka výstupní vyrovnávací paměti `TargetBuffer` , v bajtech; musí být nula nebo větší. Může být menší než nebo větší než `SourceLength`.

TargetBuffer
Typ: MQCHAR x TargetLength -výstup

To je řetězec poté, co byl převeden na znakovou sadu definovanou `TargetCCSID`. Konvertovaný řetězec může být kratší nebo delší než nekonvertovaný řetězec. Argument **DataLength** udává počet platných bajtů, které byly vráceny.

DataLength
Typ: MQLONG-výstup

Jedná se o délku řetězce vráceného ve výstupní vyrovnávací paměti `TargetBuffer`. Konvertovaný řetězec může být kratší nebo delší než nekonvertovaný řetězec.

CompCode
Typ: MQLONG-výstup

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující `CompCode`.

Je-li `CompCode` MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li CompCode MQCC_WARNING:

MQRC_CONVERTED_MSG_TOO_BIG
(2120, X'848 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

Je-li CompCode MQCC_FAILED:

CHYBA MQRC_DATA_LENGTH_ERROR
(2010, X'7DA') Parametr délky dat není platný.

CHYBA MQRC_DBCS_ERROR
(2150, X'866 ') DBCS řetězec není platný.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

CHYBA MQRC_SOURCE_BUFFER_ERROR
(2145, X'861 ') Parametr zdrojové vyrovnávací paměti není platný.

CHYBA MQRC_SOURCE_CCSID_ERROR
(2111, X'83F') Identifikátor zdrojové kódované znakové sady není platný.

MQRC_SOURCE_INTEGER_ENC_ERROR
(2112, X'840 ') Kódování celého čísla zdroje nebylo rozpoznáno.

CHYBA MQRC_SOURCE_LENGTH_ERROR
(2143, X'85F') Parametr délky zdroje není platný.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

MQRC_TARGET_BUFFER_ERROR
(2146, X'862 ') Cílový parametr vyrovnávací paměti není platný.

CHYBA MQRC_TARGET_CCSID_ERROR
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

MQRC_TARGET_INTEGER_ENC_ERROR
(2116, X'844 ') Cílové celé číslo kódování 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 najdete v tématu Zprávy a kódy příčin.

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 COBOL (pouze 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.
```

Deklarace v modulu S/390

```
         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-Ukončení převodu dat

Volání MQ_DATA_CONV_EXIT popisuje parametry, které jsou předány uživatelské proceduře pro převod dat.

Správcem front není poskytnut žádný vstupní bod s názvem MQ_DATA_CONV_EXIT (viz poznámka o použití 11).

Tato definice je součástí produktu IBM MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce produktu IBM MQ .

Syntaxe

MQ_DATA_CONV_EXIT (*DataConvExitParms*, *MsgDesc*, *InBufferLength*, *InBuffer*, *OutBufferLength*, *OutBuffer*)

Parametry

DataConvExitParms
Typ: MQDXP-input/output

Tato struktura obsahuje informace vztahující se k vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označovaly výsledek převodu. Podrobnosti o polích v této struktuře viz “MQDXP-Data-konverze-výstupní parametr” na stránce 904 .

MsgDesc
Typ: MQMD-I/O

Na vstupu do výstupu je to deskriptor zprávy přidružený k datům zprávy předaným ukončení v parametru **InBuffer** .

Poznámka: Parametr **MsgDesc** předaný do uživatelské procedury je vždy nejnovější verzí MQMD, kterou podporuje správce front, který vyvolá ukončení. Pokud má být uživatelská procedura přenositelná mezi různými prostředími, ukončí uživatelská procedura pole Version v produktu MsgDesc a ověří, zda jsou pole, která uživatelská procedura potřebuje k přístupu, přítomna ve struktuře.

V následujících prostředích je předání předáno version-2 MQMD:

- AIX
- IBM i
- Linux
- Windows

Ve všech ostatních prostředích, která podporují uživatelskou proceduru pro převod dat, je ukončena procedura version-1 MQMD.

Výstup na výstupu změní pole Encoding a CodedCharSetId na hodnoty požadované aplikací, pokud byl převod úspěšný; tyto změny se odrazí zpět do aplikace. Všechny ostatní změny, které má uživatelská procedura ke struktuře, se budou ignorovat. Neodrážejí se to zpět do aplikace.

Pokud uživatelská procedura vrátí hodnotu MQXDR_OK v poli ExitResponse struktury MQDXP, ale nezmění pole Encoding nebo CodedCharSetId v deskriptoru zpráv, správce front vrátí pro tato pole hodnoty, které měly odpovídající pole ve struktuře MQDXP na vstupu do uživatelské procedury.

InBufferDélka
Typ: MQLONG-vstup

Délka v bajtech InBuffer.

Toto je délka vstupní vyrovnávací paměti `InBuffer`a uvádí počet bajtů, které mají být zpracovány uživatelskou procedurou. `InBufferLength` je menší z hodnot délky dat zprávy před převodem a délka vyrovnávací paměti poskytnutá aplikací na volání MQGET.

Hodnota je vždy větší než nula.

InBuffer
Typ: MQBYTExInBufferLength -Vstup

Vyrovnávací paměť obsahující nepřevedené zprávy.

Obsahuje data zprávy před převodem. Pokud uživatelská procedura nemůže převést data, vrátí správce front obsah této vyrovnávací paměti do aplikace po dokončení uživatelské procedury.

Poznámka: Uživatelská procedura by neměla měnit `InBuffer` ; je-li tento parametr změněn, nejsou výsledky definovány.

V programovacím jazyku C je tento parametr definován jako ukazatel na hodnotu typu void.

Délka OutBuffer
Typ: MQLONG-vstup

Délka v bajtech `OutBuffer`.

Jedná se o délku výstupní vyrovnávací paměti `OutBuffer`a je stejná jako délka vyrovnávací paměti poskytnutá aplikací na volání MQGET.

Hodnota je vždy větší než nula.

OutBuffer
Typ: MQBYTExOutBufferLength -výstup

Vyrovnávací paměť obsahující převedenou zprávu.

Na výstupu z uživatelské procedury, pokud byl převod úspěšný (jak je indikováno hodnotou MQXDR_OK v poli `ExitResponse` parametru **`DataConvExitParms`**), obsahuje `OutBuffer` data zprávy, která mají být doručena aplikaci, v požadovaném znázornění. Pokud byl převod neúspěšný, budou všechny změny provedené v této vyrovnávací paměti ignorovány.

V programovacím jazyku C je tento parametr definován jako ukazatel na hodnotu typu void.

Poznámky k použití

1. Uživatelská procedura pro převod dat je uživatelská procedura, která přijímá řízení během zpracování volání MQGET. Funkce, kterou provádí uživatelská procedura pro převod dat, je definována poskytovatelem uživatelské procedury, avšak tato procedura musí odpovídat pravidlům, která jsou zde popsána, a v přidružené struktuře parametrů MQDXP.

   Programovací jazyky, které lze použít pro ukončení převodu dat, jsou určovány prostředím.
2. Ukončení je vyvoláno pouze v případě, že všechny následující příkazy jsou pravdivé:
   - Volba MQGMO_CONVERT je určena v rámci volání MQGET.
   - Pole `Format` v deskriptoru zprávy není MQFMT_NONE
   - Zpráva ještě není v požadované reprezentaci. To znamená, že jedna nebo obě zprávy `CodedCharSetId` a `Encoding` se liší od hodnoty zadané aplikací v deskriptoru zpráv dodaném při volání MQGET.
   - Správce front dosud neprovedl převod úspěšně.
   - Délka vyrovnávací paměti aplikace je větší než nula.
   - Délka dat zprávy je větší než nula.
   - Kód příčiny tak daleko během operace MQGET je MQRC_NONE nebo MQRC_TRUNCATED_MSG_ACCEPTED
3. Když je zapisována uživatelská procedura, zvažte její kódování způsobem, který umožňuje převod zpráv, které byly oříznuty. Zkrácené zprávy mohou nastat následujícími způsoby:

- Přijímající aplikace poskytuje vyrovnávací paměť, která je menší než zpráva, ale určuje volbu MQGMO_ACCEPT_TRUNCATED_MSG na volání MQGET.

  V tomto případě má pole Reason v parametru **DataConvExitParms** na vstupu do výstupu hodnotu MQRC_TRUNCATED_MSG_ACCEPTED.

- Odesílatel zprávy jej zkrátil, než jej odešle. Tato situace může nastat například u zpráv sestavy (další podrobnosti viz “Převod zpráv sestav” na stránce 903).

  V tomto případě má pole Reason v parametru **DataConvExitParms** na vstupu do výstupu hodnotu MQRC_NONE (pokud přijímající aplikace poskytla vyrovnávací paměť, která byla dostatečně velká pro tuto zprávu).

Tudíž hodnota pole Reason na vstupu do ukončení nemůže být vždy použita k rozhodnutí, zda byla zpráva zkrácena.

Charakteristickým rysem zkrácené zprávy je, že délka poskytnutá uživatelské proceduře v argumentu **InBufferLength** je menší než délka zahrnutá v názvu formátu, který je obsažen v poli Format v deskriptoru zpráv. Ukončení by proto mělo zkontrolovat hodnotu InBufferLength před pokusem o převod jakýchkoli dat; uživatelská procedura by neměla předpokládat, že bylo poskytnuto úplné množství dat, které je odvozeno od názvu formátu.

Pokud uživatelská procedura nebyla zapsána pro převod oříznutých zpráv a InBufferLength je nižší než očekávaná hodnota, vrátí uživatelská procedura MQXDR_CONVERSION_FAILED v poli ExitResponse parametru **DataConvExitParms** , přičemž pole CompCode a Reason jsou nastavena na hodnotu MQCC_WARNING a MQRC_FORMAT_ERROR.

Pokud byla uživatelská procedura zapsána pro převod zkrácených zpráv, bude uživatelská procedura konvertována co nejvíce dat (viz další poznámka o použití), přičemž se dbá na to, aby nedošlo k pokusu o prozkoumání nebo konverzi dat za koncem InBuffer. Je-li konverze úspěšně dokončena, ukončí uživatelská procedura pole Reason v parametru **DataConvExitParms** nezměněno. Tento příkaz vrací MQRC_TRUNCATED_MSG_ACCEPTED, pokud byla zpráva zkrácena správcem front příjemce a MQRC_NONE, pokud byla zpráva zkrácena odesílatelem zprávy.

Je také možné, aby se zpráva rozšiřovala během převodu, do bodu, kdy je větší než OutBuffer. V tomto případě se musí výstup rozhodnout, zda má být zpráva zkrácen; pole AppOptions v parametru **DataConvExitParms** udává, zda přijímající aplikace specifikovanou volbu MQGMO_ACCEPT_TRUNCATED_MSG.

4. Obecně jsou všechna data ve zprávě poskytnutá uživatelské proceduře v produktu InBuffer převedena nebo nic z toho není. Výjimka však nastane, pokud je zpráva zkrácena, buď před převodem, nebo během převodu; v tomto případě může být na konci vyrovnávací paměti nekompletní položka (například: 1 bajt dvojbajtového znaku, nebo 3 bajty 4bajtového celého čísla). V této situaci zvažte vynechání nekompletní položky a nastavení nepoužitých bajtů v parametru OutBuffer na hodnotu null. Avšak úplné prvky nebo znaky v poli nebo řetězci by měly být převedeny.
5. Když je poprvé ukončena uživatelská procedura, správce front se pokusí 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 název objektu, který obsahuje uživatelskou proceduru, ačkoli ne všechna prostředí toto nastavení vyžadují.
6. A new copy of the exit is loaded when an application attempts to retrieve the first message that uses that Format since the application connected to the queue manager. V případě aplikací CICS nebo IMS to znamená, že se subsystém CICS nebo IMS připojil ke správci front. Novou kopii lze také načíst v jiných dobách, pokud správce front zahodil dříve načtenou kopii. Z tohoto důvodu se nesmí uživatelská procedura nepokoušet o použití statického úložiště pro komunikaci informací z jednoho vyvolání procedury do dalšího-může být uvolněno mezi oběma vyvoláními.
7. Existuje-li uživatelská procedura se stejným názvem jako jeden z vestavěných formátů podporovaných správcem front, uživatelská procedura nemůže nahradit vestavěnou rutinu převodu. Pouze okolnosti, za kterých je taková východa, jsou:

   - If the built-in conversion routine cannot handle conversions to or from either the CodedCharSetId or Encoding involved, or

- Pokud vestavěná rutina převodu selhala při převodu dat (například proto, že existuje pole nebo znak, který nelze převést).

8. Rozsah ukončení je závislý na prostředí. Názvy `Format` musí být vybrány, aby se minimalizovalo riziko konfliktů s jinými formáty. Zvažte spuštění se znaky, které identifikují aplikaci definující název formátu.
9. Ukončení převodu dat se spouští v prostředí, jako je tomu u programu, který vydal volání MQGET; prostředí zahrnuje adresní prostor a profil uživatele (kde je to vhodné). Program může být agent kanálu zpráv odesílající zprávy do cílového správce front, který nepodporuje převod zpráv. Uživatelská procedura nemůže ohrozit integritu správce front, protože není spuštěna v prostředí správce front.
10. Jediné volání MQI, které lze použít při ukončení, je MQXCNVC; pokus o použití jiných volání MQI selže s kódem příčiny MQRC_CALL_IN_PROGRESS nebo jinými nepředvídatelnými chybami.
11. Správcem front není poskytnut žádný vstupní bod s názvem MQ_DATA_CONV_EXIT. Pro název MQ_DATA_CONV_EXIT v programovacím jazyku C je však k dispozici položka `typedef` a lze ji použít k deklarování uživatelské procedury, aby bylo zaručeno, že jsou parametry správné. Název uživatelské procedury musí být stejný jako název formátu (název obsažený v poli `Format` v produktu MQMD), ačkoli tento název není povinný ve všech prostředích.

    Následující příklad ukazuje, jak může být uživatelská procedura, která zpracovává formát `MYFORMAT` , deklarována v programovacím jazyce 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/OSv platnosti také uživatelská procedura překřížení rozhraní API, volá se po ukončení převodu 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 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).
```

Deklarace asembleru System/390

```
         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 zadané jako prvky MQRFH2 .

Vlastnosti deskriptoru nezprávy lze zadat jako prvky ve složkách záhlaví MQRFH2 . Přehled prvků MQRFH2 je určován jako vlastnosti.

Tím je zachována kompatibilita s předchozími verzemi klientů produktu IBM MQ JMS a XMS . Tento oddíl popisuje, jak určit vlastnosti v záhlaví MQRFH2 .

Chcete-li jako vlastnosti použít prvky MQRFH2 , určete prvky podle popisu v části Použití produktu IBM MQ classes for Java . Tyto informace doplňují informace popsané v části “MQRFH2 -Pravidla a formátování záhlaví 2” na stránce 524.

Mapování datových typů vlastností na datové typy MQRFH2

Toto téma poskytuje informace o typech vlastností zpráv mapovaných na odpovídající datové typy MQRFH2 .

Tabulka 636. Podporované datové typy MQRFH2

Typ vlastnosti zprávy	Datový typ MQRFH2
MQBYTE []	bin.hex
MQBOOL	typ boolean

Tabulka 636. Podporované datové typy MQRFH2 (pokračování)	
Typ vlastnosti zprávy	**Datový typ MQRFH2**
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR []	řetězec

Jakýkoli prvek bez datového typu je považován za řetězec typu "string".

S datovým typem MQRFH2 datového typu `int`, což znamená celé číslo nespecifikované velikosti, se zachází jako s `i8`.

Hodnota null je indikována atributem prvku `xsi:nil='true'`. Nepoužívejte atribut `xsi:nil='false'` pro jiné hodnoty než null.

Následující vlastnost má například hodnotu null:

```
<NullProperty xsi:nil='true'></NullProperty>
```

Vlastnost typu byte nebo znakový řetězec může mít prázdnou hodnotu. Tento prvek je představován prvkem MQRFH2 s hodnotou prvku s nulovou délkou.

Např. 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 všech vlastností definovaných aplikací produktu JMS , které jsou přidruženy ke zprávě. Ve složce `<usr>` nejsou povoleny skupiny.

Záhlaví MQRFH2 a produkt JMS podporuje následující další složky:

- `<mq>`

  Tato složka se používá a je vyhrazena pro vlastnosti definované produktem MQ, které jsou používány produktem IBM MQ.

- `<mq_usr>`

  Tuto složku lze použít k přenosu libovolných vlastností definovaných aplikací, které nejsou vystaveny jako uživatelem definované vlastnosti produktu JMS , protože vlastnosti nemusí splňovat požadavky vlastnosti JMS . Tato složka může obsahovat skupiny, které nemůže složka `<usr>` obsahovat.

- Jakákoli složka označená atributem `content='properties'` .

  Taková složka je ekvivalentem složky produktu `<mq_usr>` v obsahu.

- `<mqps>`

  Tato složka se používá pro vlastnosti publikování a odběru IBM MQ .

Produkt IBM MQ dále podporuje následující složky, které jsou již používány systémem WAS/SIB:

- <sib>

  Tato složka je používána a vyhrazena pro vlastnosti zpráv systému WAS/SIB, které nejsou odkryty jako vlastnosti produktu JMS nebo jsou mapovány na vlastnosti JMS_IBM_ *, jsou však vystaveny pro aplikace WAS/SIB; tyto vlastnosti zahrnují vlastnosti dopředného a zpětného směrování cest.

  Alespoň některé vlastnosti nelze vystavit jako vlastnosti JMS , protože se jedná o bajtová pole. Pokud vaše aplikace přidává vlastnosti do této složky, je tato hodnota buď ignorována, nebo odstraněna.

- <sib_usr>

  Tato složka je používána a vyhrazena pro vlastnosti zprávy uživatele WAS/SIB, které nelze vystavit jako uživatelské vlastnosti produktu JMS , protože nejsou podporované typy; jsou vystaveny aplikacím WAS/SIB.

  Jedná se o uživatelské vlastnosti, které lze získat nebo nastavit prostřednictvím rozhraní SIMessage, ale obsah pole bajtů je mapován na požadovanou hodnotu vlastnosti.

  Pokud aplikace IBM MQ zapíše do složky libovolný prvek `bin.hex` , aplikace pravděpodobně přijme `IOException`, protože se nejedná o formát očekávané k obnově. Přidáte-li něco jiného než prvek `bin.hex` , obdržíte `ClassCastException`.

  Nesnažte se zpřístupnit vlastnosti pro WAS/SIB pomocí této složky; místo toho použijte složku <usr> pro tento účel.

- <sib_context>

  Tato složka se používá pro vlastnosti zpráv systému WAS/SIB, které nejsou vystaveny pro uživatelské aplikace WAS/SIB nebo jako vlastnosti produktu JMS . Jedná se o vlastnosti zabezpečení a transakcí, které se používají pro webové služby a podobně.

  Vaše aplikace nesmí do této složky přidávat vlastnosti.

- <mqema>

  Tato složka byla použita pro WAS/SIB místo složky <mqext> .

Názvy složek MQRFH2 rozlišují velikost písmen.

Následující složky jsou vyhrazené, v libovolné směsi malých nebo velkých písmen:

- Všechny složky s předponou `mq` nebo `wmq` ; vyhrazeno pro použití produktem IBM MQ.
- Všechny složky s předponou `sib` ; vyhrazeno pro použití produktem WAS/SIB.
- Složky <Root> a <Body> , vyhrazené, ale nepoužité.

Následující složky nejsou rozpoznány jako obsahující vlastnosti zprávy:

- <psc>

  Používáno produktem IBM Integration Bus k přenosu zpráv příkazů publikování/odběru do zprostředkovatele.

- <pscr>

  Používá produkt IBM Integration Bus k ukládání informací ze zprostředkovatele v odezvě na zprávy příkazů publikování/odběru.

- Jakákoli složka není definována produktem IBM, která není označena atributem `content='properties'` .

Nezadávejte `content='properties'` ve složkách <psc> nebo <pscr> . Pokud tak učiníte, tyto složky budou považovány za vlastnosti a IBM Integration Bus pravděpodobně přestane fungovat podle očekávání.

Pokud vaše aplikace buduje zprávy s vlastnostmi, v záhlaví MQRFH2 , která má být rozpoznána jako záhlaví MQRFH2 obsahující vlastnosti, musí být záhlaví v seznamu záhlaví, která mohou být zřetězeny v záhlaví zprávy.

Před MQRFH2 může předcházet libovolný počet standardních záhlaví MQH nebo MQCIH, MQDLH, MQIIH, MQTM, MQTMC2, nebo MQXQH. Řetězec nebo MQCFH ukončí syntaktickou analýzu, protože nemohou být zřetězeny.

Je možné, že zpráva obsahuje více záhlaví MQRFH2 , přičemž všechny vlastnosti zprávy mají všechny vlastnosti zprávy. Složky se stejným názvem mohou existovat společně v různých záhlavích, pokud není jinak omezeno, např. WAS/SIB. Složky jsou považovány za jednu logickou složku, pokud jsou všechny ve významném záhlaví.

Zatímco složky z významných záhlaví nelze sloučit s těmito složkami v nevýznamných záhlavích, složky se stejným názvem v rámci výrazných záhlaví lze sloučit, odebráním případných konfliktních vlastností. Vaše aplikace nesmí záviset na rozvržení vlastností v rámci příslušné 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 IBMdefinovaných uživatelem, kromě složek <wmq> a <mq> , jsou analyzovány pro vlastnosti.

Složka MQRFH2 nemůže obsahovat smíšený obsah; složka nebo skupina může obsahovat buď skupiny, nebo vlastnosti, nebo hodnotu, nikoli však obojí.

Segment zprávy, buď první nebo následný segment, nemůže obsahovat IBM MQ definované vlastnosti jiné než tyto vlastnosti v deskriptoru zpráv. Proto vložení zprávy obsahující takové vlastnosti buď s nastavením MQMF_SEGMENT nebo MQMF_SEGMENTATION_ALLOWED, způsobí selhání operace put to s MQRC_SEGMENTATION_NOT_ALLOWED.

Skupiny zpráv však mohou obsahovat definované vlastnosti produktu IBM MQ .

Generování záhlaví MQRFH2

Pokud IBM MQ převádí vlastnosti zpráv do jejich znázornění MQRFH2 , musí přidat MQRFH2 do zprávy. Přidává MQRFH2 buď jako samostatné záhlaví, nebo ho sloučí s existujícím záhlavím.

Generování nových záhlaví MQRFH2 podle IBM MQ může narušit existující záhlaví ve zprávě. Aplikace, které analyzují vyrovnávací paměť zpráv pro záhlaví, si musí být vědomy toho, že se počet a pozice záhlaví ve vyrovnávací paměti mohou za určitých okolností změnit. IBM MQ se snaží minimalizovat dopad přidání vlastností na zprávu sloučením vlastností zprávy do existujícího záhlaví MQRFH2 , kde to může být. Pokusí se také minimalizovat dopad vložením generovaného MQRFH2 do pevné pozice vzhledem k ostatním záhlavím ve vyrovnávací paměti zpráv.

Vygenerované záhlaví MQRFH2 je umístěno za serverem MQMDa libovolný počet záhlaví MQXQH, MQRFHa MQDLH , ať už jsou v pořadí, v jakém pořadí jsou. Vygenerované záhlaví MQRFH2 se umístí těsně před první záhlaví, které není hlavičkou MQMD, MQXQH, MQDLHnebo MQRFH .

z/OS Na systémech z/OS je vygenerované záhlaví MQRFH2 vytvořeno v aplikaci CCSIDaplikace. To je definováno takto:

- Pro dávkové aplikace LE používající rozhraní DLL je CCSID CODESET přidružený k aktuálnímu národnímu prostředí v době, kdy je **MQCONN** vydáno (výchozí hodnota je 1047).
- Pro dávkové aplikace LE vázané s jednou z dávkových stubů produktu MQ je CCSID CODESET přidružená k aktuálnímu národnímu prostředí v době prvního volání MQI vydaného po **MQCONN** (výchozí hodnota je 1047).
- Pro dávkové aplikace non-LE běžící na vláknu z/OS UNIX System Services (z/OS UNIX) je hodnota CCSID hodnota THLICCSID v době prvního volání MQI vydaného po **MQCONN** (standardní hodnota je 1047).
- U jiných dávkových aplikací je CCSID správcem front CCSID .

V případě aplikací LE lze národní prostředí změnit pomocí volatelné služby setlocale() / CEESETL LE . U aplikací jiných než LE spuštěných na podprocesech z/OS UNIX lze hodnotu THLICCSID změnit pomocí makru mapování z/OS UNIX **BPXYTHLI**.

Pravidla pro slučování generovanou MQRFH2

Následující pravidla se vztahují ke sloučení vygenerovaného MQRFH2 s existujícím MQRFH2. Vygenerované záhlaví MQRFH2 je sloučeno s existujícím záhlavím MQRFH2 , pokud:

1. Existující MQRFH2 je ve stejné pozici IBM MQ umístí generovaný MQRFH2, nebo dřívější v řetězci záhlaví.
2. CCSID vygenerovaných vlastností je stejný jako u NameValueCCSID existujícího MQRFH2.

Jinak se vygenerované záhlaví umístí odděleně do vyrovnávací paměti, v místě popsaném před.

Pravidla pro slučování složek v existující struktuře MQRFH2

Pokud jsou vlastnosti zprávy sloučeny do stávajícího produktu MQRFH2, je existující MQRFH2 skenován pro složky, které odpovídají vlastnostem zprávy, a sloučí je. Pokud odpovídající složka neexistuje, přidá se nová složka na konec stávajících složek. Pokud existuje odpovídající složka, prohledá se složka. Všechny vyhovující vlastnosti se přepíší. Všechny nové položky se přidají na konec složky.

Omezení složky MQRFH2

Přehled omezení složek v záhlaví MQRFH2

Omezení MQRFH2 se vztahují na následující složky:

- Názvy prvků ve složce <usr> nesmí začínat předponou JMS ; Tyto názvy vlastností jsou rezervovány pro použití produktem JMS a nejsou platné pro uživatelem definované vlastnosti.

  Takový název prvku nezpůsobuje selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu IBM MQ .
- Názvy prvků ve složce produktu <usr> nesmějí být v žádné směsi malých nebo velkých písmen, NULL, TRUE, FALSE, NOT, AND, OR, BETWEEN, LIKE, IN, IS a ESCAPE. Tyto názvy odpovídají klíčovým slovům SQL a usnadňují analýzu selektorů, protože <usr> je výchozí složka použitá v případě, že není určena žádná složka pro konkrétní vlastnost v selektoru.

  Takový název prvku nezpůsobuje selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu IBM MQ .
- Model obsahu složky <usr> je následující:
  - Jako název prvku lze použít libovolný platný název XML a za předpokladu, že neobsahuje dvojtečku.
  - Povoleny jsou pouze jednoduché prvky, nikoli vnořené složky.
  - Všechny prvky mají výchozí typ řetězce, pokud není změněn atributem dt="xxx" .
  - Všechny prvky jsou volitelné, ale neměly by se ve složce vyskytovat více než jednou.
- Názvy prvků ve všech složkách, které mají obsahovat vlastnosti zprávy, nesmí obsahovat tečku (.) (Znak Unicode U+002E), protože se používá v názvech vlastností k označení hierarchie.

  Takový název prvku nezpůsobuje selhání syntaktické analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu IBM MQ .

Obecně platí, že záhlaví MQRFH2 , která obsahují platná data ve stylu XML, může být analyzována produktem IBM MQ bez selhání, ačkoli některé prvky MQRFH2 nejsou přístupné prostřednictvím rozhraní API vlastností zprávy produktu IBM MQ .

Konflikty názvů prvků MQRFH2

Přehled konfliktů v rámci názvů prvků MQRFH2 .

K vlastnosti zprávy může být připojena pouze jedna hodnota. Pokud se pokus o přístup k vlastnosti vede ke konfliktu hodnot, je jeden z nich vybrán přednostně nad jiným.

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 ta, která je nejblíž k záhlaví zprávy.

Toto platí, pokud jsou dvě nebo více složek stejného názvu obsaženy v různých významných záhlavích MQRFH2 v rámci stejné zprávy.

Konflikt může být výsledkem zpracování volání MQGET po nastavení vlastnosti deskriptoru mimo zprávu: volání MQSETMP a přímo v záhlaví MQRFH2 v záhlaví s přímým přístupem.

Pokud k tomu dojde, vlastnost přidružená ke zprávě prostřednictvím volání rozhraní API bude mít přednost před jednou v datech zprávy, tj. v záhlaví s přímým přístupem MQRFH2 . Dojde-li ke konfliktu, má se za to, že logicky předchází data zprávy.

Mapování z názvů vlastností na složku MQRFH2 a názvy prvků

Přehledné informace o rozdílech mezi názvy vlastností a názvy prvků v záhlaví MQRFH2 .

Používáte-li některá z definovaných rozhraní API, která v konečném důsledku generují záhlaví MQRFH2 , aby určovala vlastnosti zprávy (například MQ JMS), nemusí být název vlastnosti nutně název prvku ve složce MQRFH2 .

Proto se mapování vyskytne z názvu vlastnosti na prvek MQRFH2 a v opačném případě vezme v úvahu název složky, který obsahuje prvek, a název prvku. Některé příklady z produktu IBM MQ classes for JMS jsou již dokumentovány v příručce Použití IBM MQ classes for Java.

Tabulka 637. 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
xxx (definovaný uživatelem, kde xxx nezačíná s JMS)	usr	xxx

Proto, když aplikace JMS přistupuje k vlastnosti JMSDestination , je tato mapa mapována na prvek Dst ve složce <jms> .

Při určování vlastností jako prvků MQRFH2 definuje produkt IBM MQ své prvky takto:

Tabulka 638. 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>

Pokud se například aplikace IBM MQ pokusí o přístup k vlastnosti Property1 , je tato mapa mapována na prvek Property1 ve složce <usr> . Vlastnost wmq.Property2 se mapuje na vlastnost Property2 ve složce <wmq> .

Pokud název vlastnosti obsahuje více než jeden. je použit název prvku MQRFH2 , který následuje po konečném znění. znakové a MQRFH2 skupiny se používají k vytvoření hierarchie; jsou povoleny vnořené skupiny MQRFH2 .

Vlastnosti záhlaví JMS a specifické pro poskytovatele, které jsou obsaženy ve struktuře MQRFH2 ve složkách <mcd>, <jms>a <mqext> , jsou přístupné aplikaci IBM MQ pomocí krátkých názvů definovaných v části Použití produktu IBM MQ classes for Java .

K uživatelem definovaným vlastnostem produktu JMS se přistupuje ze složky <usr> . Aplikace IBM MQ může použít složku <usr> pro své vlastnosti aplikace, je-li přijatelné, aby se vlastnost zobrazovala pro aplikace produktu JMS jako jedna ze svých vlastností definovaných uživatelem.

Pokud to není přijatelné, zvolte jinou složku; složka <wmq_usr> se poskytuje jako standardní umístění pro takové vlastnosti, které nejsou typuJMS .

Vaše aplikace mohou určovat a používat všechny složky MQRFH2 s dobře definovaným použitím, které nejsou dokumentovány v produktu “Vlastnosti zadané jako prvky MQRFH2 .” na stránce 920 , pokud si povšimněte následujících:

1. Složka může být již používána nebo může být použita v budoucnu jinou aplikací a poskytuje nedefinovaný přístup k vlastnostem obsaženým uvnitř vlastností; viz Názvy vlastností pro doporučenou konvenci pojmenování pro názvy vlastností.
2. Vlastnosti nejsou přístupné pro předchozí verze klienta produktu IBM MQ classes for JMS nebo XMS , které mohou mít přístup pouze ke složce <usr> pro uživatelem definované vlastnosti.
3. Složka musí být označena atributem `content` s hodnotou nastavenou na `properties`, např. `content='properties'`.

   Produkt “MQSETMP-nastavení vlastnosti zprávy” na stránce 773 podle potřeby automaticky přidá tento atribut. Tento atribut nesmí být přidán do žádné z definovaných složek produktu IBM, například <jms> a <usr>. Pokud tak učiníte, bude zpráva odmítnuta klientem produktu IBM MQ classes for JMS před IBM WebSphere MQ 7.0. s `MessageFormatException`.

Protože složka <usr> je výchozím umístěním pro vlastnosti syntaxe <Property> , aplikace IBM MQ a aplikace JMS mají přístup ke stejné uživatelem definované hodnotě vlastnosti s použitím stejného názvu.

Vyhrazené názvy složek

Existuje několik vyhrazených názvů složek. Tyto názvy nemůžete používat jako předpony složek; například produkt `Root.Property1` nemá přístup k platné vlastnosti, protože je `Root` rezervováno. Následující seznam obsahuje vyhrazené názvy složek:

- Kořen
- Tělo
- Vlastnosti
- Prostředí
- LocalEnvironment
- DestinationList
- ExceptionList
- InputBody
- InputRoot
- InputProperties
- Prostředí InputLocal
- Seznam InputDestination
- Seznam InputException
- OutputRoot
- OutputLocalProstředí
- Seznam OutputDestination
- Seznam OutputException

Mapování polí deskriptoru vlastností na záhlaví MQRFH2

Když je vlastnost přeložena do prvku MQRFH2 , jsou použity následující atributy prvků k určení důležitých polí deskriptoru vlastností: Popisuje, jak jsou pole MQPD převedena na atributy prvku MQRFH2 .

Podpora

Pole deskriptoru vlastnosti podpory je rozděleno do tří atributů prvků.

- 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 jiných složkách, které obsahují vlastnosti.

Tabulka 639. Pole MQPD mapovaná na atributy prvku MQRFH2

Hodnota podpory	Atribut prvku MQRFH2	Hodnota atributu MQRFH2
PODPORA MQPD_SUPPORT_OPTIONAL	sa	volitelné Toto je výchozí hodnota.
POŽADOVÁNA PODPORA MQPD_SUPPORT_REQUIRED	sr	povinné
MQPD_SUPPORT_REQUIRED_IF_LOCAL	sx	lokální

Kontext

Pomocí atributu prvku produktu **context** označte kontext zprávy, do kterého patří vlastnost. Používejte pouze jednu hodnotu. Tento atribut prvku je platný na vlastnosti v libovolné složce, která obsahuje vlastnosti.

Tabulka 640. Kontextové hodnoty mapované na hodnoty atributu MQRFH2

Hodnota kontextu	Hodnota atributu MQRFH2
MQPD_NO_CONTEXT	Není Toto je výchozí hodnota.
M_KONTEXT MQPD_USER	uživatel

CopyOptions

Použijte atribut prvku **copy** k označení zpráv, do kterých se má vlastnost zkopírovat. Přijatelná je více než jedna hodnota; oddělte více hodnot čárkou. Například **copy='reply'** a **copy='publish,report'** jsou platné. Tento atribut prvku je platný na vlastnosti v libovolné složce, která obsahuje vlastnosti.

Poznámka: V definici atributu jsou platné jednoduché uvozovky nebo dvojité uvozovky, například **copy='reply'** nebo **copy="report"**

Tabulka 641. Hodnoty CopyOption mapované na hodnoty atributu MQRFH2

Hodnota CopyOption	Hodnota atributu MQRFH2
MQPD_COPY_FORWARD	objekt forward
MQPD_COPY_REPLY	reply
ZPRÁVA MQPD_COPY_REPORT	sestava
MQPD_COPY_PUBLISH	publikování
MQPD_COPY_ALL	vše Neuvádějte ji s žádnou jinou hodnotou. Je-li použit s jinou hodnotou, má přednost před jakoukoli hodnotou kromě **none**.

Tabulka 641. Hodnoty CopyOption mapované na hodnoty atributu MQRFH2 (pokračování)	
Hodnota CopyOption	**Hodnota atributu MQRFH2**
MQPD_COPY_DEFAULT	default Toto je výchozí hodnota. Je ekvivalentní zadání tří hodnot MQCOPY_FORWARD, MQCOPY_REPORT a MQCOPY_PUBLISH. Neuvádějte ji s žádnou jinou hodnotou.
MQPD_COPY_NONE	Není Neuvádějte ji s žádnou jinou hodnotou. Je-li použit s jinou hodnotou, má přednost.

Omezení pro složku < mq> MQRFH2 .

Je-li zpráva vložena do fronty, hledá se složka < mq>, aby bylo možné zprávu zpracovat podle jejích vlastností definovaných produktem MQ. Chcete-li povolit efektivní analýzu vlastností definovaných produktem MQ, platí pro danou složku následující omezení:

- Ve zprávě MQbudou provedeny pouze vlastnosti v první významné složce < mq> ve zprávě; vlastnosti ve všech ostatních složkách < mq> ve zprávě se ignorují.
- Je-li složka v souboru UTF-8, ve složce jsou povoleny pouze jednobajtové znaky UTF-8 . Vícebajtový znak ve složce může způsobit selhání syntaktické analýzy a zprávu, která má být odmítnuta.
- Ve složce < mq> nezahrnujte skupiny MQRFH2 . Přítomnost znaku Unicode U+003C v hodnotě vlastnosti způsobí, že zpráva bude odmítnuta.
- Nepoužívejte řídicí řetězce ve složce. S únikovým řetězcem se zachází jako se skutečnou hodnotou prvku.
- Pouze znak Unicode U+0020 je považován za bílý prostor ve složce. Všechny ostatní znaky jsou považovány za významné a mohou způsobit selhání syntaktické analýzy složky a zpráva, která má být odmítnuta.

Pokud selže syntaktická analýza složky < mq> nebo pokud složka tato omezení nepozoruje, bude zpráva odmítnuta s kódem CompCode **MQCC_FAILED** a s odůvodněním **MQRC_RFH_RESTRICTED_FORMAT_ERR**.

Záhlaví MQRFH2 , která nejsou platná

Při zpracování volání MQPUT, MQPUT1nebo MQGET se v rámci zprávy může objevit dílčí analýza všech záhlaví MQRFH2 , aby bylo možné zkontrolovat, které složky jsou zahrnuty, a určit, zda složky obsahují vlastnosti. Přehled záhlaví MQRFH2 , které nejsou platné.

Pokud dílčí analýza zprávy nemůže být úspěšně dokončena, protože struktura není platná, například pole `StrucLength` je příliš malé, pak:

- Volání MQPUT nebo MQPUT1 se nezdařilo s kódem příčiny MQRC_RFH_ERROR, pokud lze určit, že aplikace obsahuje některou volbu IBM WebSphere MQ 7 , takže existující aplikace se nezdaří.
- Volání funkce MQGET se vrátí úspěšně a ve vyrovnávací paměti, kterou jste zadali, bude vrácena hodnota MQRFH2 obsahující danou chybu.

Pokud dílčí analýza selže, protože nelze zjistit, zda určitá složka obsahuje vlastnosti, nebo ne, například složka začíná `<<jms`, takže syntaktická analýza selže před tím, než se zjistí název složky, pak:

- Volání MQPUT nebo MQPUT1 se nezdaří s kódem příčiny MQRC_RFH_FORMAT_ERROR, pokud lze určit, že aplikace obsahuje některou volbu IBM WebSphere MQ 7 , takže existující aplikace se nezdaří.
- Volání funkce MQGET se vrátí úspěšně a ve vyrovnávací paměti, kterou jste zadali, bude vrácena hodnota MQRFH2 obsahující danou chybu.

- Uvnitř správce front není zpráva odmítnuta kvůli špatně formátované složce, ale složka je vždy zpracovována, jako kdyby do ní nebyly obsaženy žádné vlastnosti.

Zpráva může procházet přes síť správce front se složkou obsahující takovou chybu syntaxe, ale nikdy ji nelze analyzovat a detekovat, zatímco jedna nebo více složek ve zprávě jsou:

- Platný
- Úspěšně analyzováno
- Používá se při zpracování zprávy

Detekce tedy není zaručena.

Pokud jedna z vašich aplikací používá produkt "MQSETMP-nastavení vlastnosti zprávy" na stránce 773nebo MQINQMP pro přístup k vlastnosti a tato akce způsobí, že složka MQRFH2 bude plně analyzována a zjistí chybu, která nemůže být dokončena, je to indikováno příslušným návratovým kódem pro volání rozhraní API. Ve složce nejsou k dispozici žádné vlastnosti pro aplikaci.

Je-li proveden pokus o plnou analýzu složky MQRFH2 a syntaktický analyzátor najde nerozpoznané atributy prvků nebo nerozpoznaný datový typ, analýza pokračuje úspěšně a je úspěšně dokončena bez jakýchkoli varování, nejedná se však o chybu analýzy.

Převod kódové stránky

Tento oddíl popisuje názvy kódových sad a identifikátory CCSID, národní jazyk, z/OS převod, IBM i převod, a kódování Unicode konverze.

V každé národní jazykové sekci jsou uvedeny následující informace:

- Podporované nativní CCSID
- Konverze kódových stránek, které nejsou podporovány

V informacích jsou použity následující termíny:

AIX
Označuje IBM MQ for AIX.

Linux
Označuje IBM MQ pro Linux for Intel a IBM MQ for Linux for 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 konverzi dat je pro konverzi, která se má provést na cílovém (přijímající) systému.

Pokud zdrojový produkt podporuje převod, lze kanál nastavit a data vyměněna nastavením atributu kanálu CONVERT na hodnotu YES na straně zdroje.

Poznámka:

1. Konverze pro IBM MQ MQI client informací se odehrává na serveru, takže server musí podporovat konverzi z CCSID klienta na CCSID serveru.
2. Převod může zahrnovat podporu přidanou CSD/PTF na nejnovější verzi produktu IBM MQ. Zkontrolujte obsah nejnovější úrovně služeb a zjistěte, zda je třeba instalovat CSD/PTF pro povolení tohoto převodu.
3. CCSID správce front IBM MQ musí být Mixed nebo SBCS.
4. Některé identifikátory CCSID, například 850 na systému AIX, které operační systém nepodporuje, mohou být stále používány aplikací a mohou být také nastaveny jako CCSID správce front IBM MQ .

Toto je povoleno pouze pro účely zpětné kompatibility a konverze selže, pokud příslušné převodní tabulky nejsou nainstalovány.

Prohlédněte si Tabulka 642 na stránce 930 pro křížový odkaz mezi některými čísly CCSID a některými názvy odvětvových kódových sad.

Související odkazy

“Národní jazyky” na stránce 930
Tyto informace obsahují jazyky podporované produktem IBM MQ.

Názvy kódových sad a CCSID

Názvy kódových sad a odpovídající identifikátory CCSID pro každý název kódové sady.

z/OS Produkt IBM MQ for z/OS poskytuje více převodů, než je uvedeno v tabulkách specifických jazyků. Úplný seznam převodů naleznete v tématu Tabulka 675 na stránce 956.

Tabulka 642. 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
ISO 8859-8	916
ISO 8859-9	920
ISO 8859-13	921
ISO 8859-15 (euro)	923
big5	950
eucJP	954 5050 33722
eucKR	970
eucTW	964
eucCN	1383
PCK	943
GBK	1386
koi8-r	878

Národní jazyky

Tyto informace obsahují jazyky podporované produktem IBM MQ.

Jazyky podporované produktem IBM MQ jsou:

- Americká angličtina-viz téma “americká angličtina” na stránce 931
- Němčina-viz téma “Němčina” na stránce 932
- Dánština a norština-viz téma “Dánština a nor” na stránce 932
- Finština a švédština-viz téma “Finnish a” na stránce 933

- Italština-viz téma “italština” na stránce 934
- Španělština-viz téma “Španělština” na stránce 935
- Britská angličtina/Gaelština-viz téma “Britská angličtina /gaelština” na stránce 936
- Francouzština-viz téma “Francouzština” na stránce 936
- Vícejazyčná-viz téma “Vícejazyčné” na stránce 937
- Portugalština-viz téma “Portugalština” na stránce 937
- Islandština-viz téma “Islandština” na stránce 938
- Východní evropské jazyky-viz téma “Jazyky východní Evropy” na stránce 939
- Cyrilice-viz téma “Cyrilice” na stránce 940
- Estonština-viz téma “Estonština” na stránce 941
- Lotyšština a litevština-viz téma “Lotyšské a litevské” na stránce 942
- Ukranian-viz téma “Ukrajinština” na stránce 943
- Řečtina-viz téma “řečtina” na stránce 944
- Turečtina-viz téma “Turečtina” na stránce 944
- Hebrejština-viz téma “Hebrejský” na stránce 945
- Farsi-viz téma “Perština” na stránce 947
- Urdu-viz téma “Urdština” na stránce 947
- Thajština-viz téma “Thajština” na stránce 948
- Laosky-viz téma “Laoština” na stránce 948
- Vietnamština-viz téma “Vietnamština” na stránce 949
- Japonština Latin SBCS-viz téma “Japonština Latin” na stránce 949
- Japonská Katakana SBCS-viz téma “Japonská Katakana SBCS” na stránce 950
- Japonština-Kanji/Latin Mixed-viz téma “Japonština-Kanji/Latin” na stránce 951
- Japonština-Kanji/Katakana Smíšený-viz téma “Smíšené červené, fialové a růžové květy” na stránce 952
- Korejština-viz téma “Korejština” na stránce 953
- Zjednodušená čínština-viz téma “Zjednodušená čínština” na stránce 954
- Tradiční čínština-viz téma “Tradiční čínština” na stránce 955

americká angličtina

Podrobnosti o CCSID a konverzi CCSID pro americkou angličtinu.

Tabulka 643. 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 platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

37
Nepřevádět 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 644. Nativní CCSID pro němčinu na podporovaných platformách

Platforma	**Nativní CCSID**
 IBM i  z/OS	273, 924, 1141
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

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 nor

Podrobnosti o CCSID a konverzi CCSID pro dánštinu a norštinu.

Tabulka 645. Nativní CCSID pro dánštinu a norštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	277, 924, 1142
AIX AIX	819, 923, 5348
Windows Windows	850, 858, 865, 1252, 5348
Linux Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných 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
Nepřevádí se na kódové stránky 924, 865, 1051, 1275

AIX

AIX

Kódová stránka:

819
Nepřevede na kódovou stránku 865

Windows

Windows

Kódová stránka:

865
Nepřevádět na kódové stránky 1051, 1275

Finnish a

Podrobnosti o CCSID a konverzi CCSID pro finštinu a švédštinu.

Tabulka 646. Nativní CCSID pro finštinu a švédštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	278, 924, 1143

Tabulka 646. Nativní CCSID pro finštinu a švédštinu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
AIX	819, 923, 5348
Windows	437, 850, 858, 865, 1252, 5348
Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

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 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
Nepřevádět na kódové stránky 1051, 1275

italština

Podrobnosti o CCSID a konverzi CCSID pro italštinu.

Tabulka 647. Nativní CCSID pro italštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	280, 924, 1144
AIX	819, 923, 5348

Tabulka 647. Nativní CCSID pro italštinu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
Windows Windows	437, 850, 858, 1252, 5348
Linux Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných 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 převodu CCSID pro španělštinu.

Tabulka 648. Nativní CCSID pro španělštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	284, 924, 1145
AIX AIX	819, 923, 5348
Windows Windows	437, 850, 858, 1252, 5348
Linux Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných 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

Britská angličtina /gaelština
Podrobnosti o CCSID a konverzi CCSID pro britskou angličtina/gaelštinu.

Tabulka 649. Nativní CCSID pro anglickou anglickou verzi/Gaelin 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 platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných 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 650. Nativní CCSID pro francouzštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	297, 924, 1147
AIX	819, 923, 5348
Windows	437, 850, 858, 1252, 5348
Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

297
Nepřevádí 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 651. Nativní CCSID pro vícejazyčný převod na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	500, 924, 1148
 AIX	819, 923, 5348
 Windows	437, 850, 858, 1252, 5348
 Linux	819, 923
klient Apple	1275

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

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 652. Nativní CCSID pro portugalštinu na podporovaných platformách

Platforma	Nativní CCSID
 IBM i	37, 500, 924, 1140

Tabulka 652. Nativní CCSID pro portugalštinu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
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 platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

37
Nepřevádí se na kódové stránky 858, 923, 1275

500
Nepřevádí se na kódové stránky 858, 923, 1275

924
Nepřevádí se na kódové stránky 858, 860, 1051, 1140, 1252, 1275, 5348

1140
Nepřevádí na kódové stránky 860, 924, 1051, 1275

Windows

Windows

Kódová stránka:

860
Nepřevádět na kódové stránky 1051, 1275

Islandština

Podrobnosti o CCSID a převodu CCSID pro islandštinu.

Tabulka 653. 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 platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

871
Nepřevádí 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

Kódová stránka:

861
Nepřevádět na kódové stránky 1051, 1275

Jazyky východní Evropy

Podrobnosti o CCSID a konverzi CCSID pro východní evropské jazyky. Mezi typické jazyky používající tyto CCSID patří albánština, chorvatština, čeština, maďarština, polština, rumunština, srbština, slovenština a slovinština.

Tabulka 654. Nativní CCSID pro východoevropské jazyky na podporovaných platformách

Platforma	**Nativní CCSID**
 IBM i  z/OS	870, 1153
 Windows	852, 1250, 5346, 9044
 AIX  Linux	912
Východoevropský klient Apple	1282
Rumunský klient Apple	1285
Chorvatský klient Apple	1284

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

870
Nepřevádět na kódové stránky 1284, 1285

1153
Nepřevádí se na kódové stránky 1250, 1284, 1285

IBM i

Kódová stránka:

870
Nepřevádí se na kódové stránky 1284, 1285, 5346, 9044

1153
Nepřevádět na kódové stránky 1282, 1284, 1285, 5346, 9044

, Linux

Kódová stránka:

912
Nepřevádět na kódové stránky 1284, 1285

Windows

Kódová stránka:

852
Nepřevádět na kódové stránky 1284, 1285

1250
Nepřevádět na kódové stránky 1284, 1285

9044
Nepřevádí se na kódové stránky 912, 1282, 1284, 1285

Cyrilice

Podrobnosti o CCSID a konverzi CCSID pro cyrilici. Typickými jazyky, které používají tyto CCSID, zahrnují Bělorusy, bulharštinu, makedonštinu, ruštinu a srbštinu.

Tabulka 655. Nativní CCSID pro cyrilici na podporovaných platformách

Platforma	**Nativní CCSID**
 z/OS	1025
 IBM i	880, 1025
 Windows	855, 866, 1131, 1251, 5347
 AIX  Linux	915
klient Apple	1283

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

880
Nepřevádí se na kódové stránky 855, 866, 878, 1131, 5347

1025
Nepřevádí se na kódové stránky 878, 5347

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 656. Nativní CCSID pro estonštinu na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	1122, 1157
 Windows	902, 922, 1257, 5353, 9449
 AIX  Linux	902, 922

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

1122
Nepřevádí se na kódové stránky 902, 1157, 9449

1157
Nepřevádí se na kódové stránky 922, 1122, 1257, 9449

IBM i

Kódová stránka:

1122
Nepřevádí se na kódové stránky 902, 5353, 9449

1157
Nepřevádí se na kódové stránky 922, 5353, 9449

Linux

Kódová stránka:

902
Nepřevádí na kódové stránky 922, 1122, 9449

922
Nepřevádí se na kódové stránky 902, 1157, 9449

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

Lotyšské a litevské

Podrobnosti o CCSID a konverze CCSID pro lotyštinu a litevštinu.

Tabulka 657. Nativní CCSID pro lotyština a litevština 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 svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

1112
Nepřevádí se na kódové stránky 901, 1156, 9449

1156
Nepřevádí se na kódové stránky 901, 1156, 9449

IBM i

Kódová stránka:

1112
Nepřevede na kódovou stránku 5353

1153
Nepřevádí na kódové stránky 921, 5353, 9449

Linux

Kódová stránka:

902
Nepřevádí na kódové stránky 921, 1112, 1257, 9449

921
Nepřevádí se na kódové stránky 901, 1156, 9449

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 ukrajinštinu.

Tabulka 658. Nativní CCSID pro Ukranian na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	1123
 Windows	1124, 1125, 1251, 5347
 AIX  Linux	1124

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

1123
Nepřevede na kódovou stránku 5347

Windows

Windows

Kódová stránka:

1125
Nepřevádí na kódovou stránku 1123

řečtina

Podrobnosti o CCSID a konverzi CCSID pro řečtinu.

Tabulka 659. Nativní CCSID pro řečtinu na podporovaných platformách

Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	875
Windows Windows	869, 1253, 5349
AIX AIX Linux Linux NCR	813
klient Apple	1280
Klient systému DOS	737

Všechny jiné než klientské platformy podporují převod mezi svými nativními CCSID, nativní 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 660. Nativní 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 platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

1026
Nepřevede na kódovou stránku 5350

Hebrejský

Podrobnosti o CCSID a převodu CCSID pro hebrejštinu.

Tabulka 661. Nativní CCSID pro hebrejštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
z/OS z/OS	424, 803, 4899, 12712
IBM i IBM i	424
AIX AIX	916, 9048
Windows Windows	1255, 5351
Linux Linux	916

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

424
Nepřevádí se na kódové stránky 867, 4899, 9048, 12712

803
Nepřevádí se na kódové stránky 867, 4899, 5351, 9048, 12712

4899
Nepřevádí 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 se na kódové stránky 867, 4899, 9048, 12712

9048
Nepřevádí na kódové stránky 424, 803, 856, 862, 916, 1255

Windows

Kódová stránka:

1255
Nepřevádí se na kódové stránky 867, 4899, 9048, 12712

5351
Nepřevede na kódovou stránku 803

Arabština

Podrobnosti o CCSID a konverzi CCSID pro arabštinu

Tabulka 662. Nativní CCSID pro arabštinu na podporovaných platformách

Platforma	Nativní CCSID
IBM i z/OS	420
AIX	1046, 1089
	1089 (viz poznámka)
Windows	720, 864, 1256, 5352
Linux	1089

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

420
Nepřevede na kódovou stránku 5352

Linux Tru64

Kódová stránka:

1089
Nepřevede na kódovou stránku 720

Windows

Kódová stránka:

720
Nepřevádí na kódové stránky 1089, 5352

5352
Nepřevede na kódovou stránku 720

Perština

Podrobnosti o CCSID a konverzi CCSID pro Farsi.

Tabulka 663. Nativní CCSID pro Farsi 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 svými nativními CCSID a nativními CCSID jiných platforem.

Urdština

Podrobnosti o CCSID a konverzi CCSID pro Urdu.

Tabulka 664. Nativní CCSID pro Urdu na podporovaných platformách

Platforma	Nativní CCSID
 IBM i  z/OS	918
 Windows	868

Tabulka 664. Nativní CCSID pro Urdu na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
AIX AIX Linux Linux	1006

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

918
Nepřevede na kódovou stránku 1006

Thajština

Podrobnosti o CCSID a převodu CCSID pro thajštinu.

Tabulka 665. Nativní CCSID pro thajštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	838
AIX AIX Linux Linux Windows Windows	874 (viz poznámka)

Poznámka: Nativní CCSID pro tyto platformy nebyl standardizován a může se změnit.

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem.

Laoština

Podrobnosti o CCSID a převodu CCSID pro Lao.

Tabulka 666. Nativní CCSID pro Lao na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i IBM i z/OS z/OS	1132
AIX AIX Linux Linux Windows Windows	1133

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem.

Vietnamština

Podrobnosti o CCSID a převodu CCSID pro vietnamštinu.

Tabulka 667. Nativní 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 svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

IBM i

Kódová stránka:

1130
Nepřevádět na kódové stránky 1129, 5354

Japonština Latin

Podrobnosti o CCSID a konverzi CCSID pro japonštinu Latin SBCS.

Tabulka 668. Nativní CCSID pro japonské Latin 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. AIX 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na AIX. CCSID hlášený operačním systémem je 33722.
2. Windows Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Toto CCSID však ne všechny platformy IBM MQ podporují.

   Na IBM MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna do souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943.

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

1027
Nepřevádí se na kódové stránky 932, 942, 943, 954, 5050, 33722

IBM i

IBM i

Kódová stránka:

1027
Nepřevede na kódovou stránku 932

AIX

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 japonštinu-katakana SBCS.

Tabulka 669. 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 AIX. CCSID hlášený operačním systémem je 33722.
2. Windows Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Toto CCSID však ne všechny platformy IBM MQ podporují.

   Na IBM MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna do souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943.
3. Kromě předchozích převodů podporuje IBM MQ konverzi 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 svými nativními CCSID a nativními 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

IBM i

Kódová stránka:

290
Nepřevede na kódovou stránku 932

AIX

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

Linux

Kódová stránka:

943
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

Japonština-Kanji/Latin

Podrobnosti o CCSID a konverzi CCSID pro japonštinu Kanji/Latin Mixed.

Tabulka 670. Nativní CCSID pro japonštinu Kanji/Latin Mixed 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. IBM i z/OS 5035 je CCSID související s kódovou stránkou 939
2. AIX 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na AIX. CCSID hlášený operačním systémem je 33722.
3. Windows Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Toto CCSID však ne všechny platformy IBM MQ podporují.

   Na IBM MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna do souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943.

Všechny platformy podporují převod mezi svými nativními CCSID a nativními 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 671. Nativní CCSID pro japonštinu Kanji/Katakana Smíšený na podporovaných platformách	
Platforma	**Nativní CCSID**
z/OS	1390, 5026 (viz poznámka 1)

Tabulka 671. Nativní CCSID pro japonštinu Kanji/Katakana Smíšený na podporovaných platformách (pokračování)	
Platforma	**Nativní CCSID**
IBM i	5026 (viz poznámka 1)
AIX	932, 5050, 33722 (viz poznámka 2)
Windows	932, 943 (viz poznámka 4)
Linux	943, 5050

Poznámka:

1. IBM i z/OS Jednobajtový režim CCSID 1390 a 5026 v EBCDIC obsahuje malá písmena v různých místech typického/invariantního rozvržení pro základní latinku a je třeba dbát na to, aby při převodu dat zprávy na jiné CCSID nedošlo ke ztrátě dat. Kromě toho může použití těchto identifikátorů CCSID jako výchozího identifikátoru CCSID správce front způsobit problémy při komunikaci s jinými správci front, například názvy kanálů používající malá písmena nemusí být ve 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 672. Nativní CCSID pro korejštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
IBM i z/OS	933, 1364
AIX Linux	970
Windows	949, 1363

Všechny platformy podporují převod mezi svými nativními CCSID a nativními 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 673. Nativní CCSID pro zjednodušenou čínštinu na podporovaných platformách	
Platforma	**Nativní CCSID**
z/OS	935, 1388
IBM i	935, 1388
AIX	1383, 1386
Windows	1381, 1386 (viz poznámka 2)
Linux	1383

Poznámka:

1. Windows Produkt Windows používá kódovou stránku 936, ale je nejlépe reprezentován identifikátorem CCSID 1386. Toto CCSID však ne všechny platformy IBM MQ podporují.

   Na IBM MQ for Windows CCSID 1381 se používá ke znázornění kódové stránky 936, ale změna do souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 1386.
2. Produkt IBM MQ podporuje standard čínských GB18030 .

   Windows Linux z/OS V systémech z/OS, Windows a Linuxje podpora konverze poskytována mezi Unicode (UTF-8 a UTF-16) a CCSID 1388 (EBCDIC s příponou GB18030), Unicode (UTF-8 a UTF-16) a CCSID 5488 (GB18030), a mezi CCSID 1388 a CCSID 5488.

   Poznámka:

IBM i V systému IBM ije podpora poskytována operačním systémem pro převod mezi Unicode (UTF-8 a UTF-16) a CCSID 1388 (EBCDIC s příponou GB18030).

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

935
Nepřevede na kódovou stránku 1383

1388
Nepřevede na kódovou stránku 1383

Tradiční čínština

Podrobnosti o CCSID a konverzi CCSID pro tradiční čínštinu.

Tabulka 674. Nativní 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 svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

z/OS

Kódová stránka:

937
Nepřevede na kódovou stránku 964

1388
Nepřevede na kódovou stránku 1383

Linux

Linux

Kódová stránka:

964
Nepřevede na kódovou stránku 938

z/OS Podpora konverze produktu z/OS

Seznam podporovaných převodů CCSID.

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID	
CCSID	**Převádí na a z CCSID.**
37	256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
256	37, 273, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 819, 833, 836, 838, 850, 852, 857, 860-866, 869-871, 875, 880, 905, 1025-1027, 1112, 1122, 1200, 1208, 1251-1252, 1275, 4386, 4929, 4932, 4934, 4946, 4948, 4953, 4960, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 13121, 13488, 16804, 17248, 17584, 28709
259	437, 808, 850-852, 855-858, 860-865, 867, 869, 872, 874, 899, 901-902, 915, 1098, 1161-1162, 1200, 1208, 1250-1258, 4946, 4948, 4951-4953, 4960, 4970, 5346, 5348, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584
273	37, 256, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1250, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
274	500, 1047
275	37, 437, 500, 819, 850, 1047, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
277	37, 256, 273, 278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
278	37, 256, 273, 277, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
280	37, 256, 273, 277-278, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
281	1047
282	500, 1047, 1200, 1208, 13488, 17584
284	37, 256, 273, 277-278, 280, 285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
285	37, 256, 273, 277-278, 280, 284, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
290	37, 256, 273, 277-278, 280, 284-285, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
293	1200, 1208, 13488, 17584
297	37, 256, 273, 277-278, 280, 284-285, 290, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1100, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
300	301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
420	37, 256, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709
423	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 737, 775, 813, 819, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
424	37, 256, 420, 437, 500, 737, 775, 803, 819, 836, 850, 852, 856-857, 860-865, 916, 1112, 1122, 1200, 1208, 1252, 1255, 4932, 4946, 4948, 4952-4953, 4960, 5012, 5351, 8229, 8612, 9044, 9049, 9056, 13488, 16804, 17248, 17584, 28709
437	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-863, 865-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1040-1043, 1047, 1051, 1097, 1098, 1114-1115, 1126, 1140-1149, 1200, 1208, 1252, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210-5211, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
500	37, 256, 273-275, 277-278, 280, 282, 284-285, 290, 297, 367, 420, 423-424, 437, 737, 775, 813, 819, 833, 836, 838, 850-852, 855-858, 860-866, 869-871, 874-875, 880, 891, 895, 897, 903-905, 912, 914-916, 920-924, 1004, 1009-1021, 1023, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097, 1100-1107, 1112, 1114-1115, 1122, 1124-1126, 1129-1133, 1137, 1140-1149, 1200, 1208, 1250-1258, 1275, 1280-1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5142, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 9238, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664, 28709
720	37, 420, 864, 1200, 1208, 1256, 4960, 8229, 8612, 9056, 13488, 16804, 17248, 17584, 28709
737	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 833, 836, 838, 850, 869-871, 875, 880, 905, 1025-1027, 1097, 1200, 1208, 1252-1253, 1280, 4386, 4909, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 9061, 13121, 13488, 16804, 17584, 28709
775	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 833, 836, 838, 850, 870-871, 875, 880, 905, 1025-1027, 1097, 1112, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
813	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
819	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 803, 813, 833, 836, 838, 850, 852, 855, 857-858, 860-861, 863-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1041-1043, 1047, 1051, 1088-1089, 1097, 1098, 1112, 1114, 1122-1123, 1126, 1130, 1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
833	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25617, 25619, 25664, 28709
834	926, 951, 1200, 1208, 1362, 4930, 9026, 13488, 17584
835	927, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
836	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25479, 25617, 25619, 25664, 28709
837	928, 1200, 1208, 1380, 1385, 4933, 13488, 17584
838	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
848	924, 1148, 1158, 1200, 1208, 5347, 13488, 17584
849	924, 1148, 1154, 1200, 1208, 5347, 13488, 17584
850	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097, 1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
851	259, 423, 500, 875, 1200, 1208, 4971, 13488, 17584

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
852	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
855	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
856	259, 273, 424, 500, 803, 850, 862, 916, 1200, 1208, 1255, 4946, 4952, 5012, 5351, 13488, 17584
857	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
858	37, 259, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 860-861, 865, 871-872, 901-902, 923-924, 1047, 1051, 1140-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
859	808, 872, 901-902, 1153-1157, 1160-1162, 1164, 1200, 1208, 13488, 17584
860	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 861, 863, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1140, 1145-1146, 1148, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
861	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857-858, 860, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 923-924, 1025-1027, 1041-1043, 1097, 1148, 1149, 1200, 1208, 1252, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
862	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 803, 833, 838, 850, 856, 870-871, 875, 880, 905, 916, 1025-1027, 1097, 1200, 1208, 1252, 1255, 4386, 4929, 4934, 4946, 4952, 4971, 5012, 5123, 5351, 8229, 8482, 8612, 9025, 9030, 12712, 13121, 13488, 16804, 17584, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
863	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 838, 850, 852, 857, 860-861, 865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1041-1043, 1051, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 28709
864	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709
865	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 819, 833, 838, 850, 858, 860, 863, 870-871, 875, 880, 905, 923-924, 1025-1027, 1097, 1142-1143, 1148, 1200, 1208, 1252, 4386, 4929, 4934, 4946, 4971, 5123, 5348, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
866	37, 256, 437, 500, 819, 850, 855, 870, 878, 880, 915, 1025, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
867	259, 1153-1155, 1160, 1200, 1208, 4899, 5351, 9048, 12712, 13488, 17584
868	918, 1006, 1200, 1208, 13488, 17584
869	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 870-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
870	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869, 871, 874-875, 880, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
871	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-865, 869, 870, 874-875, 880, 897, 903, 912, 916, 920, 923-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1112, 1122, 1140-1149, 1200, 1208, 1252, 1275, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5348, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
872	259, 808, 858-859, 923-924, 1140-1149, 1153-1155, 1200, 1208, 5347, 5348, 13488, 17584

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
874	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
875	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
878	855, 866, 880, 915, 1025, 1131, 1200, 1208, 1251, 1283, 4951, 5347, 13488, 17584
880	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 897, 903, 912, 915-916, 920, 1009, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1251-1252, 1283, 4909, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
891	500, 833, 1088, 1200, 1208, 4929, 9025, 13121, 13488, 17584, 25664
895	290, 500, 1027, 1041, 1200, 1208, 4386, 5123, 8482, 13488, 17584, 25617
896	290, 1027, 1041, 1200, 1208, 4386, 4992, 5123, 8482, 13488, 17584, 25617
897	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
899	259
901	259, 858-859, 902, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
902	259, 858-859, 901, 923-924, 1140, 1148, 1156-1157, 1200, 1208, 5348, 5353, 13488, 17584
903	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1200, 1208, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
904	37, 500, 1114, 1200, 1208, 5210, 8229, 13488, 17584, 25480, 28709
905	37, 256, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 920, 1026, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na 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, 4953, 4970-4971, 5012, 5123, 5346, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
914	37, 437, 500, 819, 850, 1200, 1208, 1252, 1257, 4946, 8229, 13488, 17584, 28709
915	37, 259, 437, 500, 819, 850, 855, 866, 870, 878, 880, 1025, 1131, 1200, 1208, 1251-1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
916	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5012, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
918	864, 868, 1006, 1200, 1208, 4960, 9056, 13488, 17248, 17584
920	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 1025-1026, 1200, 1208, 1252, 1254, 1281, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5350, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 28709
921	37, 437, 500, 819, 850, 922, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
922	37, 437, 500, 819, 850, 921, 1112, 1122, 1200, 1208, 1252, 1257, 4946, 5353, 8229, 13488, 17584, 28709
923	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 865, 871-872, 901-902, 924, 1047, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
924	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 865, 871-872, 901-902, 923, 1047, 1051, 1140-1149, 1153-1157, 1160-1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
926	834, 951, 9026
927	835, 947, 1200, 1208, 4931, 9027, 13488, 17584, 21427
928	837, 1200, 1208, 1380, 13488, 17584
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
933	934, 944, 949, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
934	933, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25510, 25525, 29621, 33717, 37813
935	936, 946, 1200, 1208, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
936	935, 946, 1381, 5031, 5477, 5484, 9127, 13223, 25512
937	938, 948, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
938	937, 950, 1370, 5033, 5046, 9142, 25514
939	930-932, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
941	300-301, 1200, 1208, 1351, 4396, 8492, 13488, 16684, 17584
942	930-932, 939, 943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
943	930-932, 939, 942, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
944	933, 949, 1200, 1208, 5029, 5045, 5460, 9125, 13221, 13488, 17317, 17584, 25520, 25525, 29616, 29621, 33717, 37813
946	935-936, 1200, 1208, 5031, 5484, 9127, 13223, 13488, 17584, 25512
947	835, 927, 1200, 1208, 4931, 9027, 13488, 17584, 21427
948	937, 950, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25524, 29620
949	933-934, 944, 1200, 1208, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25510, 25520, 25525, 29616, 29621, 33717, 37813
950	937-938, 948, 1200, 1208, 1370, 5033, 5046, 9142, 13488, 17584, 25514, 25524, 29620
951	834, 926, 1200, 1208, 1362, 4930, 9026, 13488, 17584
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1011	500, 1200, 1208, 13488, 17584
1012	500, 1200, 1208, 13488, 17584
1013	500, 1140, 1200, 1208, 13488, 17584
1014	500, 1200, 1208, 13488, 17584
1015	500, 1200, 1208, 13488, 17584
1016	500, 1200, 1208, 13488, 17584
1017	500, 1200, 1208, 13488, 17584
1018	500, 1200, 1208, 13488, 17584
1019	500, 1200, 1208, 13488, 17584
1020	500
1021	500
1023	500
1025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-866, 869-871, 874-875, 878, 880, 897, 903, 912, 915-916, 920, 1009, 1026-1027, 1040-1043, 1051, 1088, 1112, 1122, 1131, 1200, 1208, 1251-1252, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5347, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1026	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1009, 1025, 1027, 1040-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5350, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1027	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-865, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1026, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1040	37, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 833, 836, 850, 852, 855, 857, 870-871, 1025-1027, 1041-1043, 1088, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1041	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040, 1042-1043, 1088, 1200, 1208, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1042	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1043, 1088, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1043	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040, 1041, 1042, 1088, 1114, 1200, 1208, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 17584, 25473, 25479, 25617, 25619, 25664, 28709
1046	420, 500, 864, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1047	37, 273-275, 277-278, 280, 281, 282, 284-285, 290, 297, 437, 500, 819, 850, 852, 858, 870-871, 875, 912, 923-924, 1026-1027, 1140-1149, 1200, 1208, 1252, 1254, 4946, 4948, 5123, 8229, 8482, 13488, 17584, 28709
1051	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1025, 1097, 1140-1149, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1088	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1126, 1200, 1208, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
1089	420, 500, 819, 850, 864, 1046, 1127, 1200, 1208, 1256, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1097	37, 437, 500, 737, 775, 819, 850, 852, 857, 860-865, 1051, 1098, 1112, 1122, 1200, 1208, 1252, 4946, 4948, 4953, 4960, 8229, 9044, 9049, 9056, 13488, 17248, 17584, 28709
1098	259, 420, 437, 819, 850, 1097, 1200, 1208, 1252, 4946, 8612, 13488, 16804, 17584
1100	37, 273, 277-278, 280, 284-285, 297, 500, 850, 4946, 8229, 28709
1101	500
1102	500
1103	500
1104	500
1105	500
1106	500
1107	500

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1112	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1122, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1114	37, 437, 500, 819, 836, 850, 904, 1043, 1115, 1200, 1208, 4932, 4946, 5210-5211, 8229, 13488, 17584, 25480, 25619, 28709
1115	37, 367, 437, 500, 836, 903, 1114, 1200, 1208, 4932, 5210-5211, 8229, 13488, 17584, 25479, 28709
1122	37, 256, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 775, 819, 833, 836, 838, 850, 870-871, 875, 880, 905, 921-922, 1025-1027, 1097, 1112, 1200, 1208, 1252, 1257, 4386, 4929, 4932, 4934, 4946, 4971, 5123, 5353, 8229, 8482, 8612, 9025, 9030, 13121, 13488, 16804, 17584, 28709
1123	819, 1124-1125, 1148, 1200, 1208, 1251-1252, 1283, 5347, 13488, 17584
1124	37, 500, 1123, 1125, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1125	500, 1123, 1124, 1200, 1208, 1251, 1283, 5347, 13488, 17584
1126	37, 367, 437, 500, 819, 833, 850, 1088, 1200, 1208, 1252, 4929, 4946, 8229, 9025, 13121, 13488, 17584, 25664, 28709
1127	420, 864, 1046, 1089, 1256, 4960, 5142, 8612, 9056, 9238, 16804, 17248
1129	500, 1130, 1200, 1208, 1258, 5354, 13488, 17584
1130	37, 500, 819, 850, 1129, 1200, 1208, 1252, 1258, 4946, 5354, 8229, 13488, 17584, 28709
1131	37, 500, 878, 915, 1025, 1200, 1208, 1251, 1283, 5347, 8229, 13488, 17584, 28709
1132	37, 500, 819, 850, 1133, 1200, 1208, 1252, 4946, 8229, 13488, 17584, 28709
1133	500, 1132, 1200, 1208, 13488, 17584
1137	37, 500, 819, 1200, 1208, 8229, 13488, 17584, 28709
1139	290, 1027, 4386, 5123, 8482
1140	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860, 863, 871-872, 901-902, 923-924, 1013, 1047, 1051, 1141-1149, 1153-1157, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1142	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1141, 1143-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1143	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 865, 871-872, 923-924, 1047, 1051, 1140-1142, 1144-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1144	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1143, 1145-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1145	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1144, 1146-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1146	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 860, 863, 871-872, 923-924, 1047, 1051, 1140-1145, 1147-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1147	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871-872, 923-924, 1047, 1051, 1140-1146, 1148-1149, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1148	37, 273, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 848-850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1047, 1051, 1123, 1140-1147, 1149, 1153-1164, 1200, 1208, 1252, 1275, 4899, 4946, 5348, 5349, 8229, 12712, 13488, 17584, 28709
1149	37, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 861, 863, 871-872, 923-924, 1047, 1051, 1140-1148, 1153-1157, 1160-1162, 1200, 1208, 1252, 1275, 4946, 5348, 8229, 13488, 17584, 28709
1153	808, 858-859, 867, 872, 923-924, 1140-1149, 1154-1157, 1160-1162, 1200, 1208, 5348, 9044, 13488, 17584
1154	808, 849, 858-859, 867, 872, 923-924, 1140-1149, 1153, 1155-1157, 1160-1162, 1200, 1208, 5347, 5348, 13488, 17584
1155	858-859, 867, 872, 923-924, 1140-1149, 1153-1154, 1156-1157, 1160-1162, 1200, 1208, 5348, 5350, 9049, 13488, 17584
1156	858-859, 901-902, 923-924, 1140-1149, 1153-1155, 1157, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1157	858-859, 901-902, 923-924, 1140-1149, 1153-1156, 1160, 1200, 1208, 5348, 5353, 12712, 13488, 17584
1158	848, 923, 1148, 1200, 1208, 5347, 5348, 13488, 17584
1159	1148, 1200, 1208, 13488, 17584
1160	858-859, 867, 923-924, 1140-1149, 1153-1157, 1161-1162, 1200, 1208, 5348, 13488, 17584

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1161	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1162	259, 858-859, 923-924, 1140-1149, 1153-1155, 1160, 5348, 17584
1163	924, 1148, 1164, 5354, 17584
1164	858-859, 923-924, 1140, 1148, 1163, 1200, 1208, 5348, 5354, 13488, 17584
1166	1200,1208,13488,17584
1200	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1208	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1374-1379, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5026, 5035, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 17584, 21427, 28709
1250	37, 259, 273, 500, 819, 850, 852, 855, 870, 912, 1200, 1208, 1252, 1282, 4946, 4948, 4951, 5346, 8229, 9044, 13488, 17584, 28709
1251	37, 256, 259, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1252, 1283, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1252	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1025-1027, 1041, 1047, 1051, 1097-1098, 1112, 1122-1123, 1126, 1130, 1132, 1140-1149, 1200, 1208, 1250-1251, 1254-1255, 1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25617, 28709
1253	37, 259, 423, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1280, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1254	37, 259, 500, 819, 850, 857, 869, 905, 920, 1026, 1047, 1200, 1208, 1252, 1281, 4946, 4953, 5350, 8229, 9049, 9061, 13488, 17584, 28709
1255	37, 259, 424, 500, 803, 819, 850, 856, 862, 916, 1200, 1208, 1252, 1281, 4946, 4952, 5012, 5351, 8229, 13488, 17584, 28709
1256	259, 420, 500, 720, 850, 864, 1046, 1089, 1127, 1200, 1208, 4946, 4960, 5142, 5352, 8612, 9056, 9238, 13488, 16804, 17248, 17584
1257	37, 259, 437, 500, 775, 819, 850, 914, 921-922, 1112, 1122, 1200, 1208, 1252, 4946, 5353, 8229, 13488, 17584, 28709
1258	37, 259, 500, 819, 1129-1130, 1200, 1208, 5354, 8229, 13488, 17584, 28709
1275	37, 256, 273, 277-278, 280, 284-285, 297, 437, 500, 819, 850, 858, 863, 871, 923-924, 1051, 1140-1149, 1200, 1208, 1252, 4946, 5348, 8229, 13488, 17584, 28709
1276	1200, 1208, 13488, 17584
1277	1200, 1208, 13488, 17584
1280	37, 423, 437, 500, 737, 813, 819, 850, 869, 875, 1200, 1208, 1252-1253, 4909, 4946, 4971, 5349, 8229, 9061, 13488, 17584, 28709
1281	37, 437, 500, 819, 850, 857, 905, 920, 1026, 1200, 1208, 1252, 1254-1255, 4946, 4953, 5350, 8229, 9049, 13488, 17584, 28709
1282	500, 852, 870, 912, 1200, 1208, 1250, 4948, 5346, 9044, 13488, 17584
1283	37, 437, 500, 819, 850, 855, 866, 878, 880, 915, 1025, 1123-1125, 1131, 1200, 1208, 1251-1252, 4946, 4951, 5347, 8229, 13488, 17584, 28709
1284	1200, 1208, 13488, 17584
1285	1200, 1208, 13488, 17584
1351	300-301, 941, 1200, 1208, 4396, 8492, 13488, 16684, 17584
1362	834, 951, 1200, 1208, 4930, 9026, 13488, 17584
1363	933, 949, 1200, 1208, 1364, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
1364	933, 949, 1200, 1208, 1363, 5029, 5045, 5460, 9125, 9555, 13221, 13488, 13651, 17317, 17584, 25525, 29621, 33717, 37813
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
1380	837, 928, 1200, 1208, 1385, 4933, 13488, 17584
1381	935-936, 1200, 1208, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584, 25512
1385	837, 1200, 1208, 1380, 4933, 13488, 17584
1386	935, 1200, 1208, 1381, 1388, 5031, 5477, 5482, 5484, 9127, 13223, 13488, 17584
1388	935, 1200, 1208, 1381, 1386, 5031, 5477, 5482, 5484, 5488, 9127, 13223, 13488, 17584
1390	930-932, 939, 942-943, 1200, 1208, 1399, 5026, 5028, 5035, 5038-5039, 5055, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
1399	930-932, 939, 942-943, 1200, 1208, 1390, 5026, 5028, 5035, 5038-5039, 5050, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
4386	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1139, 1252, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25473, 25617, 25619, 25664, 28709
4396	300-301, 941, 1351, 8492, 16684
4899	867, 1148, 1200, 1208, 5351, 9048, 12712, 13488, 17584
4909	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1253, 1280, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
4929	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
4930	834, 951, 1200, 1208, 1362, 9026, 13488, 17584
4931	835, 927, 947, 9027, 21427
4932	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 424, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 903, 1009, 1025-1027, 1040-1043, 1088, 1112, 1114-1115, 1122, 1252, 4386, 4929, 4946, 4948, 4951, 4953, 4971, 5123, 5210-5211, 8229, 8482, 9025, 9044, 9049, 13121, 25479, 25617, 25619, 25664, 28709
4933	837, 1200, 1208, 1380, 1385, 13488, 17584
4934	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1252, 4909, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 17248, 25473, 25479, 25617, 25619, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
4946	37, 256, 259, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 737, 775, 803, 813, 819, 833, 836, 838, 850, 852, 855-858, 860-866, 869-871, 874-875, 880, 897, 903, 905, 912, 914-916, 920-924, 1004, 1025-1027, 1040-1043, 1047, 1051, 1088-1089, 1097-1098, 1100, 1112, 1114, 1122, 1126, 1130, 1132, 1140-1149, 1250-1257, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4948, 4951-4953, 4960, 4970-4971, 5012, 5123, 5210, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25617, 25619, 25664, 28709
4948	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4951	37, 259, 273, 277-278, 280, 284-285, 290, 297, 437, 500, 819, 833, 836, 850, 852, 855, 857, 866, 870-871, 878, 880, 912, 915, 1025-1027, 1040-1043, 1088, 1200, 1208, 1250-1252, 1283, 4386, 4929, 4932, 4946, 4948, 4953, 5123, 5346, 5347, 8229, 8482, 9025, 9044, 9049, 13121, 13488, 17584, 25617, 25619, 25664, 28709
4952	259, 273, 424, 500, 803, 850, 856, 862, 916, 1200, 1208, 1255, 4946, 5012, 5351, 13488, 17584
4953	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 16804, 25473, 25479, 25617, 25619, 25664, 28709
4960	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17248, 17584, 28709
4970	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4909, 4934, 4946, 4948, 4953, 4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25479, 25617, 25619, 28709
4971	37, 256, 273, 277-278, 280, 284-285, 297, 367, 423, 437, 500, 737, 775, 813, 819, 836, 838, 850-852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1009, 1025-1027, 1041-1043, 1047, 1088, 1112, 1122, 1200, 1208, 1252-1253, 1280, 4909, 4932, 4934, 4946, 4948, 4953, 4960, 4970, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 25664, 28709
4992	290, 896, 1027, 1041, 4386, 5123, 8482, 25617

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
5012	37, 273, 277-278, 280, 284-285, 297, 423-424, 437, 500, 803, 813, 819, 838, 850, 852, 856-857, 860-863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 1255, 4909, 4934, 4946, 4948, 4952-4953, 4970-4971, 5123, 5351, 8229, 9030, 9044, 9049, 9061, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
5026	930-932, 939, 942-943, 1390, 1399, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5028	930-932, 939, 942-943, 1390, 1399, 5026, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5029	933-934, 944, 949, 1363-1364, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5031	935-936, 946, 1381, 1386, 1388, 5477, 5482, 5484, 9127, 13223, 25512
5033	937-938, 948, 950, 1370, 5046, 9142, 25514, 25524, 29620
5035	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5038-5039, 9122, 9124, 9131, 9135, 1208, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5038	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
5039	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038, 9122, 9124, 9131, 9135, 13218-13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
5045	933-934, 944, 949, 1363-1364, 5029, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5046	937-938, 948, 950, 1370, 5033, 9142, 25514, 25524, 29620
5104	420, 864, 1008, 1200, 1208, 4960, 8612, 9056, 13488, 16804, 17248, 17584
5123	290, 367, 423, 437, 819, 1027, 1041, 1047, 1140-1149, 1156, 1157, 1160, 1200, 1208, 1252, 4948, 5348, 8482, 13488
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

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
5348	37, 259, 273, 275, 277-278, 280, 284-285, 297, 437, 500, 808, 819, 850, 858, 860-861, 863, 865, 871-872, 901-902, 923-924, 1051, 1140-1149, 1153-1158, 1160-1162, 1164, 1200, 1208, 1252, 1275, 4946, 8229, 13488, 17584, 28709
5349	813, 869, 875, 1148, 1200, 1208, 1253, 1280, 4909, 4971, 9061, 13488, 17584
5350	857, 920, 1026, 1155, 1200, 1208, 1254, 1281, 4953, 9049, 13488, 17584
5351	424, 856, 862, 867, 916, 1200, 1208, 1255, 4899, 4952, 5012, 9048, 12712, 13488, 17584
5352	420, 864, 1046, 1089, 1200, 1208, 1256, 4960, 5142, 8612, 9056, 9238, 13488, 16804, 17248, 17584
5353	901-902, 921-922, 1112, 1122, 1156-1157, 1200, 1208, 1257, 13488, 17584
5354	1129-1130, 1163, 1164, 1200, 1208, 1258, 13488, 17584
5460	933-934, 944, 949, 1363-1364, 5029, 5045, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
5477	935-936, 1381, 1386, 1388, 5031, 5482, 5484, 9127, 13223, 25512
5482	935, 1381, 1386, 1388, 5031, 5477, 5484, 9127, 13223
5484	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 9127, 13223, 25512
5488	1388
8229	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 16804, 17248, 25473, 25479, 25480, 25617, 25619, 25664, 28709
8482	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 895-897, 1009, 1025-1027, 1040-1043, 1047, 1088, 1112, 1122, 1139, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 4992, 5123, 8229, 9025, 9044, 9049, 9056, 13121, 13488, 17248, 17584, 25473, 25617, 25619, 25664, 28709
8492	300-301, 941, 1351, 4396, 16684
8612	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 9044, 9049, 9056, 9238, 13488, 16804, 17248, 17584, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
9025	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9044, 9049, 9056, 13121, 17248, 25617, 25619, 25664, 28709
9026	834, 926, 951, 1362, 4930
9027	835, 927, 947, 1200, 1208, 4931, 13488, 17584, 21427
9030	37, 256, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 775, 813, 819, 838, 850, 852, 857, 860-865, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1112, 1122, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4960, 4970-4971, 5012, 5123, 8229, 9044, 9049, 9056, 9061, 9066, 13488, 17248, 17584, 25473, 25479, 25617, 25619, 28709
9044	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1047, 1088, 1097, 1153, 1200, 1208, 1250, 1252, 1282, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5346, 8229, 8482, 8612, 9025, 9030, 9049, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9048	867, 1200, 1208, 4899, 5351, 12712, 13488, 17584
9049	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 905, 912, 916, 920, 1025-1027, 1040-1043, 1088, 1097, 1155, 1200, 1208, 1252, 1254, 1281, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5350, 8229, 8482, 8612, 9025, 9030, 9044, 9061, 9066, 13121, 13488, 16804, 17584, 25473, 25479, 25617, 25619, 25664, 28709
9056	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9238, 13121, 13488, 16804, 17248, 17584, 28709
9061	37, 256, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 737, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252-1254, 1280, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5349, 8229, 9030, 9044, 9049, 9066, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9066	37, 259, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1200, 1208, 1252, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 9030, 9044, 9049, 9061, 13488, 17584, 25473, 25479, 25617, 25619, 28709
9122	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
9124	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9125	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
9127	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 13223, 25512
9131	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9135	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
9142	937-938, 948, 950, 1370, 5033, 5046, 25514, 25524, 29620
9238	420, 500, 864, 1046, 1089, 1127, 1200, 1208, 1256, 4960, 5142, 5352, 8612, 9056, 13488, 16804, 17248, 17584
9555	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 13221, 13651, 17317, 25525, 29621, 33717, 37813
12712	862, 867, 1148, 1156-1157, 1200, 1208, 4899, 5351, 9048, 13488, 17584
13121	37, 256, 273, 277-278, 280, 284-285, 290, 297, 367, 437, 500, 737, 775, 819, 833, 836, 850, 852, 855, 857, 860-865, 870-871, 891, 1009, 1025-1027, 1040-1043, 1088, 1112, 1122, 1126, 1200, 1208, 1252, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4960, 5123, 8229, 8482, 9025, 9044, 9049, 9056, 13488, 17248, 17584, 25617, 25619, 25664, 28709
13218	930-932, 939, 942-943, 1200, 1208, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13219, 13231, 13488, 17314, 17584, 25508, 25518, 29614, 33698-33700, 37796
13219	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218, 13231, 17314, 25508, 25518, 29614, 33698-33700, 37796
13221	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717, 37813
13223	935-936, 946, 1381, 1386, 1388, 5031, 5477, 5482, 5484, 9127, 25512
13231	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 17314, 25508, 25518, 29614, 33698-33700, 37796

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
13488	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 16684, 16804, 17248, 17584, 21427, 28709
13651	933, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 17317, 25525, 29621, 33717, 37813
16684	300-301, 941, 1200, 1208, 1351, 4396, 8492, 13488, 17584
16804	37, 256, 420, 424, 437, 500, 720, 737, 775, 819, 850, 852, 857, 860-865, 1008, 1046, 1089, 1098, 1112, 1122, 1127, 1200, 1208, 1252, 1256, 4946, 4948, 4953, 4960, 5104, 5142, 5352, 8229, 8612, 9044, 9049, 9056, 9238, 13488, 17248, 17584, 28709
17248	37, 256, 259, 273, 277-278, 280, 284-285, 290, 297, 420, 423-424, 500, 720, 819, 833, 838, 850, 864, 870-871, 875, 880, 905, 918, 1008, 1025-1027, 1046, 1089, 1097, 1127, 1200, 1208, 1252, 1256, 4386, 4929, 4934, 4946, 4960, 4971, 5104, 5123, 5142, 5352, 8229, 8482, 8612, 9025, 9030, 9056, 9238, 13121, 13488, 16804, 17584, 28709
17314	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 25508, 25518, 29614, 33698-33700, 37796
17317	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 25510, 25520, 25525, 29616, 29621, 33717, 37813
17584	37, 256, 259, 273, 275, 277-278, 280, 282, 284-285, 290, 293, 297, 300-301, 367, 420, 423-424, 437, 500, 720, 737, 775, 803, 806, 808, 813, 819, 833-838, 848-852, 855-872, 874-875, 878, 880, 891, 895-897, 901-905, 912, 914-916, 918, 920-924, 927-928, 930, 932-933, 935, 937, 939, 941-944, 946-951, 1004, 1006, 1008-1019, 1025-1027, 1040-1043, 1046-1047, 1051, 1088-1089, 1097-1098, 1112, 1114-1115, 1122-1126, 1129-1133, 1137, 1140-1149, 1153-1160, 1164, 1166, 1200, 1208, 1250-1258, 1275-1277, 1280-1285, 1351, 1362-1364, 1370-1371, 1380-1381, 1385-1386, 1388, 1390, 1399, 4899, 4909, 4930, 4933, 4948, 4951-4952, 4960, 4971, 5012, 5039, 5104, 5123, 5142, 5210, 5346-5354, 8482, 8612, 9027, 9030, 9044, 9048-9049, 9056, 9061, 9066, 9238, 12712, 13121, 13218, 13488, 16684, 16804, 17248, 21427, 28709
21427	835, 927, 947, 1200, 1208, 4931, 9027, 13488, 17584
25473	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1252, 4386, 4909, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 8229, 8482, 9030, 9044, 9049, 9061, 9066, 25479, 25617, 25619, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
25479	37, 273, 277-278, 280, 284-285, 297, 423, 437, 500, 813, 819, 836, 838, 850, 852, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 920, 1025-1027, 1041-1043, 1115, 1252, 4909, 4932, 4934, 4946, 4948, 4953, 4970-4971, 5012, 5123, 5211, 8229, 9030, 9044, 9049, 9061, 9066, 25473, 25617, 25619, 28709
25480	37, 500, 904, 1114, 5210, 8229, 28709
25508	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25518, 29614, 33698-33700, 37796
25510	933-934, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29621, 33717, 37813
25512	935-936, 946, 1381, 5031, 5477, 5484, 9127, 13223
25514	937-938, 950, 1370, 5033, 5046, 9142
25518	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 29614, 33698-33700, 37796
25520	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25525, 29616, 29621, 33717, 37813
25524	937, 948, 950, 1370, 5033, 5046, 9142, 29620
25525	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 29616, 29621, 33717, 37813
25617	37, 273, 277-278, 280, 284-285, 290, 297, 367, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 895-897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1252, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 4992, 5012, 5123, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25619, 25664, 28709
25619	37, 273, 277-278, 280, 284-285, 290, 297, 423, 437, 500, 813, 819, 833, 836, 838, 850, 852, 855, 857, 860-861, 863, 869-871, 874-875, 880, 897, 903, 912, 916, 1025-1027, 1040-1043, 1088, 1114, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4970-4971, 5012, 5123, 5210, 8229, 8482, 9025, 9030, 9044, 9049, 9061, 9066, 13121, 25473, 25479, 25617, 25664, 28709
25664	37, 273, 277-278, 280, 284-285, 290, 297, 367, 500, 819, 833, 836, 850, 852, 855, 857, 870-871, 875, 891, 1025-1027, 1040-1043, 1088, 1126, 4386, 4929, 4932, 4946, 4948, 4951, 4953, 4971, 5123, 8229, 8482, 9025, 9044, 9049, 13121, 25617, 25619, 28709

Tabulka 675. Podpora konverze IBM MQ for z/OS CCSID (pokračování)	
CCSID	**Převádí na a z CCSID.**
28709	37, 256, 273, 275, 277-278, 280, 284-285, 290, 297, 367, 420, 423-424, 437, 500, 720, 737, 775, 813, 819, 833, 836, 838, 850, 852, 855, 857-858, 860-866, 869-871, 874-875, 880, 897, 903-905, 912, 914-916, 920-924, 1009, 1025-1027, 1040-1043, 1047, 1051, 1088, 1097, 1100, 1112, 1114-1115, 1122, 1124, 1126, 1130-1132, 1137, 1140-1149, 1200, 1208, 1250-1255, 1257-1258, 1275, 1280-1281, 1283, 4386, 4909, 4929, 4932, 4934, 4946, 4948, 4951, 4953, 4960, 4970-4971, 5012, 5123, 5210-5211, 5346, 5348, 8229, 8482, 8612, 9025, 9030, 9044, 9049, 9056, 9061, 9066, 13121, 13488, 16804, 17248, 17584, 25473, 25479, 25480, 25617, 25619, 25664
29614	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 33698-33700, 37796
29616	933, 944, 949, 5029, 5045, 5460, 9125, 13221, 17317, 25520, 25525, 29621, 33717, 37813
29620	937, 948, 950, 1370, 5033, 5046, 9142, 25524
29621	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 33717, 37813
33698	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33699-33700, 37796
33699	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698, 33700, 37796
33700	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33699, 37796
33717	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 37813
37796	930-932, 939, 942-943, 1390, 1399, 5026, 5028, 5035, 5038-5039, 9122, 9124, 9131, 9135, 13218-13219, 13231, 17314, 25508, 25518, 29614, 33698-33700
37813	933-934, 944, 949, 1363-1364, 5029, 5045, 5460, 9125, 9555, 13221, 13651, 17317, 25510, 25520, 25525, 29616, 29621, 33717

IBM i Podpora konverze produktu IBM i

Úplný seznam identifikátorů CCSID a převodů podporovaných produktem IBM ilze nalézt v příslušné publikaci IBM i .

Podporované kódové stránky jsou uvedeny v seznamu Podporovaná mapování CCSID.

Podpora konverze Unicode

Některé platformy podporují převod uživatelských dat na kódování Unicode nebo z kódování Unicode. Podporovány jsou dvě formy kódování Unicode: UTF-16 (CCSID 1200, 13488 a 17584) a UTF-8 (CCSID 1208). CCSID 1200 nebo 1208 byste měli používat tak, jak představují nejnovější podporovanou verzi Unicode.

Náhradní páry UTF-16 (dvojice znaků UTF-16 v rozsahu X'D800'až do X'DFFF', které představují bod kódu Unicode nad U + FFFF), jsou podporovány. Pokud cílové CCSID neobsahuje mapování pro kódový bod představovaný dvojicí náhradních souborů UTF-16 , dvojice znaků se převede na jednotlivý zástupný znak.

Kombinování posloupností znaků je podporováno produktem IBM MQ. To znamená, že v některých případech se předem složený znak ve zdrojovém CCSID převede na kombinaci posloupnosti znaků v cílovém CCSID nebo v opačném směru.

Poznámka: Produkt IBM MQ nepodporuje identifikátory CCSID správce front UTF-16 , takže data záhlaví zprávy nemohou být zakódována v souboru UTF-16.

Podpora kódování Unicode produktu IBM MQ AIX

AIX

Při převodu na IBM MQ for AIX a z podporovaných Unicode CCSID (pokud možno 1200 nebo 1208) jsou podporovány CCSID, které nejsou 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, 938, 946, 970, 950, 946, 970, 950, 946, 946
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

Podpora IBM MQ for Windows a Linux pro Unicode

Windows Linux

U IBM MQ for Windows a IBM MQ pro konverzi Linux na a z podporovaných Unicode CCSID (pokud možno 1200 nebo 1208) jsou podporovány pro CCSID nepoužívající 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, 864, 865, 866, 867, 868, 869, 870, 871, 874, 875, 878, 880, 891, 897, 870, 870, 870, 870, 880, 870, 878, 870, 870, 870, 870, 870, 870, 870, 870, 868, 878, 870, 8
901, 902, 903, 904, 912, 913"5" na stránce 981, 915, 916, 918, 920, 921, 922, 923, 924, 927, 928, 930, 931"1" na stránce 981, 932"2" na stránce 981, 933, 935, 937, 938"3" na stránce 981, 939, 941, 942, 943 ,947, 948, 949, 950, 951, 954"4" na stránce 981, 964, 970
1006, 1025, 1026, 1027, 1040, 1041, 1042, 1043, 1046, 1047, 1051, 1088, 1089, 1097, 1098, 1089

1112, 1114, 1115, 1122, 1123, 1124, 1129, 1130, 1132, 1133, 1140, 1141, 1142, 1143, 1144, 1145, 1146, 1147, 1148, 1149, 1153, 1156, 1157, 1140, 1140, 1146, 1146, 1146, 1145, 1146, 1147, 1148, 1149, 1153,

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 981]

9044, 9048, 9449

12712

13488

17584

33722[“4” na stránce 981]

Notes:

1. 931 používá 939 pro konverzi.
2. 932 používá 942 pro konverzi.
3. 938 používá 948 pro konverzi.
4. 954 a 33722 používají 5050 pro konverzi.
5. Pouze v systémech Windows a Linux .

Podpora produktu IBM i pro kódování Unicode

IBM i

Podrobné informace o podpoře UNICODE najdete v příslušné publikaci IBM i týkající se vašeho operačního systému.

Podpora produktu IBM MQ for z/OS pro kódování Unicode

z/OS

Při převodu na IBM MQ for z/OS a z podporovaných Unicode CCSID (pokud možno 1200 nebo 1208) jsou podporovány CCSID, které nejsou 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, 836, 837, 838, 848, 849, 851, 852, 855, 857, 858, 859, 864, 865, 866, 871, 868, 874, 875, 878, 880, 891, 895, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 896, 870,

901, 903, 904, 915, 916, 923, 930, 938, 941, 946, 947, 944, 946, 947, 946, 944, 946, 947, 944, 944, 946, 947, 948, 949, 950, 947, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 946, 9

1006, 1006, 1008, 1009, 1010, 1011, 1012, 1013, 1014, 1015, 1016, 1017, 1018, 1027, 1040, 1041, 1042, 1042, 1043, 1047, 1051, 1051, 1088, 1089, 1098, 1051, 1088, 1089, 1097, 1051, 1088, 1089, 1097, 1051, 1089, 1049, 1049, 1089, 1089

1112, 1114, 1124, 1125, 1123, 1142, 1140, 1142, 1140, 1140, 1145, 1142, 1143, 1144, 1155, 1146, 1153, 1154, 1156, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1164, 1161, 1158, 1159, 1160, 1161, 1162, 1164, 1161, 1158, 1156, 1157, 1158, 1159, 1160, 1161, 1162, 1164, 1140, 1140, 1140, 1140, 1140, 1149, 1140, 1140,

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

Kodové normy na 64bitových platformách

Tyto informace použijte, chcete-li se dozvědět více o standardech kódování na 64bitových platformách a o preferovaných datových typech.

Preferované datové typy

Tyto typy nikdy nemění velikost a jsou k dispozici na 32bitových a 64bitových platformách IBM MQ :

Tabulka 676. Názvy a délky datových typů

Název	Délka
MQLONG	4 bajty
MQULONG	4 bajty
MQINT32	4 bajty
MQUINT32	4 bajty
MQINT64	8 bajtů
MQUINT64	8 bajtů

ALW Standardní datové typy v systému AIX, Linux, and Windows

Získejte informace o standardních datových typech na 32bitových aplikacích AIX and Linuxa 64bitových aplikacích AIX, Linux, and Windows .

32bitové aplikace produktu AIX and Linux

Linux AIX

Tabulka 677. Názvy typů dat a délky pro 32bitové aplikace AIX and Linux

Název	Délka
ZNAK	1 bajt
short	2 bajty
celé číslo	4 bajty
long	4 bajty
float	4 bajty

Tabulka 677. Názvy typů dat a délky pro 32bitové aplikace AIX and Linux (pokračování)

Název	Délka
dvojitý	8 bajtů
long double	8 bajtů
Ukazatel	4 bajty
ptrdiff_t	4 bajty
velikost_t	4 bajty
time_t	4 bajty
hodin_hodin	4 bajty
wchar_t	4 bajty

AIX Všimněte si, že na AIX wchar_t je 2 bajty.

64bitové aplikace produktu AIX and Linux

Linux AIX

Tabulka 678. Názvy datových typů a délky pro 64bitové aplikace produktu AIX and Linux

Název	Délka
ZNAK	1 bajt
short	2 bajty
celé číslo	4 bajty
long	8 bajtů
float	4 bajty
dvojitý	8 bajtů
long double	8 bajtů
Ukazatel	8 bajtů
ptrdiff_t	8 bajtů
velikost_t	8 bajtů
time_t	8 bajtů
hodin_hodin	4 bajty
wchar_t	4 bajty

AIX Všimněte si, že na AIX wchar_t je 2 bajty.

Windows 64bitové aplikace

Windows

Tabulka 679. Názvy datových typů a délky pro 64bitové aplikace Windows

Název	Délka
ZNAK	1 bajt
short	2 bajty

Tabulka 679. Názvy datových typů a délky pro 64bitové aplikace Windows (pokračování)

Název	Délka
celé číslo	4 bajty
long	4 bajty
float	4 bajty
dvojitý	8 bajtů
long double	8 bajtů
Ukazatel	8 bajtů Všimněte si, že všechny ukazatele jsou 8 bajtů.
ptrdiff_t	8 bajtů
velikost_t	8 bajtů
time_t	8 bajtů
hodin_hodin	4 bajty
wchar_t	2 bajty
Word	2 bajty
DWORD	4 bajty
aplikace	8 bajtů
SOUBOR HFILE	4 bajty

Pokyny ke kódování v systému Windows

Windows

HANDLE hf;

Použití

```
hf = CreateFile((LPCTSTR) FileName,
                Access,
                ShareMode,
                xihSecAttsNTRestrict,
                Create,
                AttrAndFlags,
                NULL);
```

Nepoužívat

```
HFILE hf;
hf = (HFILE) CreateFile((LPCTSTR) FileName,
                        Access,
                        ShareMode,
                        xihSecAttsNTRestrict,
                        Create,
                        AttrAndFlags,
                        NULL);
```

při vytváření této chyby se zobrazí chyba.

zazel_t len fgets

Použití

```
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

Použití

```
printf("My struc pointer: %p", pMyStruc);
```

Nepoužívat

```
printf("My struc pointer: %x", pMyStruc);
```

Pokud potřebujete hexadecimální výstup, musíte tisknout horní a dolní 4 bajty odděleně.

char * ptr

Použití

```
char * ptr1;
char * ptr2;
size_t bufLen;

bufLen = ptr2 - ptr1;
```

Nepoužívat

```
char *ptr1;
char *ptr2;
UINT32 bufLen;

bufLen = ptr2 - ptr1;
```

alignBytes

Použití

```
alignBytes = (unsigned short) ((size_t) address % 16);
```

Nepoužívat

```
void *address;
unsigned short alignBytes;

alignBytes = (unsigned short) ((UINT32) address % 16);
```

DÉLKA

Použití

```
len = (UINT32) ((char *) address2 - (char *) address1);
```

Nepoužívat

```
void *address1;
void *address2;
```

```
UINT32 len;

len = (UINT32) ((char *) address2 - (char *) address1);
```

sscanf

Použití

```
MQLONG SBCSprt;

sscanf(line, "%d", &SBCSprt);
```

Nepoužívat

```
MQLONG SBCSprt;

sscanf(line, "%1d", &SBCSprt);
```

Produkt `%ld` se pokusí vložit 8bajtový typ do 4bajtového typu; použije se pouze `%l` , pokud se jedná o skutečný datový typ produktu `long` . MQLONG, UINT32 a INT32 jsou definovány jako čtyři bajty, stejné jako `int` na všech platformách IBM MQ :

IBM i IBM i Application Programming Reference (ILE/RPG)

Programování aplikací pro produkt IBM i.

Tyto informace vám pomohou při vývoji aplikací pro produkt IBM i.

- “Popisy datových typů v systému IBM i” na stránce 987
- “Volání funkcí v systému IBM i” na stránce 1237
- “Atributy objektů v systému IBM i” na stránce 1353
- “Aplikace” na stránce 1398
- “Návratové kódy pro IBM i (ILE RPG)” na stránce 1411
- “Pravidla pro ověření platnosti voleb MQI pro produkt IBM i (ILE RPG)” na stránce 1412
- “Kódování počítače v systému IBM i” na stránce 1415
- “Volby sestav a příznaky zpráv v systému IBM i” na stránce 1418

Zamítnutí režimu kompatibility pro aplikace RPG a COBOL v systému IBM i

IBM i

Z produktu IBM MQ for IBM i 9.0již produkt neposkytuje podporu pro aplikace v jazycích RPG nebo COBOL, které používají dynamické sestavení známé jako režim kompatibility. Tento režim operace byl potřebný pro aplikace, které jsou napsány před produktem MQSeries 5.1, a následné verze produktu za předpokladu, že pro tyto aplikace jsou kompatibilní běhové prostředí, i když zakladače potřebné pro jejich kompilaci byly odebrány v produktu IBM WebSphere MQ 6.0. Dynamické propojení (režim kompatibility) bylo poskytnuto následujícími programy v knihovně QMQM, které jsou odebrány na IBM MQ for IBM i 9.0:

- AMQVSTUB
- AMQZSTUB
- QMQM
- MQCLOSE
- MQCONN
- MQDISC
- MQGET
- MQINQ
- MQOPEN

- MQPUT
- MQPUT1
- MQSET

Z produktu IBM MQ for IBM i 9.0je třeba znovu kompilovat aplikace používající tento režim kompatibility pro použití statických svázaných volání MQ , která jsou poskytována servisními programy LIBMQM a LIBMQM_R. Ukázkový program, jako např. AMQ3PUT4 a AMQ3GET4, vám ukáže, jak používat tento programovací model. Další informace o použití těchto volání MQ najdete v příručce IBM i Application Programming Reference (ILE/RPG).

Notes:

- Je třeba rekódovat aplikace, které aktuálně používají rozhraní CALL 'QMQM', místo toho budete používat servisní program LIBMIBM.

  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 pro použití režimu kompatibility, přestanou fungovat.
- Jsou-li aplikace svázány s programem služby LIBMQM na adrese IBM MQ for IBM i 8.0, neměli byste tyto aplikace v produktu IBM MQ for IBM i 9.0 nebo později znovu kompilovat nebo znovu sestavit.
- Není možné instalovat více než jednu verzi produktu IBM MQ for IBM i do stejné oblasti.

Chcete-li zjistit, zda váš program RPG nebo COBOL používá režim kompatibility, použijte příkaz **DSPPGMREF** (Zobrazení programových odkazů) k zobrazení externích programů volaných aplikačním programem. Pokud zde existují odkazy na programy uvedené v této sekci, nebude program spuštěn v produktu IBM MQ for IBM i 9.0 nebo novějším. Následující příklad výstupu příkazu **DSPPGMREF** ukazuje tři objekty programu, které jsou zamítnuté, 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 znovu kompilovány s použitím metody Bound Procedure Call popsané v tématu Příprava programů v jazyce COBOL v produktu IBM i.

Pokusíte-li se spustit aplikační program v produktu IBM MQ for IBM i 9.0 nebo novější, který používá režim kompatibility, nejčastěji zaznamenaným prvním chybou je MCH3401 se pokouší volat program MQCONN nebo QMQM.

Související úlohy

Vývoj aplikací

IBM i Popisy datových typů v systému IBM i

Tato kolekce témat obsahuje popis datových typů používaných při programování v produktu IBM i .

Konvence použité v popisu datových typů

Pro každý elementární datový typ tyto informace poskytují popis jeho použití ve formě, která je nezávislá na programovacím jazyce. To je následováno typickými deklaracemi ve verzi ILE programovacího jazyka RPG. Definice základních datových typů jsou zde obsaženy, aby se zajistila konzistence. RPG používá specifikace 'D'', kde pracovní pole mohou být deklarována pomocí libovolných atributů, které potřebujete. Můžete to však provést ve specifikacích výpočtu, kde se pole používá.

Chcete-li použít elementární datové typy, vytvořte:

- Člen /COPY obsahující všechny datové typy, nebo
- Externí datová struktura (PF) obsahující všechny datové typy. Pak budete muset uvést svá pracovní pole s atributy 'LIKE' odpovídající pole datového typu.

Přínosem druhé volby je, že definice mohou být použity jako 'POLE REFERENČNÍ SOUBOR' pro jiné objekty IBM i . Změní-li se definice datového typu IBM MQ , je to relativně jednoduchou záležitostí k obnovení těchto objektů.

Elementární datové typy

Všechny ostatní datové typy popsané v tomto oddílu se rovnají buď přímo těmto elementárním datovým typům, nebo k agregování těchto elementárních datových typů (polí nebo struktur).

Tabulka 680. Elementární datové typy

Datový typ	**Zastupování**
MQBOOL	10ciferné celé číslo se znaménkem
MQBYTE	1bajtové alfanumerické pole
MQBYTE16	16bajtové alfanumerické pole
MQBYTE24	24bajtové alfanumerické pole
MQBYTE32	32bajtové alfanumerické pole
MQBYTE64	64bajtové 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	64bajtové alfanumerické pole
MQCHAR128	128bajtové alfanumerické pole
MQCHAR256	256bajtové alfanumerické pole
MQFLOAT32	4bajtové číslo s pohyblivou řádovou čárkou
MQFLOAT64	8bajtové číslo s pohyblivou řádovou čárkou
KONFIGURACE MQHCONFIG	Popisovač konfigurace
MQHCONN	10ciferné celé číslo se znaménkem
MQZPR	Popisovač zprávy, který poskytuje přístup ke zprávě
MQOBJ	10ciferné celé číslo se znaménkem
MQINT8	8bitové podepsané celé číslo
MQINT16	16bitové podepsané celé číslo

Tabulka 680. Elementární datové typy (pokračování)	
Datový typ	**Zastupování**
MQINT32	32bitové podepsané celé číslo
MQINT64	64bitové podepsané celé číslo
MQLONG	32bitové podepsané celé číslo
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	32bitové celé číslo bez znaménka
PMQACH	Ukazatel na datovou strukturu typu MQACH
PMQAIR	Ukazatel na datovou strukturu typu MQAIR
PMQAXC	Ukazatel na datovou strukturu typu MQAXC
PMAXP	Ukazatel na datovou strukturu typu MAXP
PMQBHO	Ukazatel na datovou strukturu typu MQBMHO
OBJEKT 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

Tabulka 680. Elementární datové typy (pokračování)	
Datový typ	**Zastupování**
PMQDMPO	Ukazatel na datovou strukturu typu MQDMPO
PMQEP.	Ukazatel na datovou strukturu typu MQEPH
PMQFLOAT32	Ukazatel na data typu MQFLOAT32
PMQFLOAT64	Ukazatel na data typu MQFLOAT64
PMQFUNC	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
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

Tabulka 680. Elementární datové typy (pokračování)	
Datový typ	**Zastupování**
PMQSMPO	Ukazatel na datovou strukturu typu MQSMPO
PMQSRO	Ukazatel na datovou strukturu typu MQSRO
PMQSTS	Ukazatel na datovou strukturu typu MQSTS
ID 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
PMQULELONG.	Ukazatel na data typu MQULONG
PMQVOID	Ukazatel
PMQWIH	Ukazatel na datovou strukturu typu MQWIH
PMQXQH	Ukazatel na datovou strukturu typu MQXQH

IBM i *MQBOOL na IBM i*

Datový typ MQBOOL představuje logickou hodnotu. Hodnota 0 reprezentuje hodnotu false. Jakákoli jiná hodnota představuje true.

MQBOOL musí být zarovnán jako pro datový typ MQLONG.

IBM i *MQBYTE na IBM i*

Datový typ MQBYTE představuje jeden bajt dat.

Žádný konkrétní výklad není umístěn na byte-je považován za řetězec bitů, a ne jako binární číslo nebo znak. Není vyžadováno žádné zvláštní zarovnání.

Pole MQBYTE se někdy používá k reprezentaci oblasti hlavní paměti s charakterem, která není známa správci front. Oblast může například obsahovat data zprávy aplikace nebo strukturu. Vyrovnání hranice této oblasti musí být slučitelné s povahou údajů, které jsou v něm obsaženy.

IBM i *MQBYTEn (řetězec n bajtů) v systému 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 zvláštní zarovnání.

Pokud jsou data v řetězci kratší než definovaná délka řetězce, musí být data polstrovaná s hodnotami null, aby vyplnily řetězec.

Když správce front vrátí do aplikace bajtové řetězce (například na volání MQGET), správce front vždy bude mít k dispozici hodnoty null s definovanou délkou řetězce.

Konstanty jsou k dispozici, které definují délky polí bajtových řetězců.

IBM i *MQCHAR (znak) v IBM i*

Datový typ MQCHAR představuje jeden znak.

Identifikátor kódované znakové sady tohoto znaku se nachází v atributu správce front (viz atribut **CodedCharSetId** v tématu CodedCharSetId). Není vyžadováno žádné zvláštní zarovnání.

Poznámka: Data zprávy aplikace určená na voláních MQGET, MQPUT a MQPUT1 jsou popsána datovým typem MQBYTE, nikoli datovým typem MQCHAR.

IBM i *MQCHARn (řetězec n znaků) v systému 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 zvláštní zarovnání.

Pokud jsou data v řetězci kratší než definovaná délka řetězce, data musí být vyplněna mezerami, aby se řetězec vyplnil. V některých případech může být znak null použit k ukončení řetězce předčasně, místo doplnění mezerami; znak hex 00 a znaky následující za ním jsou považovány za mezery, až do definované délky řetězce. Místa, kde může být použita hodnota null, jsou identifikována v popisech volání a datových typů.

Když správce front vrátí do aplikace znakové řetězce (například při volání MQGET), bude správce front vždy obsahovat mezery až do definované délky řetězce; správce front nepoužívá k oddělování řetězce znak null.

K dispozici jsou konstanty, které definují délky polí znakového řetězce.

IBM i *MQFLOAT32 v systému IBM i*

Datový typ MQFLOAT32 je 32bitové číslo s pohyblivou řádovou čárkou, které je reprezentováno pomocí standardního formátu IEEE s pohyblivou řádovou čárkou.

MQFLOAT32 musí být zarovnáno na 4bajtové hranici.

IBM i *MQFLOAT64 v systému 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 8bajtovou hranici.

MQHCONFIG-konfigurační popisovač

Datový typ MQHCONFIG představuje konfigurační popisovač, tj. komponentu, která je konfigurována pro konkrétní instalovatelnou službu. Manipulátor konfigurace musí být zarovnán na jeho přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHCONN (Manipulátor připojení) v systému IBM i*

Datový typ MQHCONN představuje manipulátor připojení, tj. připojení ke konkrétnímu správci front.

Manipulátor připojení musí být zarovnán na jeho přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHMSG (Popisovač zprávy) na IBM i*

Datový typ **MQHMSG** představuje popisovač zprávy, který poskytuje přístup ke zprávě.

Popisovač zprávy musí být zarovnán na 8bajtovou hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQHOBJ (Popisovač objektu) na IBM i*

Datový typ MQHOTBJ představuje popisovač objektu, který poskytuje přístup k objektu.

Manipulátor objektu musí být zarovnán na jeho přirozené hranici.

Poznámka: Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

IBM i *MQINT8 (8bitové celé číslo se znaménkem) v systému IBM i*

Datový typ MQINT8 je 8bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -128 až +127, pokud není jinak omezen kontextem.

IBM i *MQINT16 (16bitové podepsané celé číslo) v systému IBM i*

Datový typ MQINT16 je 16bitové celé číslo se znaménkem, které může mít libovolnou hodnotu v rozsahu -32 768 až +32 767, pokud není jinak omezen kontextem.

Hodnota MQINT16 musí být zarovnána na 2bajtovou hranici.

IBM i *MQINT32 (32bitové celé číslo) v systému IBM i*

Datový typ MQINT32 je 32bitové celé číslo se znaménkem.

Je ekvivalentní MQLONG.

IBM i *MQINT64 (64bitové celé číslo) v systému 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 není jinak omezen kontextem.

V případě jazyka COBOL je platný rozsah omezen na -999 999 999 999 999 až +999 999 999 999 999. MQINT64 by měl být zarovnán na 8bajtovou hranici.

IBM i *MQLONG (Long integer) v 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, pokud není jinak omezeno kontextem, zarovnáno podle jeho přirozené hranice.

MQPID-identifikátor procesu

Identifikátor procesu IBM MQ .

Jedná se o stejný identifikátor, který se používá v trasování IBM MQ a FFST výpisů paměti, 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; toto je 16bajtová hranice na IBM i.

Některé programovací jazyky podporují typovaní ukazatele; rozhraní MQI je také používá v několika případech.

MQTID-identifikátor podprocesu

Identifikátor podprocesu MQ .

Jedná se o stejný identifikátor, který je použit v trasování MQ a výpisů paměti produktu FFST , ale může se lišit od identifikátoru podprocesu operačního systému.

IBM i *MQUINT8 (8bitové celé číslo bez znaménka) v IBM i*

Datový typ MQUINT8 je 8bitové celé číslo bez znaménka, které může mít jakoukoli hodnotu v rozsahu 0 až +255, pokud není jinak omezen kontextem.

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 není jinak omezeno kontextem.

MQUINT16 musí být zarovnán na 2bajtovou hranici.

IBM i *MQUINT32 (32bitové celé číslo bez znaménka) v IBM i*

Datový typ MQUINT32 je 32bitové celé číslo bez znaménka. Je ekvivalentní k MQULONG.

MQUINT64 -64bitové, nepodepsané celé číslo

Datový typ MQUINT64 je 64bitové celé číslo bez znaménka, které může mít libovolnou hodnotu v rozsahu 0 až +18 446 744 073 709 551 615, pokud není jinak omezeno kontextem.

Pro COBOL je platný rozsah omezen na 0 až +999 999 999 999 999 999. Hodnota MQUINT64 by měla být zarovnána na 8bajtovou hranici.

MQULONG-32bitové, celé číslo bez znaménka

Datový typ MQULONG je 32bitové binární celé číslo bez znaménka, které může mít jakoukoli hodnotu v rozsahu 0 až + 4 294 967 294, pokud není jinak omezen kontextem.

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-pointer to a datová struktura typu MQAXP

Ukazatel na datovou strukturu typu MQAXP.

PMQBMHO-pointer to a datová struktura 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-pointer to a data structure of type MQCBC
Ukazatel na datovou strukturu typu MQCBC.

PMQCBD-pointer to a data structure of type 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-pointer to a data structure of type of 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-pointer to a data type of MQHMSG
Ukazatel na datový typ MQHMSG.

PMQHOBJ-ukazatel na data typu MQHOBJ
Ukazatel na data typu MQSMPO.

PMQIIH-pointer to a 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) v systému IBM i*
Datový typ PMQINT32 je ukazatel na data typu MQINT32. Je ekvivalentní k PMQLONG.

IBM i *PMQINT64 (Ukazatel na data typu MQINT64) v systému IBM i*
Datového typu 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-pointer to a data structure of type MQMDE
Ukazatel na datovou strukturu typu MQMDE.

PMQMDI-pointer to a data structure of type 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-pointer to a datová struktura typu MQOD
Ukazatel na datovou strukturu typu MQOD.

PMQOR-pointer to a data structure of type MQOR
Ukazatel na datovou strukturu typu MQOR.

PMQPD-pointer to a datová struktura 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-pointer to a datová struktura typu MQRMH
Ukazatel na datovou strukturu typu MQRMH.

PMQRR-pointer to a data structure of type MQRR
Ukazatel na datovou strukturu typu MQRR.

PMQSCO-pointer to a datová struktura typu MQSCO
Ukazatel na datovou strukturu typu MQSCO.

.

PMQSD-ukazatel na datovou strukturu typu MQSD
Ukazatel na datovou strukturu typu MQSD.

PMQSMPO-ukazatel na strukturu dat typu MQSMPO
Ukazatel na datovou strukturu typu MQSMPO.

PMQSRO-pointer to a datovou strukturu typu MQSRO
Ukazatel na datovou strukturu typu MQSRO.

PMQSTS-pointer to a datová struktura 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 k PMQULONG.

IBM i *PMQUINT64 (Ukazatel na data typu MQUINT64) v systému IBM i*
Datového typu PMQUINT64 je ukazatel na data typu MQUINT64.

PMQULONG-ukazatel na data typu MQULONG
Ukazatel na data typu MQULONG.

PMQVOID-ukazatel
Ukazatel.

PMQWIH-pointer to a datová struktura 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 poznámek jsou:

- “soubory COPY” na stránce 1000
- “Volání” na stránce 1002
- “Parametry volání” na stránce 1002
- “Struktury” na stránce 1002
- “Pojmenované konstanty” na stránce 1003
- “Procedury MQI” na stránce 1003
- “Aspekty používání podprocesů” na stránce 1003
- “Vázané zpracování” na stránce 1004
- “Kódování vázaných volání” na stránce 1004
- “Notační konvence” na stránce 1005

soubory COPY

K dispozici jsou různé soubory COPY, které vám pomohou při psaní aplikačních programů RPG, které používají systém front zpráv. K dispozici jsou tři sady souborů COPY:

- Soubory typu COPY s názvy končícím písmenem *G* jsou určeny pro použití s programy, které používají statické sestavení. Tyto soubory jsou inicializovány s výjimkami uvedenými v “Struktury” na stránce 1002.
- Soubory typu COPY s názvy končícím písmenem *H* jsou určeny pro použití s programy, které používají statické sestavení, ale **nejsou** inicializovány.
- Soubory typu COPY s názvy končícím písmenem *R* jsou určeny pro použití s programy, které používají dynamické sestavení. Tyto soubory jsou inicializovány s výjimkami uvedenými v “Struktury” na stránce 1002.

Soubory COPY jsou umístěny 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 Tabulka 681 na stránce 1000.

Tabulka 681. Soubory RPG COPY

Název souboru (statické sestavení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické propojení, inicializováno, CMQ* R)**	**Obsah**
CMQBOG.	CMQBOHFCH	-	Začátek struktury voleb
CMQCDG	CMQCDH	CMQCDR	Struktura definice kanálu
CMQCFBFG.	CMQCFBFH.	-	Parametr bitového filtru PCF

Tabulka 681. Soubory RPG COPY (pokračování)			
Název souboru (statické sestavení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické propojení, inicializováno, CMQ* R)**	**Obsah**
CMQCFG	-	-	Konstanty pro PCF a události
CMQCFBSG.	CMQCFBSH.	-	PCF bajtový řetězec
CMQCFGRG	CMQCFGRH	-	parametr skupiny PCF
CMQCFIFG	CMQCFIFH	-	Celočíselný parametr filtru PCF
CMQCFHG.	CMQCFHH	-	Záhlaví PCF
CMQCFILG	CMQCFILH	-	Celočíselná struktura parametrů seznamu PCF
CMQCFING	CMQCFINH	-	Struktura celočíselných parametrů PCF
CMQCFSFG.	CMQCFSFH.	-	Parametr filtru řetězce PCF
CMQCFSLG	CMQCFSLH	-	Struktura parametru seznamu řetězců PCF
CMQCFSTG.	CMQCFSTH	-	Struktura řetězcových parametrů PCF
CMQCFXLG.	CMQCFXLH	-	Krátký název PCF pro CFIL64
CMQCFXNG	CMQCFXNH	-	Krátký název PCF pro CFIN64
CMQCIHGUESTO	CMQCIHH.	-	Struktura záhlaví informací produktu CICS
CMQCNOG	CMQCNOHCH	-	Struktura voleb připojení
CMQCSPG	CMQCSPH	-	Parametry zabezpečení
CMQCXPG	CMQCXPH	CMQCXPR	Struktura výstupních parametrů kanálu
CMQDHG.	CMQDHH.	CMQDHR	Struktura záhlaví distribuce
CMQDLHG.	CMQDLHH	CMQDLHR	Struktura záhlaví nedoručených zpráv
CMQDXPG	CMQDXPH	CMQDXPR	Struktura výstupního parametru konverze dat
CMQEPHG	CMQEPHHUS	-	Struktura záhlaví Embedded PCF
CMQG	-	CMQR	Pojmenované konstanty pro hlavní rozhraní MQI
CMQGMOG	CMQGMOH	CMQGMOR	Získat strukturu voleb zprávy
CMQIIHG	CMQIIHH	CMQIIHR	Struktura záhlaví informací produktu IMS
CMQMDEG	CMQMDEHCH	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	CMQODHLICE	CMQODR	Struktura deskriptoru objektu

Tabulka 681. Soubory RPG COPY (pokračování)			
Název souboru (statické sestavení, inicializováno, CMQ* G)	**Název souboru (statické sestavení, neinicializováno, CMQ* H)**	**Název souboru (dynamické propojení, inicializováno, CMQ* R)**	**Obsah**
CMQORG	CMQORHUR.	CMQORR	Struktura záznamu objektu
CMQPMOG	CMQPMOH	CMQPMOR	Vložit strukturu voleb zprávy
CMQPSG	-	-	Konstanty pro publish/odběr
CMQRFHG.	CMQRFHH.	-	Pravidla a formátování struktury záhlaví
CMQRFH2G	CMQRFH2H	-	Pravidla a formátování struktury 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	CMQTMSTAR NAME	Struktura zprávy spouštěče
CMQWIHG.	CMQWIHH.	-	Struktura záhlaví pracovních informací
CMQXG	-	CMQXR	Pojmenované konstanty pro ukončení konverze dat
CMQXQHLG	CMQXQHH	CMQXQHR	Struktura záhlaví přenosové fronty

Volání

Volání jsou popsána pomocí jejich jednotlivých názvů.

Parametry volání

Některé parametry předané do MQI mohou mít více než jednu souběžnou funkci. Je tomu tak proto, že předávaná hodnota celého čísla je často testována na nastavení jednotlivých bitů v poli, a ne na její 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.
- PŘÍKAZ MQTMC
- MQTMC2

Tyto počáteční hodnoty jsou definovány v příslušné tabulce pro každou strukturu.

Deklarace struktury neobsahují příkazy DS . To umožňuje aplikaci deklarovat buď jedinou strukturu dat, nebo strukturu dat s více výskyty, zakódováním příkazu DS a následným použitím příkazu /COPY na kopírování 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 zajišťují 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 nikoli hodnoty, které představují, protože to zlepšuje čitelnost zdrojového kódu programu.

Když je soubor COPY CMQG zahrnut do programu pro definování konstant, kompilátor jazyka RPG vydá mnoho zpráv závažnosti-nula pro konstanty, které program nepoužívá; tyto zprávy jsou neškodné a lze je bezpečně ignorovat.

Procedury MQI

Při použití vázaných volání ILE se musíte při vytváření programu svázat s procedurami MQI. Tyto postupy jsou dle potřeby exportovány z následujících servisních programů:

QMQM/LIBMQM
Tento servisní program obsahuje vazby s jedním vláknem pro verzi 5.1 a vyšší. Prohlédněte si následující sekci pro speciální posouzení při zápisu aplikací s podporou podprocesů.

QMQM/LIBMQM_R
Tento servisní program obsahuje vazby s podporou více podprocesů pro verzi 5.1 a vyšší. Prohlédněte si následující sekci pro speciální posouzení při zápisu aplikací s podporou podprocesů.

QMQM/LIBMQIC
Tento servisní program je určen pro vázání klientských aplikací bez podprocesů.

QMQM/LIBMQIC_R
Tento servisní program je určen pro vázání klientských aplikací s podprocesy.

Pomocí příkazu CRTPGM vytvořte své programy. Následující příkaz například vytvoří jednovláknový program, který používá vázané volání ILE:

```
CRTPGM PGM(MYPROGRAM) BNDSRVPGM(QMQM/LIBMQM)
```

Aspekty používání podprocesů

Kompilátor jazyka RPG použitý pro produkt IBM i je součástí sady vývojových nástrojů WebSphere Development Toolkit a WebSphere Development Studio for IBM i a je známá jako kompilátor ILE RPG IV Compiler.

Obecně řečeno, programy RPG by neměly používat servisní programy s více vlákny. Výjimky jsou RPG programy vytvořené pomocí kompilátoru ILE RPG IV a obsahující klíčové slovo THREAD(*SERIALIZE) v rámci specifikace řízení. Avšak i když jsou tyto programy bezpečné, je třeba pečlivě zvážit celkový návrh aplikace, neboť THREAD(*SERIALIZE) vynucuje serializaci procedur RPG na úrovni modulu, a to může mít nepříznivý vliv na celkový výkon.

Jsou-li programy RPG používány jako uživatelské procedury pro převod dat, je třeba je zajistit neporušenost podprocesů a je třeba je překompilovat pomocí kompilátoru 4.4 ILE RPG nebo výše uvedeného s parametrem THREAD(*SERIALIZE) uvedeným ve specifikaci řízení.

Další informace o vytváření podprocesů naleznete v příručce *IBM i IBM MQ Development Studio: ILE RPG Referencea IBM i IBM MQ Development Studio: ILE RPG Programmer's Guide*.

Vázané zpracování

Funkce MQI syncpoint MQCMIT a MQBACK jsou dostupné pro programy ILE RPG spuštěné v normálním režimu; tato volání umožňují programu potvrdit a vrátit změny prostředků MQ .

Kódování vázaných volání

Procedury ILE MQI jsou uvedeny v seznamu Tabulka 682 na stránce 1004.

Tabulka 682. ILE RPG svázaná volání 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
MQPUT	Y	Y
MQPUT1	Y	Y
MQSET	Y	Y
MQXCNVC	Y	Y

Chcete-li použít tyto procedury, musíte:

1. Definujte externí procedury ve vašich specifikacích ' D'. Všechny tyto objekty jsou dostupné v rámci členu souboru COPY CMQG obsahující pojmenované konstanty.
2. Použijte kód operace CALLP pro volání procedury spolu s jeho parametry.

Volání MQOPEN vyžaduje například 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, budete potřebovat 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 členu COPY CMQODG, který jej rozdělí na své komponenty.

Notační konvence

Poslední uvedená 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 polí nebo znakových řetězců s velikostí, která není pevná. Pro znázornění numerických konstant se používá malá písmena "n" . Je-li deklarace pro daný parametr kódována, musí být "n" nahrazena číselnou hodnotou, která je povinná.

IBM i MQAIR (záznam ověřovacích informací) v systému IBM i

Struktura MQAIR představuje záznam ověřovacích informací.

Přehled

Účel: Struktura MQAIR umožňuje aplikaci spuštěnou jako klient produktu IBM MQ zadat informace o ověřovateli, který má být použit pro připojení klienta. Struktura je vstupním parametrem volání MQCONNX.

Znaková sada a kódování: Data v aplikaci MQAIR musí být ve znakové sadě dané atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT.

- “Pole” na stránce 1005
- “Počáteční hodnoty” na stránce 1007
- “Deklarace RPG” na stránce 1007

Pole

Struktura MQAIR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

AICN (10ciferné celé číslo se znaménkem)

Jedná se o název hostitele nebo síťovou adresu hostitele, na kterém je spuštěn server LDAP. Za ním může následovat volitelné číslo portu uzavřené v závorkách.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Není-li hodnota platná, volání selže s kódem příčiny RC2387.

Výchozí číslo portu je 389.

Toto je vstupní pole. Délka tohoto pole je dána LNAICN. Počáteční hodnota tohoto pole obsahuje prázdné znaky.

AITYP (10ciferné celé číslo se znaménkem)

Jedná se o typ ověřovacích informací obsažených v záznamu.

Hodnota musí být:

AITLDP
Odvolání certifikátu pomocí serveru LDAP.

Není-li hodnota platná, volání selže s kódem příčiny RC2386.

Toto je vstupní pole. Počáteční hodnota tohoto pole je AITLDP.

AIPW (10ciferné celé číslo se znaménkem)

Jedná se o heslo potřebné pro přístup k serveru CRL LDAP.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Pokud server LDAP nevyžaduje heslo nebo vynechte jméno uživatele LDAP, *AIPW* musí mít hodnotu null nebo být prázdný. Vynecháte-li jméno uživatele LDAP a *AIPW* hodnotu null nebo je prázdné, volání selže s kódem příčiny RC2390.

Toto je vstupní pole. Délka tohoto pole je dána LNLDPW. Počáteční hodnota tohoto pole obsahuje prázdné znaky.

AILUL (10ciferné celé číslo se znaménkem)

Toto je délka v bajtech jména uživatele LDAP adresovaného polem *AILUP* nebo *AILUO* . Hodnota musí být v rozsahu nula až LNDISN. Není-li hodnota platná, volání selže s kódem příčiny RC2389.

Pokud zahrnutý server LDAP nevyžaduje jméno uživatele, nastavte toto pole na nulu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AILUO (desetimístné podepsané celé číslo)

Jedná se o posun v bajtech jména uživatele LDAP od začátku struktury MQAIR.

Odsazení může být kladné nebo záporné. Pole je ignorováno, pokud *LDAPUserNameLength* je nula.

Můžete použít buď *LDAPUserNamePtr* nebo *LDAPUserNameOffset* , abyste uvedli jméno uživatele LDAP, ale ne obojí; podrobnosti najdete v popisu pole *LDAPUserNamePtr* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

AILUP (10ciferné celé číslo se znaménkem)

Jedná se o jméno uživatele LDAP.

Skládá se z rozlišujícího jména uživatele, který se pokouší o přístup k serveru LDAP CRL. Je-li hodnota kratší než délka zadaná parametrem *AILUL*, ukončete ji znakem null nebo jej odpalovat mezerami na délku *AILUL*. Pole je ignorováno, pokud *AILUL* je nula.

Jméno uživatele služby LDAP můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *AILUP*

  V takovém případě může aplikace deklarovat řetězec, který je oddělen od struktury MQAIR, a nastavit proměnnou *AILUP* na adresu řetězce.

  Zvažte použití *AILUP* pro programovací jazyky, které podporují datový typ ukazatele v módě, který je přenosný do různých prostředí (například programovací jazyk C).

- Použití pole offsetu *AILUO*

  V takovém případě musí aplikace deklarovat složenou strukturu obsahující strukturu MQSCO, za kterou následuje pole záznamů MQAIR, za nimiž následují řetězce názvů uživatelů LDAP, a nastavit proměnnou *AILUO* na posun příslušného řetězce názvu od začátku struktury MQAIR. Ujistěte se, že je tato hodnota správná a že má hodnotu, která může být umístěna v rámci MQLONG (nejvíce omezující programovací jazyk je COBOL, pro který je platný rozsah -999 999 999 až +999 999 999).

  Zvažte použití *AILUO* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele v módě, který nemusí být přenosný do různých prostředí (například programovací jazyk COBOL).

Zvolená technika je vybrána, používá se pouze jeden z *AILUP* a *AILUO* ; volání selže s kódem příčiny RC2388.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null v těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null.

Poznámka: Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

AISID (10ciferné celé číslo se znaménkem)

Hodnota musí být:

AISIDV
Identifikátor pro záznam ověřovacích informací.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je AISIDV.

AIVER (10ciferné celé číslo se znaménkem)

Hodnota musí být:

AIVER1
Záznam ověřovacích informací Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AIRVERC
Aktuální verze záznamu ověřovacích informací.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je AIVER1.

Počáteční hodnoty

Tabulka 683. Počáteční hodnoty polí v aplikaci MQAIR pro MQAIR

Název pole	Název konstanty	Hodnota konstanty
AISID	AISIDV	'AIR¬'
AIVER	AIVERC	1
AITYP	AITLDP	1
AICN	Není	Nulový řetězec nebo prázdné znaky
AILUP	Není	Nulový ukazatel nebo bajty null
AILUO	Není	0
AILUL	Není	0
AIPW	Není	Nulový řetězec nebo prázdné znaky

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 (Vyrovnávací paměť pro volby zpracování vyrovnávací paměti) v systému IBM i

Struktura definující volby obslužné rutiny vyrovnávací paměti pro zprávy.

Přehled

Účel: Struktura MQBMHO umožňuje aplikacím určit volby, které řídí způsob, jakým jsou zpracovávány manipulátory zpráv z vyrovnávacích pamětí. Struktura je vstupním parametrem volání MQBUFMH.

Znaková sada a kódování: Data v MQBMHO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1008
- "Počáteční hodnoty" na stránce 1009
- "Deklarace RPG" na stránce 1009

Pole

Struktura MQBMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

BMSID (10číslicové celé číslo se znaménkem)

Struktura vyrovnávací paměti pro zpracování zpráv-pole StrucId .

Jedná se o identifikátor struktury. Hodnota musí být:

BMSIDV
Identifikátor pro vyrovnávací paměť pro strukturu zpracování vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMSIDV.

BMVER (10ciferné celé číslo se znaménkem)

Buffer to message handle structure-Version field.

Jedná se o číslo verze struktury. Hodnota musí být:

BMVER1
Číslo verze pro vyrovnávací paměť pro strukturu vyrovnávací paměti zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

BMVERVC
Aktuální verze vyrovnávací paměti pro strukturu zpracování zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMVER1.

BMOPT (desetimístné podepsané celé číslo)

Buffer to message handle structure-Options field.

Hodnota může být následující:

BMDLPR.
Vlastnosti, které jsou přidány do popisovače zpráv, jsou z vyrovnávací paměti odstraněny. Pokud se nezdaří volání, nebudou odstraněny žádné vlastnosti.

Výchozí volby: Pokud nepotřebujete uvedenou volbu použít, použijte následující volbu:

BMNONE
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BMDLPR.

Počáteční hodnoty

Tabulka 684. Počáteční hodnoty polí 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 jednotky práce.

Přehled

Účel: Struktura je vstupním/výstupním parametrem pro volání MQBEGIN.

Znaková sada a kódování: Data ve znakové sadě MQBO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT.

- "Pole" na stránce 1009
- "Počáteční hodnoty" na stránce 1010
- "Deklarace RPG" na stránce 1010

Pole

Struktura MQBO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

BOOPT (10číslicové celé číslo se znaménkem)

Volby, které řídí akci MQBEGIN.

Hodnota musí být:

BONON
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BONONE.

BOSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

BOSIDV
Identifikátor pro strukturu begin-options.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BOSIDV.

BOVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

BOVER1
Číslo verze pro strukturu begin-options.

Následující konstanta uvádí číslo verze aktuální verze:

PŘETÍRAČ
Aktuální verze struktury begin-options.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je BOVER1.

Počáteční hodnoty

Tabulka 685. Počáteční hodnoty polí v MQBO

Název pole	Název konstanty	Hodnota konstanty
BOSID	BOSIDV	'BO¬¬'
BOVER	BOVER1	1
BOOPT	BONON	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) v systému IBM i

Struktura popisující rutinu zpětného volání.

Přehled

Účel
Struktura MQCBC se používá k určení informací o kontextu, které jsou předány funkci zpětného volání.

Struktura je vstupní/výstupní parametr na volání rutiny spotřebitele zprávy.

Verze
Aktuální verze MQCBC je CBCV2.

Znaková sada a kódování
Data v MQCBC se nacházejí ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Pokud je však aplikace spuštěna jako klient produktu IBM MQ , struktura se nachází ve znakové sadě a kódování klienta.

- "Pole" na stránce 1011
- "Počáteční hodnoty" na stránce 1016

- “Deklarace RPG” na stránce 1016

Pole

Struktura MQCBC obsahuje následující pole; pole jsou popsána v abecedním pořadí:

CBCBUFFLEN (10ciferné celé číslo se znaménkem)

Vyrovnávací paměť může být větší než hodnota délky MaxMsgdefinovaná pro spotřebitele a hodnota ReturnedLength v produktu MQGMO.

Kontextová struktura zpětného volání-pole BufferLength .

Toto je délka vyrovnávací paměti zpráv, která byla předána této funkci, v bajtech.

Skutečná délka zprávy se dodává v poli DataLength .

Aplikace může pro své vlastní účely použít celou vyrovnávací paměť po dobu trvání funkce zpětného volání.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny výjimky.

CBCCALLBA (Celé číslo s 10 číslicemi)

Kontextová struktura zpětného volání-pole CallbackArea .

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a v rámci struktury MQCBD je předávána v nezměněné podobě z pole CBDCALLBA , což je parametr volání MQCB použitý k definování funkce zpětného volání.

Změny v produktu *CBCCALLBA* jsou zachovány v rámci vyvolání funkce zpětného volání pro produkt *CBCHOBJ*. Toto pole není sdíleno s funkcemi zpětného volání pro jiné popisovače.

Jedná se o vstupní/výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CBCCALLT (10ciferné celé číslo se znaménkem)

Kontextová struktura zpětného volání-pole CallType .

Pole obsahující informace o tom, proč byla tato funkce volána. Jsou definovány následující typy volání.

Typy volání doručování zpráv: Tyto typy volání obsahují informace o zprávě. Parametry **CBCLEN** a **CBCBUFFLEN** jsou platné pro tyto typy volání.

CBCTMR
Funkce odběratele zpráv byla vyvolána se zprávou, která byla destruktivně odebrána z manipulátoru objektu.

Je-li hodnota *CBCCC* CCWARN, hodnota pole *Reason* je RC2079 nebo jeden z kódů označující problém konverze dat.

CBCTMN
Funkce odběratele zpráv byla vyvolána zprávou, která nebyla dosud destruktivně odebrána z manipulátoru objektu. Zpráva může být destruktivně odebrána z popisovače objektu pomocí *MsgToken*.

Je možné, že zpráva nebyla odebrána, protože:

- Volby MQGMO požádaly o operaci procházení, GMBR*
- Zpráva je větší než dostupná vyrovnávací paměť a volby MQGMO neurčují gmatm

Je-li hodnota *CBCCC* CCWARN, hodnota pole *Reason* je RC2080 nebo jeden z kódů označující problém konverze dat.

Typy volání ovládacího prvku zpětného volání: Tyto typy volání obsahují informace o kontrole zpětného volání a neobsahují podrobnosti o zprávě. Tyto typy volání jsou vyžádány pomocí CBDOPT ve struktuře MQCBD.

Parametry **CBCLEN** a **CBCBUFFLEN** nejsou platné pro tyto typy volání.

CBCTRC
Účelem tohoto typu volání je umožnit funkci zpětného volání, aby provedla nějaké 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 to požadováno, je to první vyvolání funkce zpětného volání.

Hodnota pole *CBCREA* je RCNONE.

CBCTSC
Účelem tohoto typu volání je povolit funkci zpětného volání, aby provedla určité nastavení při spuštění, například obnovení prostředků, které byly vyčištěny, když již bylo dříve zastaveno.

Funkce zpětného volání je vyvolána při spuštění připojení buď s parametrem CTRLR 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 povolit funkci zpětného volání, aby provedla určité vyčištění, když je například zastavena, například při čištění dalších prostředků, které byly získány během přijímání zpráv.

Funkce zpětného volání je vyvolána při vyvolání volání MQCTL s použitím hodnoty pole *Operation* v parametru CTLSP.

Tento typ volání se používá pouze pro spotřebitele zpráv.

Hodnota pole *CBCREA* je nastavena na indikování důvodu zastavení.

CBCTDC
Účelem tohoto typu volání je umožnit funkci zpětného volání, aby provedla závěrečný úklid na konci procesu spotřeby. Funkce zpětného volání je vyvolána, když:

- Funkce zpětného volání je deregistrovaná pomocí volání MQCB s BCUNR.
- Fronta je zavřena, což způsobí implicitní deregistraci. V této instanci je funkce zpětného volání předána HOUNUH jako manipulátor objektu.
- Volání MQDISC bylo dokončeno-způsobilo implicitní zavření a proto zrušení registrace. V tomto případě není připojení okamžitě odpojeno a žádná probíhající transakce nebyla dosud potvrzena.

Pokud se některá z těchto akcí provádí uvnitř samotné funkce zpětného volání, akce se vyvolá až po vrácení zpětného volání.

Tento typ volání se používá jak pro spotřebitele zpráv, tak pro obslužné rutiny událostí.

Je-li to požadováno, jedná se o poslední vyvolání funkce zpětného volání.

Hodnota pole *CBCREA* je nastavena na indikování důvodu zastavení.

CBCTEC
Funkce obslužné rutiny událostí

Funkce obslužné rutiny událostí byla vyvolána bez zprávy, když:

- Volání MQCTL je vydáno s hodnotou pro pole *Operation* v 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 odběratele zpráv**

  Funkce odběratele zpráv byla vyvolána bez zprávy, pokud byla zjištěna chyba (*CBCCC* = CCFAIL), která je specifická pro popisovač objektu; například kód *CBCREA* = RC2016 .

Hodnota pole *CBCREA* je nastavena na indikování důvodu pro volání.

Toto je vstupní pole. CBCTMR a CMCTMN lze použít pouze pro funkce spotřebitele zpráv.

CBCCC (10ciferné celé číslo se znaménkem)

Kontextová struktura zpětného volání-pole CompCode .

Toto je kód dokončení. Označuje, zda došlo k problémům při příjmu zprávy; jedná se o jednu z následujících možností:

KEK
Úspěšné dokončení

CCWARN
Varování (částečné dokončení)

CCFIL
Volání se nezdařilo

Toto je vstupní pole. Počáteční hodnota tohoto pole je CCOK.

CBCCONNAREA (Celé číslo se znaménkem 10 číslic)

Kontextová struktura zpětného volání-pole ConnectionArea .

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole ConnectionArea ve struktuře MQCTLO, což je parametr volání MQCTL, který slouží k řízení funkce zpětného volání.

Jakékoli změny provedené v tomto poli pomocí funkcí zpětného volání jsou zachovány v rámci vyvolání funkce zpětného volání. Tato oblast může být použita k předávání informací, které mají být sdíleny všemi funkcemi zpětného volání. Na rozdíl od *CallbackArea*je tato oblast společná pro všechna zpětná volání pro popisovač připojení.

Jedná se o vstupní a výstupní pole. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CBCLEN (10ciferné 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 nemusí nutně být délka dat zpráv předaných spotřebiteli. Může se stát, že zpráva byla zkrácena. Pole GMRL v produktu MQGMO použijte k určení toho, kolik dat bylo odběrateli předáno.

Pokud kód příčiny indikuje, že zpráva byla zkrácena, můžete použít pole CBCLEN k určení, jak velká je skutečná zpráva. To vám umožní určit velikost vyrovnávací paměti potřebné k umístění dat zprávy a pak vydat volání MQCB k aktualizaci CBDMML v MQCBD s příslušnou hodnotou.

Je-li uvedena volba GMCONV, převedená zpráva může být větší než vrácená hodnota pro DataLength. V takových případech bude pravděpodobně aplikace vyžadovat, aby volání MQCB aktualizovala CBDMML v MQCBD, aby byla vyšší než hodnota vrácená správcem front pro DataLength.

Chcete-li se vyhnout problémům s oříznutím zprávy, zadejte hodnotu MaxMsgLength jako CBDFM. To způsobí, že správce front alokuje vyrovnávací paměť pro úplnou délku zprávy po převodu dat. Uvědomte si však, že i když je tato volba zadána, je stále možné, že není k dispozici dostatek paměti pro správné zpracování požadavku. Aplikace by měly vždy kontrolovat návratový kód příčiny.

Není-li například možné přidělit dostatečnou paměť pro převod zprávy, budou zprávy vráceny do nepřevedené aplikace.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

CBCFLG (Celé číslo se znaménkem 10 číslic)

Příznaky obsahující informace o tomto odběrateli.

Je definována následující volba:

CBCFBE
Tento příznak může být vrácen v případě, že předchozí volání MQCLOSE používající volbu COQSC selhalo s kódem příčiny RC2458.

Tento kód indikoval, že je vrácena poslední zpráva dopředného čtení a že vyrovnávací paměť je nyní prázdná. Pokud aplikace vydá další volání MQCLOSE s použitím volby COQSC, uspěje.

Všimněte si, že aplikace není garantována, aby byla poskytnuta zpráva s touto sadou příznaků, protože stále mohou existovat zprávy v vyrovnávací paměti čtení napřed, které neodpovídají aktuálním kritériím výběru. V této instanci je funkce odběratele vyvolána s kódem příčiny RC2019 .

Je-li vyrovnávací paměť dopředného čtení prázdná, je spotřebitel vyvolán s příznakem CBCFBE a kódem příčiny RC2518.

Jedná se o vstupní pole pro funkci odběratele zpráv; není relevantní pro funkci obslužné rutiny událostí.

CBCHOBJ (10 číslic se znaménkem celé číslo)

Kontextová struktura zpětného volání-pole CBCHOBJ.

Pro volání spotřebitele zpráv se jedná o popisovač objektu vztahujícího se ke spotřebiteli zpráv.

Pro obslužnou rutinu událostí je tato hodnota PRÁZDNÁ

Aplikace může použít tento popisovač a token zprávy v bloku Volby načtení zprávy k získání zprávy, pokud zpráva nebyla z fronty odebrána.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je HOUNUH.

CBCRCD (10ciferné celé číslo se znaménkem)

CBCRCD udává, jak dlouho správce front čeká před pokusem o nové připojení. Toto pole může upravit obslužnou rutinou událostí, aby došlo ke změně prodlevy nebo zastavení opakovaného 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 parametru **CBCRCD** počet milisekund, po které bude správce front čekat, než provede pokus o opětovné připojení. V produktu Tabulka 686 na stránce 1014 jsou uvedeny hodnoty, které lze nastavit k úpravě chování správce front při návratu z obslužné rutiny událostí.

*Tabulka 686. **CBCRCD** hodnoty*

Hodnota	Popis
-1	Neprovádět další pokusy o opětovné připojení. Do aplikace se vrátí chyba.
0	Zkuste se okamžitě znovu připojit.
>0	Než se znovu pokusíte o připojení, počkejte tento počet milisekund.

CBCREA (10ciferné celé číslo se znaménkem)

Kontextová struktura zpětného volání-pole Příčina.

To je kód příčiny, který kvalifikují *CBCCC*

Toto je vstupní pole. Počáteční hodnota tohoto pole je RCNONE.

CBCSTATE (10ciferné celé číslo se znaménkem)

Indikace stavu aktuálního spotřebitele. Toto pole má největší hodnotu pro aplikaci, pokud je do funkce spotřebitele předán nenulový kód příčiny.

Toto pole můžete použít ke zjednodušení programování aplikací, protože pro každý kód příčiny není třeba chování kódu.

Toto je vstupní pole. Počáteční hodnota tohoto pole je CSNONE.

Tabulka 687. Hodnoty CBCSTATE a výsledné akce

Stav	**Akce správce front**	**Hodnota konstanty**
CSNONE Tento kód příčiny představuje běžné volání bez dalších informací o důvodu	Není; jedná se o normální operaci.	0
CSSUST Tyto kódy příčiny představují dočasné podmínky.	Rutina zpětného volání je volána k hlášení podmínky a poté pozastavena. Po určité době se systém může pokusit o provedení operace znovu, což může vést ke stejné podmínce, že bude znovu vyvolána podmínka.	1
CSSUSU Tyto kódy příčiny představují podmínky, za kterých zpětné volání potřebuje k vyřešení podmínky.	Spotřebitel je pozastaven a je volána rutina zpětného volání za účelem hlášení podmínky. Rutina zpětného volání by měla vyřešit podmínku, je-li to možné, a buď RESUME, nebo zavřít připojení.	2
CSSUS Tyto kódy příčiny představují selhání, která brání dalším zpětným voláním zpráv.	Správce front automaticky pozastaví funkci zpětného volání. Je-li funkce zpětného volání obnovena, je pravděpodobné, že bude znovu přijmout stejný kód příčiny.	3
CSSTOP Tyto kódy příčiny představují konec spotřeby zpráv.	Doručeno do obslužné rutiny výjimek a na zpětná volání, která byla určena CBDTC. Žádné další zprávy nelze spotřebovat.	4

CBCSID (celé číslo se znaménkem za 10 číslic)

Kontextová struktura zpětného volání-pole StrucId .

Jedná se o identifikátor struktury; hodnota musí být:

CBCSI
Identifikátor pro strukturu kontextu zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBCSI.

CBCVER (10ciferné celé číslo se znaménkem)

Kontextová struktura zpětného volání-pole Verze.

Jedná se o číslo verze struktury; hodnota musí být:

CBCV1
Version-1 -struktura kontextu zpětného volání.

Následující konstanta uvádí 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 688. Počáteční hodnoty polí v MQCBC

Název pole	Název konstanty	Hodnota konstanty
CBCSID	CBCSI	'CBC¬'
CBCVER	CBCV1	1
CBCCALLT	Není	0
CBCHOBJ	HOUUH	-1
CBCCALLBA	Není	Nulový ukazatel nebo bajty null
CBCCONNAREA	Není	Nulový ukazatel nebo bajty null
CBCCC	KEK	0
CBCREA	RCNONE	0
CBCSTATE	CSNANE	0
CBCLEN	Není	0
CBCBUFFLEN	Není	0
CBCFLG	Není	0
CBCRCD	žádný	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) v 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 vstupním parametrem 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; tyto údaje jsou dány atributem správce front **CodedCharSetId** a ENNAT. Je-li však aplikace spuštěna jako IBM MQ MQI client, musí být struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1017
- "Počáteční hodnoty" na stránce 1021
- "Deklarace RPG" na stránce 1021

Pole

Struktura MQCBD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CBDCALLBA (10ciferné celé číslo se znaménkem)

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole CBCCALLBA struktury MQCBD, což je parametr v deklaraci funkce zpětného volání.

Hodnota se používá pouze u *Operation* mající hodnotu CBREG, bez momentálně definovaného zpětného volání, nenahrazuje předchozí definici.

Jedná se o vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CBDCALLBF (10ciferné celé číslo se znaménkem)

Funkce zpětného volání je vyvolána jako volání funkce.

Toto pole slouží k zadání ukazatele na funkci zpětného volání.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Pokud uvedete obojí, vrátí se kód příčiny RC2486 .

Pokud ani *CallbackName* ani *CallbackFunction* není 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 na ukazatel funkce

V takových situacích volání selže s kódem příčiny RC2486.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

CBDCALLBN (10ciferné celé číslo se znaménkem)

Funkce zpětného volání je vyvolána jako dynamicky propojený program.

Musíte zadat buď *CallbackFunction* , nebo *CallbackName*. Pokud uvedete obojí, vrátí se kód příčiny RC2486 .

Pokud není hodnota *CallbackName* nebo *CallbackFunction* pravdivá, volání selže s kódem příčiny RC2486.

Modul je načten při registraci první rutiny zpětného volání a uvolnění při posledním volání rutiny zpětného volání, která má být použita pro zrušení registrace.

Není-li uvedeno v následujícím textu, jméno je zarovnáno vlevo uvnitř pole bez vložených mezer; jméno samotné je doplněno mezerami do délky pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

IBMi

Název zpětného volání může být jeden z následujících formátů:

- Knihovna "/" Program
- Knihovna "/" ServiceProgram "("FunctionName")"

Například `MyLibrary/MyProgram(MyFunction)`.

Název knihovny může být *LIBL. Názvy knihoven a programů jsou omezeny na maximálně 10 znaků.

AIX and Linux

Název zpětného volání je název dynamicky zaváděného modulu nebo knihovny s příponou s příponou s názvem funkce umístěné v dané knihovně. Název funkce musí být uzavřen do závorek. Název knihovny může být volitelně s předponou cesty k adresáři:

```
[path]library(function)
```

Není-li cesta zadána, použije se systémová cesta pro vyhledávání.

Název je omezen na maximálně 128 znaků.

Windows

Název zpětného volání je název knihovny s dynamicky propojovacím odkazem s příponou s názvem funkce umístěné v dané knihovně. Název funkce musí být uzavřen do závorek. Název knihovny může být volitelně uvozeno cestou k adresáři a jednotkou:

```
[d:][path]library(function)
```

Není-li cesta a cesta uvedena, použije se systémová cesta pro vyhledávání.

Název je omezen na maximálně 128 znaků.

z/OS

Název zpětného volání je název načítaného modulu, který je platný pro specifikaci v parametru EP makra LINK nebo LOAD.

Název je omezen na maximálně 8 znaků.

z/OS CICS
Název zpětného volání je název načítaného modulu, který je platný pro specifikaci v parametru PROGRAM příkazu EXEC CICS LINK.

Název je omezen na maximálně 8 znaků.

Program může být definován jako vzdálený pomocí volby REMOVESYTEM nainstalované definice PROGRAMU nebo pomocí dynamického programu směrování.

Vzdálená oblast CICS musí být připojena k serveru IBM MQ , pokud má program používat volání rozhraní API produktu IBM MQ . Všimněte si však, že pole CBCHOBJ ve struktuře MQCBC není platné ve vzdáleném systému.

Dojde-li k selhání při pokusu o načtení *CallbackName*, vrátí se do aplikace jeden z následujících kódů chyby:

- RC2495
- RC2496
- RC2497

Do protokolu chyb se zapíše také zpráva obsahující název modulu, pro který byl pokus o načtení proveden, a kód příčiny selhání z operačního systému.

Toto je vstupní pole. Počáteční hodnota tohoto pole je prázdný řetězec nebo prázdný řetězec.

CBDCALLBT (10ciferné 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 se volá tehdy, je-li zpráva splňující zadaná kritéria výběru k dispozici na manipulátoru objektu a připojení je spuštěno.

CBTEH
Definuje toto zpětné volání jako rutinu asynchronních událostí; neřídí se spotřebovávat zprávy pro manipulátor.

Příkaz *Hobj* není vyžadován při volání MQCB, který definuje obslužnou rutinu událostí, a je-li zadán, je ignorován.

Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí spotřebitele zpráv. Funkce odběratele je vyvolána bez zprávy při výskytu události, například zastavení správce front nebo zastavení připojení nebo uvedení do klidového stavu. Nevolá se pro podmínky, které jsou specifické pro jednotlivého spotřebitele zpráv, například RC2016.

Události jsou doručovány 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 (10ciferné celé číslo se znaménkem)

Toto je délka v bajtech nejdelší zprávy, kterou lze přečíst z popisovače a poskytnuta pro rutinu zpětného volání. Má-li zpráva delší délku, přijímá rutina zpětného volání *MaxMsgLength* bajtů zprávy a kód příčiny:

- RC2080 nebo
- RC2079 , pokud jste uvedli GMATM.

Skutečná délka zprávy je dodána v poli “CBCLEN (10ciferné celé číslo se znaménkem)” na stránce 1013 struktury MQCBC.

Je definována následující speciální hodnota:

CBDFM
Délka vyrovnávací paměti je přizpůsobena systémem pro vrácení zpráv bez oříznutí.

Je-li k dispozici dostatek paměti pro přidělení vyrovnávací paměti k 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, nekonvertované zprávy se předají do funkce zpětného volání.

Toto je vstupní pole. Počáteční hodnota pole *MaxMsgLength* je CBDFM.

CBDOPT (10ciferné celé číslo)

Struktura deskriptoru zpětného volání-Pole Volby.

Lze zadat libovolný, nebo všechny následující hodnoty. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace, které nejsou platné, jsou zaznamenány; jakékoli jiné kombinace jsou platné.

CBDFQ
Volání MQCB selže, 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) ve stavu uvedení do klidového stavu.

Zadejte GMFIQ, v rámci voleb MQGMO předaných volání MQCB, abyste způsobovali oznámení spotřebitelům zpráv, když jsou uvádění do klidového stavu.

Volby ovládacího prvku: Následující volby řídí, zda je funkce zpětného volání volána bez zprávy, když se změní stav spotřebitele:

CBDRT
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.

Viz “CBCCALLT (10ciferné celé číslo se znaménkem)” na stránce 1011 , kde získáte další podrobnosti o těchto typech volání.

Výchozí volba: Pokud nepotřebujete žádné 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 definováno v dokumentaci programu pomoci; není zamýšleno, aby tato volba byla použita s jinou, ale jako její hodnota je nula, takové použití nelze detekovat.

Toto je vstupní pole. Počáteční hodnota pole *Options* je CBDNO.

CBDSID (10ciferné celé číslo se znaménkem)

Struktura deskriptoru zpětného volání-pole StrucId .

Jedná se o identifikátor struktury; hodnota musí být:

CBDSI
Identifikátor pro strukturu deskriptoru zpětného volání.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CBDSI.

CBDVER (10ciferné celé číslo se znaménkem)

Struktura deskriptoru zpětného volání-pole Verze.

Jedná se o číslo verze struktury; hodnota musí být:

CBDV1
Struktura deskriptoru zpětného volání Version-1 .

Následující konstanta uvádí 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 689. Počáteční hodnoty polí v MQCBD

Název pole	Název konstanty	Hodnota konstanty
StrucId	CBDSI	'CBD¬'
Version	CBDV1	1
CallBackType	CBTMC	1
Options	CBDNO	0
CallbackArea	Není	Nedefinované bajty
CallbackFunction	Není	Nedefinované bajty
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 IBM i

Strukturu MQCHARV použijte k popisu řetězce proměnné délky.

Přehled

Znaková sada a kódování: Data ve struktuře MQCHARV musí být v kódování lokálního správce front, který je poskytnut ENNAT a znaková sada pole VCHRC v rámci struktury. Je-li aplikace spouštěna jako IBM MQ MQI client, musí se struktura nacházet v kódování klienta. Některé znakové sady mají reprezentaci, která závisí na daném kódování. Je-li VCHRC jedna z těchto znakových sad, použité kódování je stejné kódování jako u ostatních polí ve struktuře MQCHARV. Znaková sada identifikovaná VSCCSID může být dvoubajtová znaková sada (DBCS).

Použití: Struktura MQCHARV adresuje data, která mohou být nesousedící se strukturou obsahující tuto strukturu. Chcete-li adresovat tato data, lze použít pole deklarovaná s datovým typem ukazatele.

- "Pole" na stránce 1022
- "Počáteční hodnoty" na stránce 1023
- "Deklarace RPG" na stránce 1023
- "Předefinování CSAPL" na stránce 1024

Pole

Struktura MQCHARV obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

VCHRC (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor znakové sady proměnné délky proměnné adresované polem VCHRP nebo VCHRO.

Počáteční hodnota tohoto pole je CSAPL. Tato hodnota je definována argumentem IBM MQ , aby indikovala, že správce front by měl být změněn na identifikátor skutečné znakové sady správce front. To je stejné, jako se CSQM chová. V důsledku toho není hodnota CSAPL nikdy přidružena k řetězci s proměnnou délkou. Počáteční hodnotu tohoto pole lze změnit definováním odlišné hodnoty pro konstantu CSAPL pro danou kompilační jednotku vhodnými prostředky pro programovací jazyk vaší aplikace.

VCHRL (10ciferné celé číslo se znaménkem)

Délka řetězce proměnné délky adresovaná polem VCHRP nebo VCHRO v bajtech.

Počáteční hodnota tohoto pole je 0. Hodnota musí být buď větší než nebo rovna nule, nebo následující speciální hodnotu, která je rozeznána:

VNLT

Není-li VSNLT uvedeno, VCHRL bajty jsou zahrnuty jako část řetězce. Pokud jsou přítomny znaky null, neoddělují řetězec.

Je-li uvedeno VSNLT, řetězec je oddělen první hodnotou null zjištěnou v řetězci. Samotná hodnota null není zahrnuta jako součást tohoto řetězce.

Poznámka: Nulový znak použitý k ukončení řetězce, je-li VSNLT zadán, je hodnota null z kódové sady uvedené VCHRC.

Například v UTF-16 (CCSID 1200, 13488 a 17584) se jedná o 2bajtové kódování Unicode, kde hodnota null je představována 16bitovým číslem všech nul. V UTF-16 je běžné najít jednotlivé bajty nastavené na všechny nuly, které jsou součástí znaků (například 7-bitové ASCII znaky), ale

řetězce budou ukončeny pouze tehdy, když se dva 'nula' bajtů nachází na rovnoměrné hranici bajtů. Je možné získat dva 'nula' bajtů na liché hranici, když jsou každá část platných znaků. Například x '01' x '00' x '00' x '30' představuje dva platné znaky Unicode a tento řetězec neukončí null.

VCHRO (10ciferné celé číslo se znaménkem)

Posunutí v bajtech proměnné délky proměnné od začátku MQCHARV nebo do struktury obsahující tento řetězec.

Je-li struktura MQCHARV vložena do jiné struktury, bude tato hodnota posunem v bajtech proměnné délky proměnné od začátku struktury, která obsahuje tuto strukturu MQCHARV. Není-li struktura MQCHARV vložena do jiné struktury, například pokud je zadána jako parametr ve volání funkce, posunutí je relativní vzhledem ke spuštění struktury MQCHARV.

Odsazení může být kladné nebo záporné. 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 null bajtů.

VCHRS (10ciferné celé číslo se znaménkem)

Velikost vyrovnávací paměti adresovaná polem VCHRP nebo VCHRO v bajtech.

Je-li struktura MQCHARV použita jako výstupní pole ve funkci volání funkce, toto pole musí být inicializováno s délkou poskytnuté vyrovnávací paměti. Je-li hodnota VCHRL větší než VCHRS, vrátí se volajícímu do vyrovnávací paměti pouze bajty VCHRS dat.

Hodnota musí být větší než nula nebo rovna nule nebo následující speciální hodnotu, která je rozpoznána:

VSUS1
Je-li uvedeno VSUSL, délka vyrovnávací paměti se vezme z pole VCHRL ve struktuře MQCHARV. Tato speciální hodnota není vhodná, když se struktura používá jako výstupní pole a je poskytnuta vyrovnávací paměť. Toto je počáteční hodnota tohoto pole.

Počáteční hodnoty

Tabulka 690. Počáteční hodnoty MQCHARV pro konstanty

Název pole	Název konstanty	Hodnota konstanty
VCHRP	Není	Nulový ukazatel nebo bajty null.
VCHRO	Není	0
VCHRS	VSUS1	-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
```

Předefinování CSAPL

Na rozdíl od programovacích jazyků podporovaných na jiných platformách RPG nemá způsob předefinování definované konstanty, takže musíte každý VCHRC nastavit speciálně v případě, že chcete použít jinou hodnotu než CSAPL.

IBM i MQCIH (záhlavíCICS bridge) v systému IBM i

Struktura MQCIH popisuje informace, které mohou být přítomny na začátku zprávy odeslané do produktu CICS bridge prostřednictvím produktu IBM MQ for z/OS.

Přehled

Název formátu: FMCICS.

Verze: Aktuální verze MQCIH je CIVER2. Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQCIH, přičemž počáteční hodnota pole *CIVER* je nastavena na CIVER2.

Znaková sada a kódování: Speciální podmínky platí pro znakovou sadu a kódování použité pro strukturu MQCIH a data zprávy aplikace:

- Aplikace, které se připojují ke správci front, který je vlastníkem fronty produktu CICS bridge , musí poskytovat strukturu MQCIH, která se nachází ve znakové sadě a kódování správce front. Důvodem je, že převod dat struktury MQCIH se v tomto případě neprovádí.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQCIH, která se nachází ve všech podporovaných znakových sadách a kódováních; konverze MQCIH je prováděna přijímajícím agentem kanálu zpráv připojeným ke správci front, který vlastní frontu CICS bridge .

  Poznámka: Existuje jedna výjimka. Pokud správce front, který je vlastníkem fronty CICS bridge , používá CICS pro distribuované ukládání do fronty, musí být MQCIH ve znakové sadě a v kódování správce front, který vlastní frontu CICS bridge .
- Data zprávy aplikace následující za strukturou MQCIH musí být ve stejné znakové sadě a kódování jako struktura MQCIH. Pole *CICSI* a *CIENC* ve struktuře MQCIH nelze použít k uvedení znakové sady a kódování dat zprávy aplikace.

  Uživatelská procedura pro převod dat musí být poskytnuta uživatelem za účelem převodu dat zprávy aplikace, nejsou-li data jedním z vestavěných formátů podporovaných správcem front.

Použití: Pokud jsou hodnoty požadované aplikací stejné jako počáteční hodnoty zobrazené v produktu Tabulka 692 na stránce 1033a je-li most spuštěn s parametrem AUTH=LOCAL nebo AUTH=IDENTIFY, může být struktura MQCIH vynechána ze zprávy. 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 "request" měla odpovídající hodnoty ve zprávě odeslané do mostu; tato pole jsou vstupem do mostu.

Pole dokumentovaná jako pole "odezva" jsou nastavena CICS bridge ve zprávě odpovědi, kterou most odesílá do aplikace. Informace o chybě jsou vráceny v polích *CIRET*, *CIFNC*, *CICC*, *CIREA*a *CIAC* , ale

ne všechny jsou nastaveny ve všech případech. Tabulka 691 na stránce 1025 ukazuje, která pole jsou nastavena pro různé hodnoty *CIRET*.

Tabulka 691. Obsah polí s informacemi o chybě ve struktuře MQCIH

CIRET	***CIFNC***	***CICC***	***CIREA***	***CIAC***
CRC000	-	-	-	-
CRC003	-	-	FBC *	-
CRC002 CRC008	Název volání produktu IBM MQ	IBM MQ *CMPCOD*	IBM MQ *REASON*	-
CRC001 CRC006 CRC007 CRC009	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	-
CRC004 CRC005	-	-	-	CICS ABCODE

- "Pole" na stránce 1025
- "Počáteční hodnoty" na stránce 1033
- "Deklarace RPG" na stránce 1035

Pole

Struktura MQCIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CIAC (čtyřbajtový znakový řetězec)

Kód nestandardního konce.

Hodnota vrácená v tomto poli je významná pouze v případě, že pole *CIRET* má hodnotu CRC005 nebo CRC004. Pokud ano, *CIAC* obsahuje hodnotu ABCODE CICS .

Toto je pole odezvy. Délka tohoto pole je dána LNABNC. Počáteční hodnota tohoto pole je 4 prázdné znaky.

Jedná se o indikátor, který určuje, zda mají být při příkazu SEND a RECEIVE BMS odesílány deskriptory ADS. Jsou definovány tyto hodnoty:

ADNONE
Neodesílat nebo přijímat deskriptor ADS.

ADSEND
Odeslat deskriptor souboru ADS.

ADRECV
Přijmout deskriptor ADS.

ADMSGF
Použijte formát zpráv pro deskriptor ADS.

To způsobí odeslání nebo přijetí deskriptoru ADS pomocí dlouhé formy deskriptoru ADS. Dlouhá forma má pole, která jsou zarovnána na 4bajtové hranice.

Pole *CIADS* by mělo být nastaveno takto:

- Pokud nejsou použity deskriptory ADS, nastavte pole na ADNONE.
- Pokud jsou použity deskriptory ADS *jsou* a s *stejným* CCSID v každém prostředí, nastavte pole na součet ADSEND a ADRECV.
- Pokud jsou použity deskriptory ADS *jsou* použity, ale s *odlišnými* CCSID v každém prostředí, nastavte pole na součet ADSEND, ADRECV a ADMSGF.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je ADNONE.

CIADS (10ciferné celé číslo se znaménkem)

Odeslání/přijetí deskriptoru ADS.

Jedná se o indikátor, který určuje, zda mají být při příkazu SEND a RECEIVE BMS odesílány deskriptory ADS. Jsou definovány tyto hodnoty:

ADNONE
Neodesílat nebo přijímat deskriptor ADS.

ADSEND
Odeslat deskriptor souboru ADS.

ADRECV
Přijmout deskriptor ADS.

ADMSGF
Použijte formát zpráv pro deskriptor ADS.

To způsobí odeslání nebo přijetí deskriptoru ADS pomocí dlouhé formy deskriptoru ADS. Dlouhá forma má pole, která jsou zarovnána na 4bajtové hranice.

Pole *CIADS* by mělo být nastaveno takto:

- Pokud nejsou použity deskriptory ADS, nastavte pole na ADNONE.
- Pokud jsou použity deskriptory ADS *jsou* a s *stejným* CCSID v každém prostředí, nastavte pole na součet ADSEND a ADRECV.
- Pokud jsou použity deskriptory ADS *jsou* použity, ale s *odlišnými* CCSID v každém prostředí, nastavte pole na součet ADSEND, ADRECV a ADMSGF.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je ADNONE.

CIAI (čtyřbajtový znakový řetězec)

Klíč AID.

Jedná se o počáteční hodnotu klíče AID, když je transakce spuštěna. Jedná se o 1bajtovou hodnotu, zarovnanou doleva.

Toto je pole požadavku použité pouze pro transakce 3270. Délka tohoto pole je dána LNATID. Počáteční hodnota tohoto pole je 4 mezery.

CIAUT (8bajtový znakový řetězec)

Heslo nebo přístupový lístek.

Toto je heslo nebo přístupový lístek. Pokud je ověření identifikátoru uživatele aktivní pro CICS bridge, použije se *CIAUT* spolu s identifikátorem uživatele v kontextu identity MQMD k ověření odesílatele zprávy.

Toto je pole požadavku. Délka tohoto pole je dána LNAUTH. Počáteční hodnota tohoto pole je 8 mezer.

CICC (10číslicové celé číslo se znaménkem)

IBM MQ kód dokončení nebo CICS EIBRESP.

Hodnota vrácená v tomto poli je závislá na *CIRET* ; viz Tabulka 691 na stránce 1025.

Toto je pole odezvy. Počáteční hodnota tohoto pole je CCOK.

CICNC (čtyřbajtový znakový řetězec)

Nekonečný kód transakce.

Jedná se o kód nestandardního ukončení, který má být použit k ukončení transakce (obvykle konverzační transakce, která požaduje více dat). Jinak je toto pole nastaveno na mezery.

Toto je pole požadavku použité pouze pro transakce 3270. Délka tohoto pole je dána LNCNCL. Počáteční hodnota tohoto pole je 4 mezery.

CICP (10ciferné celé číslo se znaménkem)

Pozice kurzoru.

Jedná se o počáteční pozici kurzoru při spuštění transakce. Později, v případě konverzačních transakcí, je pozice kurzoru v vektoru PŘÍJMU.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud *CIVER* je menší než CIVER2.

CICSI (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

CICT (10ciferné celé číslo se znaménkem)

Určuje, zda úloha může být konverzační.

Jedná se o indikátor, který uvádí, zda by měla být úloha povolena vydávat požadavky pro více informací, nebo by měla skončit. Hodnota musí být jedna z následujících:

TYKY
Úloha je dialogová.

CTNO
Úloha není dialogová.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je CTNO.

CIENC (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

CIEO (10ciferné 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 do umístění neplatných dat.

Jedná se o pole odezvy použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0. Toto pole není přítomno, pokud *CIVER* je menší než CIVER2.

CIFAC (8bajtový bitový řetězec)

Token zařízení mostu.

Jedná se o 8bajtový token funkce mostu. Účelem tokenu prostředku mostu je povolit více transakcí v pseudokonverzaci pro použití stejné funkce mostu (virtuální terminál 3270). V první nebo jediné zprávě v pseudokonverzaci by měla být nastavena hodnota FCNONE; ta sděluje příkazu CICS , aby přidělil novou funkci mostu pro tuto zprávu. Token prostředku mostu je vrácen ve zprávách odezvy, je-li na vstupní zprávě uveden nenulový *CIFKT* . Následné vstupní zprávy pak mohou používat stejný token prostředku mostu.

Je definována následující speciální hodnota:

FCNONE
Nebyl zadán token BVT.

Jedná se o pole požadavku i pole odezvy použité pouze pro transakce 3270. Délka tohoto pole je dána LNFAC. Počáteční hodnota tohoto pole je FCNONE.

CIFKT (10ciferné celé číslo se znaménkem)

Uvolňovací čas zařízení mostu.

Jedná se o dobu v sekundách, po kterou bude funkce mostu uchována po ukončení uživatelské transakce. U nekonverzačních transakcí by hodnota měla být nula.

Toto je pole požadavku použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je 0.

CIFL (čtyřbajtový znakový řetězec)

Terminal emulované atributy.

Jedná se o název instalovaného terminálu, který má být použit jako model pro zařízení mostu. Hodnota mezer znamená, že produkt *CIFL* je převzat z definice profilu transakce mostu, nebo se použije výchozí hodnota.

Toto je pole požadavku použité pouze pro transakce 3270. Délka tohoto pole je dána LNFACL. Počáteční hodnota tohoto pole je 4 mezery.

CIFLG (Celé číslo se 10 číslicemi)

Příznaky.

Hodnota musí být:

CIFNON
Žádné vlajky.

Toto je pole požadavku. Počáteční hodnota tohoto pole je CIFNON.

CIFMT (8bajtový znakový řetězec)

Název formátu produktu IBM MQ pro data následující MQCIH.

Určuje název formátu produktu IBM MQ pro data, která následují za strukturou MQCIH.

Na základě 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.

Tento název formátu se také používá pro zprávu odpovědi, pokud má pole *CIRFM* hodnotu FMNONE.

- V případě požadavků DPL musí být *CIFMT* název formátu COMMAREA.
- Pro požadavky 3270 musí být *CIFMT* CSQCBDCIa *CIRFM* musí být CSQCBDCO.

Uživatelské procedury pro převod dat pro tyto formáty musí být instalovány ve správci front, ve kterém mají být spuštěny.

Pokud zpráva požadavku vede k vygenerování zprávy s odpovědí na chybu, má zpráva s chybovou zprávou 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 funkce CICS EIBFN.

Hodnota vrácená v tomto poli je závislá na *CIRET* ; viz Tabulka 691 na stránce 1025. Následující hodnoty jsou možné, když *CIFNC* obsahuje název volání IBM MQ :

MFCONN
Volání MQCONN.

CFGET
Volání MQGET.

CFINQNAME
Volání MQINQ.

CFOPEN
Volání MQOPEN.

CFPUT
Volání MQPUT.

CFPUT1
Volání MQPUT1 .

CFNONE
Žádné telefonát.

Toto je pole odezvy. Délka tohoto pole je dána LNFUNC. Počáteční hodnota tohoto pole je CFNONE.

CIGWI (10číslicové podepsané celé číslo)

Interval čekání na volání MQGET vydaný úlohou mostu.

Toto pole lze použít pouze v případě, že *CIUOW* má hodnotu CUFRST. Umožňuje odesílající aplikaci určit přibližnou dobu v milisekundách, po kterou by volání MQGET vydaná mostem měla čekat na druhé a následné zprávy požadavků pro jednotku práce spuštěnou touto zprávou. Tento parametr přepíše výchozí čekací interval použitý mostem. Mohou být použity následující speciální hodnoty:

WIDFTA
Předvolený interval čekání.

To způsobí, že CICS bridge čeká po dobu, kdy byl spuštěn most.

WIULIM
Neomezený interval čekání.

Toto je pole požadavku. Počáteční hodnota tohoto pole je WIDFLT.

CIII (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 0. Toto pole není přítomno, pokud *CIVER* je menší než CIVER2.

CILEN (10ciferné celé číslo se znaménkem)

Délka struktury MQCIH.

Hodnota musí být jedna z následujících:

CILEN1
Délka struktury záhlaví informačního obsahu version-1 CICS .

CILEN2
Délka struktury záhlaví informačního obsahu version-2 CICS .

Následující konstanta uvádí délku aktuální verze:

CILENC
Délka aktuální verze struktury záhlaví informačního obsahu produktu CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CILEN2.

CILT (10ciferné celé číslo se znaménkem)

Typ odkazu.

Označuje typ objektu, který by most měl zkusit propojit. Hodnota musí být jedna z následujících:

LTPROG
Program DPL.

LTTRAN
transakce 3270.

Toto je pole požadavku. Počáteční hodnota tohoto pole je LTPROG.

CINTI (čtyřbajtový znakový řetězec)

Další transakce k připojení.

Jedná se o název další transakce vrácené uživatelskou transakcí (obvykle EXEC CICS RETURN TRANSID). Pokud žádná další transakce neexistuje, je toto pole nastaveno na mezery.

Jedná se o pole odezvy použité pouze pro transakce 3270. Délka tohoto pole je dána LNTRID. Počáteční hodnota tohoto pole je 4 mezery.

CIODL (10ciferné číslicové celé číslo)

Výstupní délka dat COMMAREA.

Jedná se o délku uživatelských dat, která má být vrácena klientovi ve zprávě s odpovědí. Tato délka zahrnuje 8bajtový název programu. Délka oblasti COMMAREA předaná k propojenému programu je maximum tohoto pole a délka uživatelských dat ve zprávě požadavku minus 8.

Poznámka: Délka uživatelských dat ve zprávě je délka zprávy *vyloučení* struktury MQCIH.

Je-li délka uživatelských dat ve zprávě požadavku menší než hodnota *CIODL*, použije se volba DATALENGTH příkazu LINK ; to umožňuje efektivní funkci LINK v jiném regionu CICS .

Je možné použít následující speciální hodnotu:

OLINT
Délka výstupu je stejná jako vstupní délka.

Tato hodnota může být potřebná i v případě, že není požadována žádná odpověď, aby se zajistilo, že COMMAREA předaná do propojeného programu má dostatečnou velikost.

Toto je pole požadavku použité pouze pro programy DPL. Počáteční hodnota tohoto pole OLINPT.

CREA (10ciferné celé číslo se znaménkem)

IBM MQ důvod nebo zpětnovazební kód, nebo CICS EIBRESP2.

Hodnota vrácená v tomto poli je závislá na *CIRET* ; viz Tabulka 691 na stránce 1025.

Toto je pole odezvy. Počáteční hodnota tohoto pole je RCNONE.

CIRET (Celé číslo se znaménkem pod 10 číslicemi)

Návratový kód z mostu.

Jedná se o návratový kód z CICS bridge popisující výsledek zpracování prováděného mostem. Pole *CIFNC*, *CICC*, *CIREA*a *CIAC* mohou obsahovat další informace (viz Tabulka 691 na stránce 1025). Hodnota je jedna z následujících možností:

CRC000
(0, X'000 ') Bez chyby.

CRC001
(1, X'001 ') EXEC CICS detekovala chybu.

CRC002
(2, X'002 ') Volání IBM MQ detekovalo chybu.

CRC003
(3, X'003 ') CICS bridge detekoval chybu.

CRC004
(4, X'004 ') CICS bridge abnormálně skončil.

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 jednotky práce, která nebyla přijata ve stanoveném čase.

CRC009
(9, X'009 ') Transakce není k dispozici.

Toto je pole odezvy. Počáteční hodnota tohoto pole je CRC000.

CIRFM (8bajtový znakový řetězec)

Název formátu IBM MQ zprávy odpovědi.

Jedná se o název formátu IBM MQ zprávy odpovědi, který bude odeslán jako odpověď na aktuální zprávu. Pravidla pro kódování jsou stejná jako pravidla pro pole *MDFMT* v produktu MQMD.

Toto je pole požadavku použité pouze pro programy DPL. Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

CIRSI (čtyřbajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 4 mezery. Délka tohoto pole je dána LNRSID.

CIRS1 (8bajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 8 mezer.

CIRS2 (8bajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 8 mezer.

CIRS3 (8bajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 8 mezer.

CIRS4 (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 0. Toto pole není přítomno, pokud *CIVER* je menší než CIVER2.

CIRTI (čtyřbajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Hodnota musí být 4 mezery. Délka tohoto pole je dána LNTRID.

CISC (4bajtový znakový řetězec)

Počáteční kód transakce.

Jedná se o indikátor určující, zda most emuluje transakci terminálu nebo transakci START. Hodnota musí být jedna z následujících:

SSTRANÍ
Spustit.

DATA ZOBRAZENÍ
Spustit data.

POSUNUTÍ
Ukončete vstup.

SCNONE
Není.

V odpovědi na můstek je toto pole nastaveno na počáteční kód odpovídající dalšímu ID transakce, které je obsaženo v poli *CINTI* . V odezvě jsou možné následující spouštěcí kódy:

- SSTRANÍ
- DATA ZOBRAZENÍ
- POSUNUTÍ

Pro CICS Transaction Server 1.2 toto pole je pouze pole požadavku; jeho hodnota v odpovědi není definována.

Pro produkt CICS Transaction Server 1.3 a následující 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 (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

CIDV
Identifikátor pro strukturu záhlaví informací produktu CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CISIDV.

CITES (10ciferné celé číslo se znaménkem)

Stav na konci úlohy.

Toto pole zobrazuje stav transakce uživatele na konci úlohy. Je vrácena jedna z následujících hodnot:

TENOSKY
Nesynchronizováno.

Transakce uživatele nebyla dosud dokončena a nebyla synchronizovaná. Pole *MDMT* v MQMD je MTRQST v tomto případě.

TECMIT
Potvrdit jednotku práce.

Transakce uživatele se ještě nedokončila, ale syncpointa první transakce byla synchronizována. Pole *MDMT* v MQMD je MTDGRM v tomto případě.

ODTRŽENÍ
Zazálohujte jednotku práce.

Transakce uživatele nebyla dosud dokončena. Aktuální jednotka práce bude vrácena. Pole *MDMT* v MQMD je MTDGRM v tomto případě.

TEENDT
Ukončit úlohu.

Transakce uživatele byla ukončena (nebo ukončena). Pole *MDMT* v MQMD je MTRPLY v tomto případě.

Jedná se o pole odezvy použité pouze pro transakce 3270. Počáteční hodnota tohoto pole je TENOSY.

CITI (čtyřbajtový znakový řetězec)

Transakce pro připojení.

Má-li *CILT* hodnotu LTTRAN, *CITI* je identifikátor transakce uživatelské transakce, která se má spustit; v tomto případě musí být zadána neprázdná hodnota.

Má-li *CILT* hodnotu LTPROG, *CITI* je kód transakce, pod kterým mají být spuštěny všechny programy v jednotce práce. Je-li uvedená hodnota prázdná, použije se výchozí kód transakce mostu CICS DPL (CKBP). Pokud je hodnota neprázdná, musela být definována na CICS jako lokální TRANSACTION s výchozím programem CSQCBP00. Toto pole lze použít pouze v případě, že *CIUOW* má hodnotu CUFRST nebo CUONLY.

Toto je pole požadavku. Délka tohoto pole je dána LNTRID. Počáteční hodnota tohoto pole je 4 mezery.

CIRUOW (10ciferné celé číslo se znaménkem)

Ovládací prvek z pracovní jednotky.

Tím se řídí zpracování jednotky práce prováděné serverem CICS bridge. Můžete požadovat, aby most spustil jednu transakci, nebo jeden nebo více programů v rámci transakce. Pole uvádí, zda by měl produkt CICS bridge spustit jednotku práce, provést požadovanou funkci v rámci aktuální jednotky práce nebo ukončit jednotku práce tím, že ji potvrdí nebo ji zálohuje. Jsou podporovány různé kombinace, aby se optimalizovalo toky přenosu dat.

Hodnota musí být jedna z následujících:

POUZE KUSY
Spuštění pracovní jednotky, provedení funkce, následné potvrzení jednotky práce (DPL a 3270).

VYJMOUTÝ
Další data pro aktuální jednotku práce (pouze 3270).

CUFRST
Spuštění jednotky práce a provedení funkce (pouze DPL).

KUMÍK
Provést funkci v rámci aktuální jednotky práce (pouze DPL).

PĚSTOVÁNÍ
Provést funkci, pak potvrdit jednotku práce (pouze DPL).

VYKUMOVÁNÍ
Potvrdit jednotku práce (pouze DPL).

ODŘEZLENÉ
Zálohovat pouze jednotku práce (pouze DPL).

Toto je pole požadavku. Počáteční hodnota tohoto pole je POUZE JEDINÁ.

CIVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

CIVER1
Struktura informačního záhlaví Version-1 CICS .

CIVER2
Struktura informačního záhlaví Version-2 CICS .

Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

CIVERC
Aktuální verze struktury záhlaví informací produktu CICS .

Toto je pole požadavku. Počáteční hodnota tohoto pole je CIVER2.

Počáteční hodnoty

Tabulka 692. Počáteční hodnoty polí v MQCIH

Název pole	Název konstanty	Hodnota konstanty
CISID	CIDV	'CIH¬'
CIVER	CIVER2	2
CILEN	CILEN2	180
CIENC	Není	0
CICSI	Není	0
CIFMT	FMNONE	Mezery
CIFLG	CIFNON	0

Tabulka 692. Počáteční hodnoty polí v MQCIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
CIRET	CRC000	0
CICC	KEK	0
CIREA	RCNONE	0
CIUOW	POUZE KUSY	273
CIGWI	WIDFTA	-2
CILT	LTPROG	1
CIODL	OLINT	-1
CIFKT	Není	0
CIADS	ADNONE	0
CICT	CTNO	0
CITES	TENOSKY	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
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 zpracování zpráv) v systému IBM i

Struktura **MQCMHO** umožňuje aplikacím určovat volby, které řídí způsob vytváření obslužných rutin zpráv.

Přehled

Účel
Struktura je vstupním parametrem na volání **MQCRTMH** .

Znaková sada a kódování
Data v souboru **MQCMHO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1036
- "Počáteční hodnoty" na stránce 1037
- "Deklarace RPG" na stránce 1038

Pole

Struktura MQCMHO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

CMOPT (10ciferné celé číslo se znaménkem)

Je možné zadat jednu z následujících možností:

CMVAL

Je-li volána funkce **MQSETMP** k nastavení vlastnosti v tomto popisovači zprávy, je název vlastnosti ověřen, aby bylo zajištěno, že:

- neobsahuje neplatné znaky.
- nezačíná "JMS" nebo "usr.JMS" s výjimkou následujících:
  - JMSCorrelationID
  - JMSReplyTo
  - JMSType.
  - JMSXGroupID
  - JMSXGroupSeq

  Tyto názvy jsou vyhrazeny pro vlastnosti produktu JMS .
- není jedním z následujících klíčových slov, v libovolné směsi malých nebo velkých písmen:
  - "A"
  - "MEZI"
  - "ESCAPE"
  - "NEPRAVDA"
  - "V"
  - "JE"
  - "JAKO"
  - "NE"
  - "NULL"
  - "NEBO"
  - "TRUE"
- nezačíná "Tělo." nebo "Root." (kromě "Root.MQMD.").

Je-li vlastnost definovaná v produktu MQ("mq. *") a název je rozpoznán, pole deskriptoru vlastností jsou nastavena na správné hodnoty pro vlastnost. Není-li vlastnost rozpoznána, je pole *Support* deskriptoru vlastností nastaveno na **PDSUPO** (další informace viz PDSUP).

CMDEFV

Tato hodnota určuje, že dojde k výchozí úrovni ověřování názvů vlastností.

Výchozí úroveň ověření je stejná jako úroveň, která je určena parametrem **CMVAL**.

V budoucím vydání může být definována administrativní volba, která změní úroveň ověření platnosti, k níž dojde, když je **CMDEFV** definován.

Toto je výchozí hodnota.

CMNOVA

Nedojde k ověření platnosti názvu vlastnosti. Viz popis **CMVAL**.

Výchozí volba: Není-li požadována žádná z voleb dříve popsaných v této sekci, lze použít následující volbu:

CMNONE

Všechny volby předpokládají jejich výchozí hodnoty. Použijte tuto hodnotu, chcete-li označit, že nebyly zadány žádné další volby. **CMNONE** pomáhá programovou dokumentaci; není určena, aby byla tato volba použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CMDEFV**.

CMSID (celé číslo se znaménkem za 10 číslic)

Jedná se o identifikátor struktury; hodnota musí být:

CMSIDV
Identifikátor pro strukturu voleb pro vytváření zpracování zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CMSIDV**.

CMVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

CMVER1
Version-1 vytvoří strukturu voleb zpracování zpráv.

Následující konstanta uvádí 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 693. Počáteční hodnoty polí 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 1038
- "Počáteční hodnoty" na stránce 1043
- "Deklarace RPG" na stránce 1044

Pole

Struktura MQCNO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CCDTUL (celé číslo se znaménkem 10 číslic)

CCDTUL je délka řetězce identifikovaného buď CCDTUP, nebo CCDTUO, který obsahuje adresu 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 adresu URL identifikující umístění tabulky kanálu připojení klienta, která se má použít 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 adresu URL pro 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.

V 9.2.0 **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 .

V 9.2.0 **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.

V 9.2.0 **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.

V 9.2.0 Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CNVER7.

Počáteční hodnoty

Tabulka 694. Počáteční hodnoty polí v MQCNO

Název pole	Název konstanty	Hodnota konstanty
CNSID	CNSIDV	'CNO¬
CNVER	CNVER5	1
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

Tabulka 694. Počáteční hodnoty polí v MQCNO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     00000000000000000000000000-
     D                                     000000000000000')
     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                                     00000000000000000000000000-
     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) v systému IBM i

Souhrn struktury MQCSP pro IBM i.

Přehled

Účel: Struktura MQCSP povoluje autorizační službu pro ověření ID uživatele a hesla. Struktura parametrů zabezpečení připojení MQCSP je určena na volání MQCONNX.

Znaková sada a kódování: Data v MQCSP musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT.

- "Pole" na stránce 1045
- "Počáteční hodnoty" na stránce 1047
- "Deklarace RPG" na stránce 1047

Pole

Struktura MQCSP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CSAUTHT (10ciferné celé číslo se znaménkem)

Jedná se o typ ověření, které se má provést.

Platné jsou tyto hodnoty:

CSAN
Nepoužívejte pole ID uživatele a heslo.

CSAUIAP
Ověřte ID uživatele a pole hesel.

Toto je vstupní pole. Počáteční hodnota tohoto pole je CSAN.

CSCPPL (10ciferné celé číslo se znaménkem)

Toto je délka hesla, které se má použít při ověření.

Maximální délka hesla není závislá na platformě. Je-li délka hesla větší než povolená, požadavek na ověření selže s hodnotou RC2035.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCPPO (10ciferné celé číslo se znaménkem)

Toto je posun v bajtech hesla, které má být použito při ověření.

Odsazení může být kladné nebo záporné.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCPPP (ukazatel)

Jedná se o adresu hesla, které má být použito při ověřování.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null.

CSCSPUIL (10ciferné celé číslo se znaménkem)

Toto je délka ID uživatele, které se má použít při ověření.

Maximální délka ID uživatele není závislá na platformě. Je-li délka ID uživatele větší než povolená, požadavek na ověření selže s hodnotou RC2035.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCSPUIO (10číslicové celé číslo se znaménkem)

Jedná se o ofset v bajtech ID uživatele, které se má použít při ověření.

Odsazení může být kladné nebo záporné.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

CSCSPUIP (ukazatel)

Jedná se o adresu ID uživatele, které má být použito pro ověření.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud je CSVER menší než CSVER5.

CSRE1 (čtyřbajtový znakový řetězec)

Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

CSRS2 (8bajtový znakový řetězec)

Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

CSSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

CSSIDV
Identifikátor struktury parametrů zabezpečení.

CSVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

CSVER1
Struktura parametrů zabezpečení Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

CSVERC
Aktuální verze struktury parametrů zabezpečení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CSVER1.

Počáteční hodnoty

Tabulka 695. Počáteční hodnoty polí 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

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 (Řídicí struktura voleb zpětného volání) v systému IBM i

Struktura určující funkci zpětného volání řízení.

Přehled

Účel

Struktura MQCTLO se používá k určení voleb souvisejících s funkcí zpětného volání řízení.

Struktura je vstupním a výstupním parametrem na volání MQCTL .

Verze

Aktuální verze MQCTLO je CTLV1.

Znaková sada a kódování

Data v MQCTLO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1048
- "Počáteční hodnoty" na stránce 1049
- "Deklarace RPG" na stránce 1049

Pole

Struktura MQCTLO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

COCONNAREA (Celé číslo se znaménkem 10)

Struktura voleb ovládacího prvku-pole ConnectionArea .

Toto je pole, které je k dispozici pro funkci zpětného volání, které má být použito.

Správce front nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole CBCCONNAREA struktury MQCBC, což je parametr volání MQCB.

Toto pole je ignorováno pro všechny operace jiné než CTLSR a CTLSW.

Jedná se o vstupní a výstupní pole pro funkci zpětného volání. Počáteční hodnota tohoto pole je ukazatel null nebo null bajtů.

COOPT (10ciferné celé číslo se znaménkem)

Volby, které řídí akci MQCTLO.

CTLFQ
Vynutit selhání volání MQCTLO, je-li správce front nebo připojení ve stavu uvedení do klidového stavu.

Zadejte GMFIQ, v rámci voleb MQGMO předaných volání MQCB, abyste způsobovali oznámení spotřebitelům zpráv, když jsou uvádění do klidového stavu.

CTLTHSTAR
Tato volba informuje systém o tom, že aplikace vyžaduje, aby všichni spotřebitelé zpráv, pro stejné připojení, byli volání na stejném podprocesu.

Výchozí volba: Pokud nepotřebujete žádné z popsaných voleb, použijte následující volbu:

CLNO
Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty. CTLNO je definováno v dokumentaci programu pomoci; není

zamýšleno, aby tato volba byla použita s jinou, ale jako její hodnota je nula, takové použití nelze detekovat.

Toto je vstupní pole. Počáteční hodnota pole *COOPT* je CTLNO.

CORSV (10ciferné celé číslo se znaménkem)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak.

COSID (10ciferné celé číslo se znaménkem)

Struktura voleb řízení- StrucId .

Jedná se o identifikátor struktury; hodnota musí být:

CLSI
Identifikátor pro strukturu voleb ovládacích prvků.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CTLSI.

COVER (10ciferné celé číslo se znaménkem)

Struktura voleb řízení-pole Verze.

Jedná se o číslo verze struktury; hodnota musí být:

CTLV1
Version-1 Struktura voleb řízení.

Následující konstanta uvádí číslo verze aktuální verze:

CLCV
Aktuální verze struktury voleb řízení.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je CTLV1.

Počáteční hodnoty

Tabulka 696. Počáteční hodnoty polí v MQCTLO

Název pole	Název konstanty	Hodnota konstanty
COSID	CLSI	'CTLO'
COVER	CTLV1	1
COOPT	CLNO	Hodnoty null
CORSV	Vyhrazené pole	
COCONNAREA	Není	Nulový ukazatel nebo bajty 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) v systému IBM i

Struktura MQDH popisuje další data, která se nacházejí ve zprávě, když se jedná o zprávu rozdělovníku uloženou v přenosové frontě.

Přehled

Účel: Zpráva distribučního seznamu je zpráva, která je odeslána do více cílových front. Další data sestávají ze struktury MQDH, za nimiž následuje pole záznamů MQOR a pole záznamů MQPMR.

Tato struktura je určena pro použití specializovaných aplikací, které vložila zprávy přímo do přenosových front nebo které odebírají zprávy z přenosových front (například: agenti kanálů pro zprávy).

Tato struktura by neměla být používána běžnými aplikacemi, které jednoduše chtějí vložit zprávy do distribučních seznamů. Tyto aplikace by měly používat strukturu MQOD k definování cílů v distribučním seznamu a struktura MQPMO pro uvedení vlastností zpráv nebo příjmu informací o zprávách odeslaných do jednotlivých míst určení.

Znaková sada a kódování: Data v MQDH musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT pro programovací jazyk C.

Znaková sada a kódování MQDH musí být nastaveno do polí *MDCSI* a *MDENC* v:

- MQMD (je-li struktura MQDH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQDH (všechny ostatní případy).

Použití: Pokud aplikace vloží zprávu do distribučního seznamu a některá nebo všechna místa určení jsou vzdálená, předpona správce front obsahuje předpony dat aplikační zprávy se strukturami MQXQH a MQDH a umístí zprávu do příslušné přenosové fronty. Data se proto objevují v následujícím pořadí, když se zpráva nachází 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 takových zpráv a umístit je do různých přenosových front. V takovém případě struktury MQDH v těchto zprávách identifikují různé podmnožiny cílů definovaných v seznamu distribucí otevřeném aplikací.

Aplikace, která vloží zprávu do přenosové fronty přímo do přenosové fronty, se musí podřídit dříve popsané posloupnosti a musí zajistit správnost struktury MQDH. Pokud struktura MQDH není platná, může správce front rozhodnout o selhání volání MQPUT nebo MQPUT1 s kódem příčiny RC2135.

Zprávy lze ukládat do fronty v podobě distribučního seznamu pouze v případě, že je fronta definovaná jako schopnost podpory zpráv distribučního seznamu (viz atribut fronty **DistLists** popsaný v části “Atributy pro fronty” na stránce 1353). Pokud aplikace umístí zprávu distribučního seznamu přímo do fronty, která nepodporuje distribuční seznamy, rozdělí správce front zprávu distribučního seznamu do jednotlivých zpráv a umístí je do fronty místo toho.

- “Pole” na stránce 1050
- “Počáteční hodnoty” na stránce 1053
- “Deklarace RPG” na stránce 1054

Pole

Struktura MQDH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DHCNT (10ciferné celé číslo se znaménkem)

Počet záznamů MQOR, které jsou k dispozici.

Tato hodnota definuje počet míst určení. Rozdělovník musí vždy obsahovat alespoň jedno místo určení, takže *DHCNT* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

DHCSI (10číslicové podepsané celé číslo)

Identifikátor znakové sady, která následuje za záznamy MQOR a MQPMR.

Tato hodnota určuje identifikátor znakové sady dat, která následují za polem záznamů MQOR a MQPMR. Nevztahuje se na znaková data ve struktuře MQDH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

DHENC (10ciferné celé číslo se znaménkem)

Numerické kódování dat za záznamy MQOR a MQPMR.

Určuje číselné kódování dat, která jsou uvedena za pole MQOR a záznamů MQPMR; nevztahuje se na číselná data ve struktuře MQDH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

DHFLG (10ciferné celé číslo se znaménkem)

Obecné příznaky.

Lze zadat následující příznak:

DHFNEW
Generujte nové identifikátory zpráv.

Tento příznak označuje, že má být vygenerován nový identifikátor zprávy pro každé místo určení v rozdělovníku. Toto nastavení lze nastavit pouze v případě, že nejsou přítomny žádné záznamy vložení zpráv nebo jsou-li záznamy přítomny, ale neobsahují pole *PRMID* .

Použití tohoto parametru dekóduje generování identifikátorů zpráv až do poslední možné chvíle, konkrétně v okamžiku, kdy je zpráva distribučního seznamu konečně rozdělena na jednotlivé zprávy. Tím se minimalizuje množství řídicích informací, které musí tok obsahovat zprávu distribuční seznam.

Když aplikace vloží zprávu do distribučního seznamu, správce front nastaví DHFNEW v MQDH, který vygeneruje, když jsou obě následující příkazy pravdivé:

- K dispozici nejsou žádné záznamy vložení zpráv poskytnuté aplikací nebo zadané záznamy neobsahují pole *PRMID* .
- Pole *MDMID* v MQMD je MINONE, nebo pole *PMOPT* v MQPMO obsahuje PMNMID

Nejsou-li vyžadovány žádné příznaky, lze zadat následující údaje:

DHFNON
Žádné vlajky.

Tato konstanta označuje, že nebyly zadány žádné parametry. DHFNON je definován v dokumentaci programu pomoci. Není určeno, aby byla tato konstanta použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je DHFNON.

DHFMT (8bajtový znakový řetězec)

Název formátu dat, který následuje za záznamy MQOR a MQPMR.

Určuje název formátu dat, která následují za pole záznamů MQOD a MQPMR (podle toho, co nastane dříve).

Na základě 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.

Počáteční hodnota tohoto pole je FMNONE.

DHLEN (10ciferné celé číslo se znaménkem)

Délka struktury MQDH plus následující záznamy MQOR a MQPMR.

Jedná se o počet bajtů od začátku struktury MQDH do začátku dat zprávy za pole záznamů MQOR a MQPMR. Data se objevují 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 jedním nebo více strukturou MQDH, poli záznamů a daty zprávy, tyto nepoužívané bajty musí být zahrnuty do hodnoty *DHLEN*, ale obsah těchto bajtů není správcem front zachován. Je platný pro pole záznamů MQPMR, aby bylo před polem záznamů MQOR předcházet.

Počáteční hodnota tohoto pole je 0.

DHORO (10ciferné celé číslo se znaménkem)

Odsazení prvního záznamu MQOR od začátku MQDH.

Toto pole uvádí posun v bajtech prvního záznamu v poli záznamů objektů MQOR, který obsahuje názvy cílových front. V tomto poli jsou záznamy *DHCNT* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem objektu a předchozím polem) jsou zahrnuty do délky zadané v poli *DHLEN* .

Rozdělovník musí vždy obsahovat alespoň jedno místo určení, takže *DHORO* musí být vždy větší než nula.

Počáteční hodnota tohoto pole je 0.

DHPRF (10číslicové celé číslo se znaménkem)

Příznaky určující, která pole MQPMR jsou přítomna.

Lze zadat nula nebo více z následujících příznaků:

PFMID
Zobrazí se pole identifikátoru zprávy.

PFARCID
Pole identifikátoru korelace je přítomno.

PFGID
Pole identifikátoru skupiny je přítomno.

PFFB
Je přítomno pole zpětné vazby.

PFAC
Pole Účetní-token je přítomno.

Nejsou-li přítomna žádná pole MQPMR, lze zadat následující:

PFNONE
Nejsou přítomna žádná pole záznamu vložení zprávy.

PFNONE je definován v dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je PFNONE.

DHPRO (10ciferné celé číslo se znaménkem)

Offset prvního záznamu MQPMR od začátku MQDH.

Toto pole uvádí posun v bajtech prvního záznamu v poli záznamů vložených zpráv MQPMR, který obsahuje vlastnosti zprávy. Je-li přítomen, v tomto poli jsou záznamy *DHCNT* . Tyto záznamy (plus všechny bajty přeskočené mezi prvním záznamem vložení zprávy a předchozím polem) jsou zahrnuty do délky zadané v poli *DHLEN* .

Záznamy vložení zpráv jsou volitelné; pokud nejsou poskytnuty žádné záznamy, *DHPRO* je nula a *DHPRF* má hodnotu PFNONE.

Počáteční hodnota tohoto pole je 0.

DHSID (4-bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

DHSIDV
Identifikátor pro strukturu záhlaví distribuce.

Počáteční hodnota tohoto pole je DHSIDV.

DHVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DHVER1
Číslo verze pro strukturu záhlaví distribuce.

Následující konstanta uvádí číslo verze aktuální verze:

DHVERC
Aktuální verze struktury záhlaví distribuce.

Počáteční hodnota tohoto pole je DHVER1.

Počáteční hodnoty

Tabulka 697. Počáteční hodnoty polí 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

Tabulka 697. Počáteční hodnoty polí v MQDH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
DHFLG	DHFNON	0
DHPRF	PFNONE	0
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 zpráv) v systému IBM i

Přehled

Účel

Struktura MQDLH popisuje informace, které preřadí data zpráv aplikací ve frontě nedoručených zpráv (undelivered-message). Do fronty nedoručených zpráv může být doručena zpráva, protože správce front nebo agent kanálu zpráv je přesměroval do fronty. Aplikace může odeslat zprávu přímo do fronty.

Název formátu

FMDLH

Znaková sada a kódování

Objekt MQDLH může být na začátku dat zprávy aplikace. Pokud tomu tak je, pole ve struktuře MQDLH se nacházejí ve znakové sadě a kódování poskytnuté poli MDCSI a MDENC . Pokud ne, znaková sada a kódování se nastavují v polích MDCSI a MDENC ve struktuře záhlaví, která předchází MQDLH.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky, které jsou platné v názvech front.

Použití

Aplikace, které vložila zprávy přímo do fronty nedoručených zpráv, musí před daty zprávy uvést strukturu MQDLH a inicializovat pole s příslušnými hodnotami. Správce front však nevyžaduje, aby byla přítomna struktura MQDLH , nebo že jsou pro pole zadány platné hodnoty.

Je-li zpráva příliš dlouhá na vložení do fronty nedoručených zpráv, musí aplikace zvážit provedení jedné z následujících možností:

- Ořízněte data zprávy tak, aby se vešly do fronty nedoručených zpráv.
- Zaznamenejte zprávu do pomocné paměti a do fronty nedoručených zpráv uveďte zprávu o výjimce indikující, že zpráva je příliš dlouhá.
- Vyřazovat zprávu a vrátit chybu původci. Je-li zpráva kritická zpráva. Zlikvidujte zprávu pouze v případě, že je známo, že původce stále má kopii zprávy. Příklad: Zpráva přijatá agentem kanálu zpráv z komunikačního kanálu.

Výběr vhodných voleb závisí na návrhu aplikace.

Správce front provádí speciální zpracování, je-li zpráva, která je částí, vložena se strukturou MQDLH na přední straně. Další podrobnosti lze najít v popisu struktury MQMDE .

- "Vložení zpráv do fronty nedoručených zpráv" na stránce 1055
- "Získávání zpráv z fronty nedoručených zpráv" na stránce 1056
- "Pole" na stránce 1056
- "Počáteční hodnoty" na stránce 1059
- "Deklarace RPG" na stránce 1060

Vložení zpráv do fronty nedoručených zpráv

Je-li zpráva vložena do fronty nedoručených zpráv, musí být struktura MQMD použitá pro volání MQPUT nebo MQPUT1 stejná jako struktura MQMD přidružená ke zprávě. MQMD je obvykle ten, který je vrácen voláním MQGET , s výjimkou následujících případů:

- Pole MDCSI a MDENC musí být nastavena na libovolnou znakovou sadu a kódování se používá pro pole ve struktuře MQDLH .
- Pole MDFMT musí být nastaveno na FMDLH , aby indikovalo, že data začínají strukturou MQDLH .
- Kontextové pole MDACC, MDAID, MDAOD, MDPAN, MDPAT, MDPD, MDPTa MDUID musí být nastaveny pomocí volby kontextu odpovídající podmínkám:
  - Aplikace, která vkládá do fronty nedoručených zpráv zprávu, která nesouvisí s žádnou předchozí zprávou, musí použít volbu PMDEFC . Volba PMDEFC způsobí, že správce front nastaví všechny kontextové pole v deskriptoru zpráv na jejich výchozí hodnoty.
  - Aplikace serveru, která vkládá do fronty nedoručených zpráv zprávu, kterou přijme, musí použít volbu PMPASA , aby se zachovalo původní informace o kontextu.
  - Aplikace serveru, 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 o původu na serveru.
  - Agent oznamovacího kanálu, který vloží do fronty nedoručených zpráv zprávu, kterou obdrží z 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, které popisují data, která následují za strukturou MQDLH . Tyto hodnoty jsou obvykle hodnoty z původního deskriptoru zpráv.
- Kontextové pole DLPAT, DLPAN, DLPDa DLPT musí být nastaveny na hodnoty odpovídající aplikaci, která vkládá zprávu do fronty nedoručených zpráv. Tyto hodnoty nesouvisí s původní zprávou.
- Jiná pole musí být nastavena podle potřeby.

Aplikace musí zajistit, aby všechna pole měla platné hodnoty a že znaková pole jsou vyplněna mezerami do definované délky pole. Znaková data nesmí být ukončena předčasně s použitím znaku null. Správce front nekonvertuje hodnotu 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, zda zprávy začínají na strukturu MQDLH . Aplikace může určit, zda se struktura MQDLH vyskytuje prozkoumáním pole MDFMT v deskriptoru zprávy MQMD. Má-li pole hodnotu FMDLH, data zprávy začínají strukturou MQDLH . Zprávy ve frontě zablokovaných dopisů mohou být zkráceny, pokud byly původně příliš dlouhé pro frontu, pro kterou byly určeny.

Pole

Struktura MQDLH obsahuje následující pole; pole jsou popsána v abecedním pořadí:

DLCSI (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady, která následuje za MQDLH.

Parametr DLCSI určuje identifikátor znakové sady dat, která následuje za strukturou MQDLH . Data jsou obvykle z původní zprávy. Nevztahuje se na znaková data v samotné struktuře MQDLH .

V případě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech po této struktuře jsou 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 chybě, hodnota CSINHT není vrácena voláním funkce MQGET .

CSINHT nelze použít, je-li hodnota pole MDPAT v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

DLDM (48-bajtový znakový řetězec)

Název původního správce cílové fronty.

Jedná se o název správce front, který byl původním cílem pro zprávu.

Délka tohoto pole je dána LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

DLDQ (48-bajtový znakový řetězec)

Název původní cílové fronty.

Jedná se o název fronty zpráv, která byla původním cílem zprávy.

Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

DLENC (10ciferné celé číslo se znaménkem)

Číselné kódování dat, která následují za MQDLH.

DLENC uvádí číselné kódování dat, která následují za strukturou MQDLH . Data jsou obvykle z původní zprávy. Nevztahuje se na číselná data v samotné struktuře MQDLH .

V případě 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, který následuje za MQDLH.

Uvádí název formátu dat, která následuje za strukturou MQDLH (obvykle data z původní zprávy).

V případě 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 MQMD.

Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

DLPAN (28bajtový znakový řetězec)

Název aplikace, která vložila zprávu do fronty nedoručených zpráv (undelivered-message).

Formát názvu závisí na poli DLPAT . Viz popis pole MDPAN v příručce "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099.

Je-li správce front, který přesměrovává zprávu do fronty nedoručených zpráv, obsahuje řetězec DLPAN prvních 28 znaků názvu správce front. Název bude vyplněn mezerami, je-li to nutné.

Délka tohoto pole je dána LNPAN. Počáteční hodnota tohoto pole je 28 prázdných znaků.

DLPAT (10ciferné celé číslo se znaménkem)

Typ aplikace, která vložila zprávu do fronty nedoručených zpráv (undelivered-message).

Toto pole má stejný význam jako pole MDPAT v deskriptoru zprávy MQMD (podrobnosti viz "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099).

Je-li správce front, který přesměrovává zprávu do fronty nedoručených zpráv, má parametr DLPAT hodnotu ATQM.

Počáteční hodnota tohoto pole je 0.

DLPD (8bajtový znakový řetězec)

Datum, kdy byla zpráva vložena do fronty smrtelného dopisu (nedoručená zpráva).

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- YYYYMMDD

kde znaky představují:

YYYY
rok (čtyři číselné 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 jsou systémové hodiny nastaveny přesně na GMT.

Délka tohoto pole je dána LNPDAT. Počáteční hodnota tohoto pole je osm prázdných znaků.

DLPT (8bajtový znakový řetězec)

Čas, kdy byla zpráva vložena do fronty nedoručených zpráv (undelivered-message).

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSSTH

kde znaky představují (v pořadí):

HH
hodin (00 až 23)

MM
minut (00 až 59)

SS
sekundy (00 až 59; viz poznámka dále v tomto tématu)

T
desetiny sekundy (0 až 9)

H
setiny sekundy (0 až 9)

Poznámka: Je-li časová základna systému synchronizována s přesným časovým standardem, je možné, aby 60 nebo 61 byly vráceny pro sekundy v DLPT. Druhá sekunda se vyskytne, když se do globálního časového standardu vloží přestupné sekundy.

Greenwichský střední čas (GMT) se používá pro pole DLPD a DLPT za předpokladu, že jsou systémové hodiny nastaveny přesně na GMT.

Délka tohoto pole je dána LNPTIM. Počáteční hodnota tohoto pole je osm prázdných znaků.

DLREA (10ciferné celé číslo se znaménkem)

Zpráva o příčině byla doručena do fronty nedoručených zpráv (nedoručená zpráva).

To identifikuje důvod, proč byla zpráva umístěna do fronty nedoručených zpráv místo na původní cílové frontě. Musí to být jedna z hodnot FB* nebo RC* (např. RC2053). Podrobnosti o společných hodnotách FB*, které se mohou vyskytnout, najdete v popisu pole *MDFB* v příručce "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099 .

Je-li hodnota v rozsahu FBIFST až FBILST, skutečný kód chyby IMS může být určen odečtením FBIERR od hodnoty pole *DLREA* .

Některé hodnoty FB* se vyskytují pouze v tomto poli. Souvisí s zprávami úložiště, spouštěcími zprávami nebo zprávami přenosové fronty, které jsou přeneseny do fronty nedoručených zpráv. Tyto hodnoty jsou:

FBABEG
Aplikaci nelze spustit.

Aplikace spouštěla zprávu spouštěcího impulsu nemůže spustit aplikaci uvedenou v poli TMAI zprávy spouštěče; viz "MQTM-Zpráva spouštěče" na stránce 1222.

FBATYP
Chyba typu aplikace.

Aplikace spouštějící zprávu nespustilo aplikaci, protože pole TMAT zprávy spouštěče je neplatné; viz "MQTM-Zpráva spouštěče" na stránce 1222.

FBBOCD
Přijímací kanál klastru byl odstraněn.

Zpráva byla uvedena ve přenosové frontě klastru určené pro frontu klastru, která byla otevřena pomocí volby FBIERR . Kanál příjemce vzdáleného klastru, který má být použit k přenosu zprávy do cílové fronty, byl odstraněn před odesláním zprávy. Protože byl zadán parametr 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 souboru MQMD uvádí FMTM, ale zpráva nezačíná platnou strukturou MQTM . Například mnemonický modul *TMSID* mnemonic může být neplatný. Je možné, že produkt *TMVER* nebyl rozpoznán. Délka zprávy spouštěče může být nedostatečná, aby mohla obsahovat 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 oznamovacího kanálu umístí 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 s dead-letter.

Počáteční hodnota tohoto pole je DLSIDV.

DLVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DLVER1
Číslo verze pro strukturu záhlaví dead-letter.

Následující konstanta uvádí číslo verze aktuální verze:

DLVERC
Aktuální verze struktury záhlaví dead-letter.

Počáteční hodnota tohoto pole je DLVER1.

Počáteční hodnoty

Tabulka 698. Počáteční hodnoty polí v 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
Rozhraní 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 (Výmaz voleb zpracování zpráv) na IBM i

Struktura **MQDMHO** umožňuje aplikacím určit volby, které řídí způsob odstranění manipulátorů zpráv.

Přehled

Účel: Struktura je vstupním parametrem na volání **MQDLTMH** .

Znaková sada a kódování: Data v souboru **MQDMHO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1060
- "Počáteční hodnoty" na stránce 1061
- "Deklarace RPG" na stránce 1061

Pole

Struktura MQDMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DMOPT (10číslicové celé číslo se znaménkem)

Hodnota musí být:

DMNONE

Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMNONE**.

DMSID (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor struktury; hodnota musí být:

DMSIDV

Identifikátor pro strukturu voleb pro zpracování odstranění zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMSIDV**.

DMVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

DMVER1
Version-1 -odstranění struktury voleb zpracování zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

DMVERC
Aktuální verze struktury voleb pro zpracování odstranění zprávy.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **DMVER1**.

Počáteční hodnoty

Tabulka 699. Počáteční hodnoty polí 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 (Výmaz voleb vlastností zprávy) v systému IBM i

Struktura definující volby vlastností odstranění zprávy.

Přehled

Účel: Struktura MQDMPO umožňuje aplikacím zadávat volby, které řídí způsob, jakým se odstraňují vlastnosti zpráv. Struktura je vstupním parametrem volání MQDLTMP.

Znaková sada a kódování: Data ve struktuře MQDMPO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1061
- "Počáteční hodnoty" na stránce 1062
- "Deklarace RPG" na stránce 1063

Pole

Struktura MQDMPO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

DPOPT (10číslicové celé číslo se znaménkem)

Odstraňte strukturu voleb vlastností zprávy-pole DPOPT.

Volby umístění: Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti.

DPDELF
Odstraní první vlastnost, která odpovídá uvedenému názvu.

DPDELCKUN
Odstraní vlastnost, na kterou ukazuje kurzor vlastností. Jedná se o vlastnost, která byla naposledy dotazovaná pomocí volby IPINQF nebo IPINQN.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy. Je také resetováno, když je popisovač zprávy uveden v poli *HMSG* MQGMO na volání MQGET, nebo MQPMO struktury na volání MQPUT.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy, nebo když je popisovač zprávy uveden v poli *HMSG* struktury MQGMO na struktuře MQGET na volání MQGET nebo MQPMO na volání MQPUT.

Volání selže s kódem dokončení CCFAIL a s příčinou RC2471 , je-li tato volba použita, když se kurzor vlastnosti ještě nezavedl. Také selže s těmito kódy, je-li vlastnost, na kterou ukazuje kurzor vlastností, již odstraněna.

Pokud není ani jedna z těchto voleb povinná, lze použít následující volbu:

DPNONE
Nejsou uvedeny žádné volby.

Počáteční hodnota tohoto vstupního pole je DPDELF.

DPSID (10ciferné celé číslo se znaménkem)

Struktura voleb vlastností pro odstranění zprávy-pole DPSID.

Jedná se o identifikátor struktury. Hodnota musí být:

PPSIDVERS
Identifikátor pro strukturu voleb vlastností odstranění zprávy.

Toto pole je vždy vstupním polem. Počáteční hodnota tohoto pole je DPSIDV.

DPVER (10ciferné celé číslo se znaménkem)

Odstraňte strukturu voleb vlastností zprávy-pole DPVER.

Jedná se o číslo verze struktury. Hodnota musí být:

DPVER1
Číslo verze pro strukturu voleb vlastností odstranění zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

DPVERC
Aktuální verze struktury voleb pro odstranění vlastností zprávy.

Toto pole je vždy vstupním polem. Počáteční hodnota tohoto pole je DPVER1.

Počáteční hodnoty

Tabulka 700. Počáteční hodnoty polí v MQDPMO

Název pole	Název konstanty	Hodnota konstanty
DPSID	PPSIDVERS	'DMPO'

Tabulka 700. Počáteční hodnoty polí v MQDPMO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
DPVER	DPVER1	1
DPOPT	Volby, které řídí akci příkazu 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 (Vestavěné záhlaví PCF) v systému IBM i

Přehled

Účel

Struktura MQEPH popisuje další data, která se vyskytují ve zprávě, když je tato zpráva programovatelná zpráva ve formátu příkazu (PCF). Pole *EPPFH* definuje parametry PCF, které následují za touto strukturou, a to vám umožňuje sledovat data zprávy PCF s ostatními záhlavími.

Název formátu

EPFMT

Znaková sada a kódování

Data v MQEPH musí být ve znakové sadě a kódování lokálního správce front; toto je dáno atributem správce front **CCSID** .

Nastavte znakovou sadu a kódování MQEPH do polí *MDCSI* a *MDENC* v:

- MQMD (je-li struktura MQEPH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQEPH (všechny ostatní případy).

Použití

Struktury MQEPH nelze použít k odeslání příkazů na příkazový server nebo na jiný server PCF-accepting správce front.

Podobně ani příkazový server nebo jakýkoli jiný server PCF-acceptor správce front negeneruje odezvy nebo události obsahující struktury MQEPH.

- "Pole" na stránce 1063
- "Počáteční hodnoty" na stránce 1065
- "Deklarace RPG" na stránce 1065

Pole

Struktura MQEPH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

EPCSI (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor znakové sady dat, která následuje strukturu MQEPH a přidružené parametry PCF; nepoužívá se pro znaková data v samotné struktuře MQEPH.

Počáteční hodnota tohoto pole je EPCUND.

EPENC (10ciferné celé číslo se znaménkem)

Jedná se o číselné kódování dat, která se řídí strukturou MQEPH a s přiřazovanými parametry PCF; nepoužívá se pro znaková data ve struktuře MQEPH.

Počáteční hodnota tohoto pole je 0.

EPFLG (10ciferné celé číslo se znaménkem)

K dispozici jsou tyto hodnoty:

HODNOTA EPNONE
Nebyly zadány žádné parametry. *MDCSI* EPNONE je definována pro dokumentaci programu podpory. Není určeno, aby tato konstanta byla použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

EPCSEM
Znaková sada parametrů, které obsahují znaková data, se zadává jednotlivě v poli *CCSID* v každé struktuře. Znaková sada polí *EPSID* a *EPFMT* je definována *CCSID* ve struktuře záhlaví, která předchází struktuře MQEPH, nebo pole *MDCSI* v deskriptoru MQMD, pokud se MQEPH nachází na začátku zprávy.

Počáteční hodnota tohoto pole je EPNONE.

EPFMT (8bajtový znakový řetězec)

Jedná se o název formátu dat, která se řídí strukturou MQEPH a s přidruženými parametry PCF.

Počáteční hodnota tohoto pole je EPFMNO.

EPLEN (10ciferné celé číslo se znaménkem)

Jedná se o množství dat, která předchází další struktuře záhlaví. Zahrnuje:

- Délka záhlaví MQEPH
- Délka všech parametrů PCF za záhlavím
- Jakákoli prázdná výplň za těmito parametry

EPLEN musí být násobkem 4.

Část struktury pevné délky je definována hodnotou EPSTLF.

Počáteční hodnota tohoto pole je 68.

EPPCFH (MQCFH)

Jedná se o záhlaví PCF (Programmable command format) definující parametry PCF, které se řídí strukturou MQEPH. To vám umožní sledovat data zprávy PCF s ostatními záhlavími.

Hlavička PCF je na počátku definována s následujícími hodnotami:

Tabulka 701. Počáteční hodnoty polí 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	KEK	0

Tabulka 701. Počáteční hodnoty polí v EPPCFH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
EP3REA	RCNONE	0
EP3CNT	Není	0

Aplikace musí změnit EP3TYP z CFTNON na platný typ struktury pro použití vložené hlavičky PCF.

EPSID (čtyřbajtový znakový řetězec)

Hodnota musí být:

EPSTIDSKÝ
Identifikátor pro strukturu záhlaví vloženého kódu PCF.

Počáteční hodnota tohoto pole je EPSTID.

EPVER (10ciferné celé číslo se znaménkem)

Hodnota může být následující:

EPVER1
Číslo verze pro vloženou strukturu záhlaví PCF.

Následující konstanta uvádí číslo verze aktuální verze:

EPVER3
Aktuální verze vestavěné struktury záhlaví PCF.

Počáteční hodnota tohoto pole je EPVER3.

Počáteční hodnoty

Tabulka 702. Počáteční hodnoty polí v MQEPH		
Název pole	**Název konstanty**	**Hodnota konstanty**
EPSID	EPSTIDSKÝ	'EP¬¬'
EPVER	EPVER1	1
EPLEN	EPSTLF	68
EPENC	Není	0
EPCSI	EPCUNSKÉ	0
EPFMT	EPFMNO	Mezery
EPFLG	HODNOTA EPNONE	0
EPPCFH	Názvy a hodnoty, jak jsou definovány v produktu Tabulka 701 na stránce 1064	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 získání zpráv) v systému IBM i

Struktura MQGMO umožňuje aplikaci určit volby, které řídí způsob odebírání zpráv z front.

Přehled

Účel

Struktura je vstupním/výstupním parametrem na volání MQGET.

Verze

Aktuální verze MQGMO je GMVER4. Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková 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 , aplikace musí nastavit pole *GMVER* na číslo verze požadované verze.

Znaková sada a kódování

Data v produktu MQGMO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého produktem ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

- “Pole” na stránce 1066
- “Počáteční hodnoty” na stránce 1086
- “Deklarace RPG” na stránce 1086

Pole

Struktura MQGMO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

GMGST (1bajtový znakový řetězec)

Příznak označující, zda je zpráva načtená, ve skupině.

Má jednu z následujících hodnot:

GSNIGA
Zpráva se nenachází ve skupině.

GSMIG.
Zpráva se nachází ve skupině, ale není poslední ve skupině.

GSLMIG
Zpráva je poslední ve skupině.

Tato hodnota je také návratová 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 *GMVER* je menší než GMVER2.

GMMH (celé číslo se znaménkem 10 číslic)

popisovač zprávy

Je-li uvedena volba GMPREQ a atribut fronty PRPCTL není nastaven na PRPRFH, pak se jedná o popisovač zprávy, který je naplněn vlastnostmi zprávy načítané z fronty. Popisovač je vytvořen voláním MQCRTMH. Všechny vlastnosti, které jsou již přidruženy k popisovači, jsou před načtením zprávy vymazány.

Je možné zadat také následující hodnotu:
MQM_NONE
Nebyl zadán popisovač zprávy.

Pokud je zadán platný popisovač zprávy a ve výstupu obsahuje vlastnosti zprávy, není na volání MQGET vyžadován žádný deskriptor zprávy. Pro vstupní pole se použije deskriptor zprávy přidružený k popisovači zpráv.

Je-li v rámci volání MQGET zadán deskriptor zprávy, má vždy přednost před deskriptorem zpráv přidruženým k manipulátoru zprávy.

Je-li uveden GMPRRF, nebo je uveden GMPRAQ a atribut fronty PRPCTL je PRPRFH, 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 zpráv aktualizovány tak, aby odrážely stav načtené zprávy (stejně jako deskriptor zprávy, pokud byl dodán na volání MQGET). Vlastnosti této zprávy lze poté provést zjišťování pomocí volání MQINQMP.

S výjimkou rozšíření deskriptoru zpráv, je-li přítomna vlastnost, která může být inquired s voláním MQINQMP, není obsažena v datech zprávy; pokud zpráva ve frontě obsahuje vlastnosti v datech zprávy, tyto jsou odebrány z dat zprávy před tím, než se data vrátí do aplikace.

Pokud není poskytnut žádný popisovač zprávy, nebo je verze nižší než GMVER4 , pak musíte zadat platný deskriptor zprávy na volání MQGET. Všechny vlastnosti zprávy (s výjimkou vlastností obsažených v deskriptoru zpráv) jsou vráceny v datech zprávy pod hodnotou volby 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 *GMVER* je menší než GMVER4.

GMMO (10ciferné celé číslo se znaménkem)

Volby, které řídí kritéria výběru použitá pro MQGET.

Tyto volby umožňují aplikaci zvolit, která pole v parametru **MSGDSC** se použijí k výběru zprávy vrácené voláním MQGET. Aplikace nastavuje 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 tuto zprávu, jsou kandidáty na načtení pomocí parametru **MSGDSC** na volání MQGET. Pole, pro která není zadána odpovídající volba shody, jsou při výběru zprávy, která má být vrácena, ignorována. Pokud nemají být použita žádná kritéria výběru na volání MQGET (tj. jakákoli zpráva je přijatelná), měl by parametr *GMMO* být nastaven na hodnotu MONONE.

Je-li uvedena hodnota GMLOGO, jsou pro další volání MQGET způsobilé pouze určité zprávy:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, mohou být vráceny pouze zprávy, které mají *MDSEQ* rovnu 1 a *MDOFF* rovnající se 0 . V této situaci lze použít jednu nebo více následujících voleb k výběru, které z vhodných zpráv se vrátí:
  - MOMGI
  - MODORŠTINA
  - MOGRPI

- Existuje-li aktuální skupina nebo logická zpráva, je možné vrátit pouze další zprávu ve skupině nebo v dalším segmentu v logické zprávě a nelze ji změnit zadáním voleb MO*.

V obou případech nelze zadat volby shody, které nelze použít, ale hodnota relevantního pole v parametru **MSGDSC** se musí shodovat s hodnotou odpovídajícího pole ve zprávě, která má být vrácena; volání selže s kódem příčiny RC2247 , že tato podmínka není splněna.

Parametr *GMMO* je ignorován, pokud je zadán buď GMMUC, nebo GMBRWC.

Je možné zadat jednu nebo více následujících voleb:

MOMGI
Načtěte zprávu s uvedeným identifikátorem zprávy.

Tato volba uvádí, 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 týkat (například identifikátor korelace).

Není-li tato volba zadána, bude pole *MDMID* v parametru **MSGDSC** ignorováno a všechny identifikátory zpráv se shodují.

Poznámka: Identifikátor zprávy MINONE je speciální hodnota, která odpovídá libovolnému identifikátoru zprávy v produktu MQMD pro zprávu. Proto uvedení MOMSGI s parametrem MINONE je stejné jako neuvedení MOMSGI.

MODORŠTINA
Načíst zprávu s určeným identifikátorem korelace.

Tato volba uvádí, že načtená zpráva musí mít korelační identifikátor, který odpovídá hodnotě pole *MDCID* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou aplikovat (například identifikátor zprávy).

Není-li tato volba zadána, bude pole *MDCID* v parametru **MSGDSC** ignorováno a všechny identifikátory korelace se budou shodovat.

Poznámka: Identifikátor korelace CINONE je speciální hodnota, která odpovídá libovolnému identifikátoru korelace v deskriptoru MQMD pro zprávu. Proto uvedení MOCORI s CINONE je stejné jako neuvedení MOCORI.

MOGRPI
Načtěte zprávu s uvedeným identifikátorem skupiny.

Tato volba uvádí, 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 týkat (například identifikátor korelace).

Není-li tato volba zadána, bude pole *MDGID* v parametru **MSGDSC** ignorováno a všechny identifikátory skupin se shodují.

Poznámka: Identifikátor skupiny GINONE je speciální hodnota, která odpovídá libovolnému identifikátoru skupiny v deskriptoru MQMD pro zprávu. Proto uvedení parametru MOGRPI s GINONE je stejné jako neuvedení MOGRPI.

MOSEQN
Načtěte zprávu s uvedeným pořadovým číslem zprávy.

Tato volba uvádí, 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 týkat (například identifikátor skupiny).

Není-li tato volba zadána, bude pole *MDSEQ* v parametru **MSGDSC** ignorováno a všechny shody s pořadovými čísly zpráv se budou shodovat.

MOOKIE
Načíst zprávu s určeným posunutím.

Tato volba uvádí, že načtená zpráva musí mít offsetu, který odpovídá hodnotě pole *MDOFF* v parametru **MSGDSC** volání MQGET. Tato shoda je navíc k dalším shodám, které se mohou aplikovat (například pořadové číslo zprávy).

Není-li tato volba zadána, pole *MDOFF* v parametru **MSGDSC** se ignoruje a všechny odchylky se shodují.

Není-li zadána žádná z uvedených voleb, lze použít následující volbu:

MONONE
Žádné shody.

Tato volba určuje, že při výběru zprávy, která má být vrácena, se nepoužívají žádné shody; proto jsou všechny zprávy ve frontě vhodné pro načtení (je však předmětem řízení voleb GMAMSA, GMASGA a GMCMPM).

MONONE je definována pro dokumentaci programu podpory. Není určeno, aby tato volba byla použita s jinou volbou MO*, ale protože její hodnota je nula, takové použití nelze zjistit.

Toto pole je vstupní pole. Počáteční hodnota tohoto pole je MOMSGI s MOCORI. Toto pole je ignorováno, pokud *GMVER* je menší než GMVER2.

Poznámka: Počáteční hodnota pole *GMMO* je definována pro kompatibilitu se správci front dřívějších verzí. Avšak při čtení posloupnosti zpráv z fronty bez použití kritérií výběru tato počáteční hodnota vyžaduje, aby aplikace resetoval pole *MDMID* a *MDCID* do polí MINONE a CINONE před každým voláním MQGET. Potřeba resetovat *MDMID* a *MDCID* se lze vyhnout nastavením *GMVER* na GMVER2a *GMMO* na hodnotu MONONE.

GMOPT (10ciferné celé číslo se znaménkem)

Volby, které řídí akci MQGET.

Může být uvedena nula nebo více z následujících popsaných voleb. Pokud je požadováno více než jedna, lze tyto hodnoty přidat (nepřidávat stejnou konstantu víckrát než jednou). Kombinace voleb, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné.

Volby čekání: Následující volby se vztahují k čekání na příchod zpráv do fronty:

GMWT.
Počkejte na doručení zprávy.

Aplikace bude čekat, dokud nepřijde vhodná zpráva. Maximální doba, po kterou aplikace čeká, je uvedena v *GMWI*.

Pokud jsou požadavky MQGET blokovány nebo požadavky MQGET jsou při čekání blokovány, čekání je zrušeno a volání je dokončeno s CCFAIL a kódem příčiny RC2016, bez ohledu na to, zda ve frontě existují vhodné zprávy.

Tato volba může být použita s volbami GMBRWF nebo GMBRWN.

Pokud je několik aplikací čeká ve stejné sdílené frontě, aplikace nebo aplikace, které jsou aktivovány při doručení vhodné zprávy, jsou popsány dále v této sekci.

Poznámka: V následujícím popisu se jedná o volání procházení MQGET, které určuje jednu z voleb procházení, nikoli však GMLK; volání MQGET s uvedením volby GMLK je považováno za volání bez procházení.

- Pokud jedna nebo více volání MQGET bez procházení čeká, ale čekání na procházení MQGET nebude čekat, aktivuje se jedna z nich.
- Pokud jedna nebo více volání procházení MQGET čeká, ale žádná volání MQGET bez procházení čeká, jsou všechny aktivovány.
- Pokud jedna nebo více volání MQGET bez procházení a jedno nebo více volání procházení MQGET čeká, je aktivována jedna volání operace MQGET bez procházení a žádná, některá nebo všechna volání procházení MQGET. (Aktivované číslo procházení volání MQGET nelze předvídat, protože závisí na aspektech plánování operačního systému a na dalších faktorech.)

Pokud ve stejné frontě čeká více než jedno volání operace MQGET bez procházení, aktivuje se pouze jedna; v této situaci se správce front pokusí o prioritu při čekání na volání mimo procházení v následujícím pořadí:

1. Specifické požadavky typu get-wait, které mohou být uspokojeny pouze určitými zprávami, například s určitými zprávami, které mají specifický *MDMID* nebo *MDCID* (nebo obojí).
2. Obecné požadavky typu get-wait, které mohou být uspokojeny jakoukoli zprávou.

Je třeba poznamenat následující skutečnosti:

- V první kategorii není poskytnuta žádná další priorita pro více konkrétních požadavků typu get-wait, například pro ty, které uvádějí jak *MDMID* , tak *MDCID*.
- V jedné z kategorií nelze předpovědět, která aplikace je vybrána. Zvláště čekání na aplikaci není nutně tím, co je vybráno.
- Délka cesty a aspekty plánování priority operačního systému mohou znamenat, že čeká se aplikace nižší priority operačního systému, než se očekává, že tato zpráva načte zprávu.
- Může se také stát, že aplikace, která nečeká, načte zprávu v preferovaném pořadí na takový, který je.

Hodnota GMWT je ignorována, je-li uvedena s GMBRWC nebo GMMUC; není vygenerována žádná chyba.

GMNWT.
Okamžitě se vraťte, pokud není žádná vhodná zpráva.

Aplikace nebude čekat, pokud není k dispozici žádná vhodná zpráva. Toto je opak volby GMWT a je definován pro dokumentaci programu pomoci. Je-li uveden žádný, je to výchozí nastavení.

GMFIQ
Selhání, pokud je správce front uváděn 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 doba čekání je nevyřízena v době, kdy správce front vstoupí do klidového stavu, postupujte takto:

- Čekání je zrušeno a volání vrátí kód dokončení CCFAIL s kódem příčiny RC2161 .

Není-li GMFIQ zadán a správce front přejde do klidového stavu, nebude čekání zrušeno.

Volby synchronizačního bodu: Následující volby souvisí s účastí volání MQGET v rámci pracovní jednotky:

GMSYP
Získejte zprávu s řízením synchronizačního bodu.

Požadavek má fungovat v rámci běžných protokolů jednotky práce. Zpráva je označena jako nedostupná pro jiné aplikace, ale je vymazána z fronty pouze tehdy, když je potvrzena transakce. Zpráva je znovu zpřístupněna, pokud je jednotka práce zálohována.

Není-li tato volba nebo GMNSYP uvedena, požadavek na získání není v rámci transakce.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWFCH.
- GMBRWC
- GMBRWN
- GMLK
- GMNSYP
- GMPSYP
- GMUNK

GMPSYP
Získat zprávu s řízením synchronizačního bodu, je-li zpráva trvalá.

Požadavek má fungovat v rámci normálních protokolů jednotky práce, ale pouze tehdy, je-li zpráva načtená, trvalá. Trvalá zpráva má hodnotu PEPER v poli *MDPER* v produktu MQMD.

- Je-li zpráva trvalá, bude správce front zpracovávat volání, jako by aplikace měla zadáno GMSYP.
- Pokud zpráva není trvalá, správce front zpracuje volání, jako by aplikace měla určený GMNSYP (podrobnosti naleznete v následující sekci).

Tato volba není platná s žádnou z následujících voleb:

- GMBRWFCH.
- GMBRWC
- GMBRWN
- GMCMPM
- GMNSYP
- GMSYP
- GMUNK

GMNSYP
Získat zprávu bez řízení synchronizačního bodu.

Požadavek má fungovat mimo běžné protokoly jednotek práce. Zpráva se okamžitě odstraní z fronty (pokud se nejedná o požadavek na procházení). Zprávu nelze znovu zpřístupnit tak, že zazálohujete jednotku práce.

Tato volba se předpokládá, pokud je zadán GMBRWF nebo GMBRWN.

Pokud tato volba a GMSYP nejsou uvedeny, požadavek na získání není součástí pracovní jednotky.

Tato volba není platná s žádnou z následujících voleb:

- GMSYP
- GMPSYP

Volby procházení: Následující volby se vztahují k procházení zpráv ve frontě:

GMBRWFCH.
Procházet od začátku fronty.

Když je fronta otevřena s volbou OOBRW, je umístěn kurzor procházení, umístěný logicky před první zprávou ve frontě. Následná volání MQGET určující volbu GMBRWF, GMBRWN nebo GMBRWC lze použít k načtení zpráv z fronty nedestruktivně. Přehledující kurzor označuje pozici ve zprávách ve frontě, od které další volání MQGET s GMBRWN vyhledá vhodnou zprávu.

Volání MQGET s GMBRWF způsobí, že předchozí pozice kurzoru procházení bude ignorována. Načítá se první zpráva ve frontě, která splňuje podmínky uvedené v deskriptoru zpráv. Zpráva zůstává ve frontě a kurzor procházení je umístěn na této zprávě.

Po tomto volání je kurzor procházení umístěn ve zprávě, která byla vrácena. Je-li zpráva odebrána z fronty před tím, než bude vydána další volání MQGET s GMBRWN, kurzor procházení zůstane na pozici ve frontě, kterou zpráva obsazovala, i když je tato pozice nyní prázdná.

Volbu GMMUC lze poté použít s voláním MQGET bez procházení, je-li to nutné, a odebrat tak zprávu z fronty.

Kurzor procházení se nepřesunul pomocí volání příkazu MQGET bez procházení pomocí stejného popisovače *HOBJ* . Nepřesunuje se ani při procházení voláním MQGET, které vrací kód dokončení CCFAIL, nebo kód příčiny RC2080 .

Volba GMLK může být určena společně s touto volbou, aby se zablokovala zpráva, která je procházena.

GMBRWF může být zadán s jakoukoli platnou kombinací voleb GM* a MO*, které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Je-li uveden GMLOGO, zprávy jsou procházeny v logickém pořadí. Je-li tato volba vynechána, budou zprávy zkontrolovány ve fyzickém pořadí. Je-li uveden GMBRWF, je možné přepínat mezi logickým pořadím a fyzickým pořadím, ale následné volání MQGET pomocí GMBRWN musí procházet frontu ve stejném pořadí, jako je nejnovější volání, které uvádí GMBRWF pro popisovač fronty.

Informace o skupinách a segmentech, které správce front uchovává pro volání MQGET a procházení zpráv ve frontě, je oddělen od informací o skupině a segmentu, které správce front uchovává pro volání MQGET, která odebírá zprávy z fronty. Je-li uveden GMBRWF, správce front ignoruje informace o skupině a segmentu pro procházení a prohledá frontu, jako by neexistovala žádná aktuální skupina a žádná aktuální logická zpráva. Je-li volání MQGET úspěšné (kód dokončení CCOK nebo CCWARN), informace o skupině a segmentu pro procházení se nastaví na vrácenou zprávu; pokud se volání nezdaří, informace o skupině a segmentu zůstanou stejné, jako před voláním.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWC
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNK

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení.

GMBRWN
Procházet z aktuální pozice ve frontě.

Kurzor procházení je zálohován na další zprávu ve frontě, která splňuje kritéria výběru zadaná ve volání MQGET. Zpráva se vrátí do aplikace, ale zůstane ve frontě.

Po otevření fronty pro procházení má první volání procházení s použitím manipulátoru stejný účinek, ať už určuje volbu GMBRWF nebo GMBRWN.

Je-li zpráva odebrána z fronty před tím, než se vydá další volání MQGET s GMBRWN, kurzor procházení logicky zůstává na pozici ve frontě, ve které byla zpráva obsazena, i když je tato pozice nyní prázdná.

Zprávy jsou ukládány do fronty jedním ze dvou způsobů:

- FIFO v rámci priority (MCSPO), nebo
- FIFO bez ohledu na prioritu (MSFIFO)

Atribut fronty **MsgDeliverySequence** označuje, která metoda se použije (podrobnosti viz “Atributy pro fronty” na stránce 1353).

Pokud má fronta *MsgDeliverySequence* MSPRIO a zpráva dorazí do fronty, která má vyšší prioritu než ta, na kterou momentálně odkazuje kurzor procházení, tato zpráva nebyla nalezena během aktuálního procházení fronty pomocí GMBRWN. Může být nalezen pouze poté, co byl kurzor procházení obnoven s GMBRWF (nebo opětovným otevřením fronty).

Volbu GMMUC lze později použít při neprocházení volání MQGET, je-li to nutné, aby mohla být zpráva odebrána z fronty.

Kurzor procházení se nepřesunuje pomocí volání MQGET bez procházení pomocí stejného popisovače *HOBJ* .

Volba GMLK může být určena společně s touto volbou, aby se zablokovala zpráva, která je procházena.

GMBRWN lze zadat s jakoukoli platnou kombinací voleb GM* a MO*, které řídí zpracování zpráv ve skupinách a segmentech logických zpráv.

Je-li uveden GMLOGO, zprávy jsou procházeny v logickém pořadí. Je-li tato volba vynechána, budou zprávy zkontrolovány ve fyzickém pořadí. Je-li uveden GMBRWF, je možné přepínat mezi logickým pořadím a fyzickým pořadím, ale následné volání MQGET pomocí GMBRWN musí procházet frontu ve stejném pořadí, jako je nejnovější volání, které uvádí GMBRWF pro popisovač fronty. Volání selže s kódem příčiny RC2259 , pokud tato podmínka není splněna.

Poznámka: Speciální péče je nutná, pokud se volání MQGET používá k procházení za koncem skupiny zpráv (nebo logické zprávy, která není ve skupině), když není uveden GMLOGO. Pokud například poslední zpráva ve skupině bude předcházet první zprávě ve skupině ve frontě, pomocí GMBRWN pro procházení za koncem skupiny by uvedení MOSEQN s *MDSEQ* nastaveným na 1 (nalezení první zprávy další skupiny) vrátilo znovu první zprávu ve skupině, která již byla procházena. K tomu může dojít okamžitě nebo k několika dalším voláním MQGET (pokud jsou mezi nimi nějaké vedlejší skupiny).

Možnost nekonečné smyčky se lze vyhnout tak, že otevřete frontu dvakrát pro procházení:

- Použijte první popisovač k procházení pouze první zprávy v každé skupině.
- Druhý ovladač použijte k procházení pouze zpráv v rámci určité skupiny.
- Použijte volby MO* k přesunu druhého kurzoru pro procházení na pozici prvního kurzoru pro procházení před prohlížením zpráv ve skupině.
- Nepoužívejte GMBRWN k procházení za koncem skupiny.

Informace o skupinách a segmentech, které správce front uchovává pro volání MQGET, které procházejí zprávy ve frontě, jsou oddělena od informací o skupině a segmentu, které uchovává pro volání MQGET, která odebírá zprávy z fronty.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWFCH.
- GMBRWC
- GMMUC
- GMSYP
- GMPSYP
- GMUNK

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení.

GMBRWC
Procházet zprávu pod kurzorem procházení.

Tato volba způsobí, že zpráva, na kterou se odkazuje kurzor procházení, bude nedestruktivně načtena bez ohledu na volby MO* uvedené v poli *GMMO* v MQGMO.

Zpráva, na kterou ukazuje procházení kurzorem, je ta, která byla naposledy načtena buď pomocí volby GMBRWF, nebo GMBRWN. Volání se nezdaří, pokud ani jedna z těchto volání nebyla pro tuto frontu vydána od jeho otevření, nebo pokud byla zpráva pod kurzorem procházení od té doby destruktivně načtena.

Poloha kurzoru procházení se při tomto volání nezmění.

Volbu GMMUC lze poté použít s voláním MQGET bez procházení, je-li to nutné, a odebrat tak zprávu z fronty.

Kurzor procházení se nepřesunul pomocí volání příkazu MQGET bez procházení pomocí stejného popisovače *HOBJ* . Nepřesunuje se ani při procházení voláním MQGET, které vrací kód dokončení CCFAIL, nebo kód příčiny RC2080.

Je-li GMBRWC uvedeno s GMLK:

- Pokud je již zpráva uzamčena, musí být pod kurzorem, takže je vrácena bez odemknutí a odemknutí; zpráva zůstane uzamknuta.
- Pokud není zamknuta žádná zpráva, je zpráva pod kurzorem procházení (pokud existuje) uzamčena a vrácena do aplikace; pokud v rámci kurzoru procházení není žádná zpráva, volání selže.

Je-li GMBRWC uvedeno bez GMLK:

- Je-li již zpráva uzamknuta, musí být pod kurzorem. Tato zpráva je vrácena do aplikace a poté odemknuta. Vzhledem k tomu, že zpráva je nyní odemknuta, neexistuje žádná záruka, že ji lze znovu procházet nebo načíst destruktivně (může být načítána destruktivně jinou aplikací získávajícím zprávy z fronty).
- Pokud zpráva neobsahuje žádnou zamknutou zprávu, zobrazí se zpráva pod kurzorem (pokud je zde jedna) vrácena aplikaci; pokud není pod kurzorem procházení, volání selže.

Je-li GMCMPM zadán společně s GMBRWC, musí kurzor procházení identifikovat zprávu s polem *MDOFF* v deskriptoru MQMD, který je nulový. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2246 .

Informace o skupinách a segmentech, které správce front uchovává pro volání MQGET, které procházejí zprávy ve frontě, jsou oddělena od informací o skupině a segmentu, které uchovává pro volání MQGET, která odebírá zprávy z fronty.

Tato volba není platná s žádnou z následujících voleb:

- GMBRWFCH.
- GMBRWN
- GMMUC
- GMSYP
- GMPSYP
- GMUNK

Jedná se také 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 se odebere z fronty.

Zpráva, na kterou ukazuje procházení kurzorem, je ta, která byla naposledy načtena buď pomocí volby GMBRWF, nebo GMBRWN.

Je-li GMCMPM zadán společně s GMMUC, musí kurzor procházení identifikovat zprávu s polem *MDOFF* v deskriptoru MQMD, který má hodnotu nula. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2246 .

Tato volba není platná s žádnou z následujících voleb:

- GMBRWFCH.
- GMBRWC
- GMBRWN
- GMUNK

Jedná se také o chybu, pokud nebyla fronta otevřena pro procházení i pro vstup. Pokud kurzor procházení aktuálně neukazuje na zprávu s možností načtení, vrátí se chyba volání MQGET.

Volby uzamčení: Následující volby se vztahují k zamykání zpráv ve frontě:

GMLK
Uzamknout zprávu.

Tato volba uzamkne zprávu, která je procházena, takže se zpráva stane neviditelnou pro všechny ostatní ovladače otevřené pro danou frontu. Volbu lze zadat pouze v případě, že je zadána také jedna z následujících voleb:

- GMBRWFCH.
- GMBRWN
- GMBRWC

Pro popisovač fronty může být uzamčena pouze jedna zpráva, ale může se jednat o logickou zprávu nebo o fyzickou zprávu:

- Je-li uvedena hodnota GMCMPM, všechny segmenty zprávy, které tvoří logickou zprávu, jsou uzamčeny ke zpracování fronty (pokud jsou všechny přítomné ve frontě a jsou k dispozici pro načtení).
- Není-li parametr GMCMPM zadán, je pro popisovač fronty zamknuta pouze jedna fyzická zpráva. Pokud se tato zpráva stane segmentem logické zprávy, uzamčený segment zabraňuje ostatním aplikacím používám GMCMPM k načtení nebo procházení logické zprávy.

Zamknutá zpráva je vždy ta pod kurzorem procházení a zpráva může být odebrána z fronty pozdějším voláním MQGET, které určuje volbu GMMUC. Ostatní volání MQGET používající obslužnou rutinu fronty mohou také odebrat zprávu (například volání, které určuje identifikátor zprávy zamčené zprávy).

Pokud volání vrátí kód dokončení CCFAIL nebo CCWARN s kódem příčiny RC2080, žádná zpráva se nezamkne.

Pokud se aplikace rozhodne neodebrat zprávu z fronty, zámek se uvolní pomocí:

- Vydává se další volání MQGET pro tento popisovač, buď s uvedeným GMBRWF nebo GMBRWN (s nebo bez GMLK); zpráva se odemkne, pokud se volání dokončí s CCOK nebo CCWARN, ale zůstává uzamčeno, pokud je volání dokončeno s CCFAIL. Platí však následující výjimky:
  - Zpráva není odemknuta, pokud je CCWARN vrácen s kódem RC2080.
  - Zpráva je odemknuta, pokud je CCFAIL vrácen s kódem RC2033.

  Je-li uvedeno také GMLK, je vrácená zpráva zamčena. Není-li parametr GMLK zadán, nebude po volání žádná zamčená zpráva.

  Je-li uvedena hodnota GMWT a žádná zpráva není okamžitě k dispozici, dojde k odemknutí původní zprávy před začátkem čekání (za předpokladu, že volání je jinak volné z chyby).
- Vydáním dalšího volání MQGET pro tento popisovač s GMBRWC (bez GMLK); zpráva se odemkne, je-li volání dokončeno s CCOK nebo CCWARN, ale zůstává uzamčeno, je-li volání dokončeno s CCFAIL. Platí však následující výjimka:
  - Zpráva není odemknuta, pokud je CCWARN vrácen s kódem RC2080.
- Vydáním dalšího volání MQGET pro tento popisovač s GMUNLK.
- Vydáním volání MQCLOSE pro tento popisovač (ať už explicitně, nebo implicitně ukončením aplikace).

Pro zadání této volby není vyžadována žádná speciální volba otevření, jiná než OOBRW, která je nutná k tomu, aby bylo možné zadat doprovodnou volbu procházení.

Tato volba není platná s žádnou z následujících voleb:

- GMSYP
- GMPSYP
- GMUNK

GMUNK
Odemknout zprávu.

Zpráva, která má být odemknuta, musí být dříve zamčena voláním MQGET s volbou GMLK. Pokud pro tento popisovač není uzamčena žádná zpráva, volání bude dokončeno s CCWARN a RC2209 .

Parametry **MSGDSC**, **BUFLEN**, **BUFFER**a **DATLEN** se nekontrolují ani nemění, je-li uveden GMUNLK. V produktu *BUFFER*není vrácena žádná zpráva.

Pro uvedení této volby není požadována žádná speciální volba otevření (ačkoli OOBRW je zapotřebí k vydání požadavku na uzamčení na prvním místě).

Tato volba není platná s žádnými volbami kromě následujících:

- GMNWT.
- GMNSYP

Obě tyto možnosti se předpokládají bez ohledu na to, zda jsou zadány nebo ne.

Volby datové zprávy: Následující volby se vztahují ke zpracování dat zprávy, když je zpráva přečtena z fronty:

GMATM
Povolit oříznutí dat zprávy.

Je-li vyrovnávací paměť zpráv příliš malá, aby mohla obsahovat úplnou zprávu, umožní tato volba volání MQGET zaplňovat vyrovnávací paměť tak velkou část zprávy, jakou může vyrovnávací paměť zadržet, vydat kód pro dokončení varování a dokončit zpracování. To znamená:

- Při procházení zpráv je kurzor procházení pro vracenou zprávu rozšířený.
- Při odebírání zpráv je vrácená zpráva odebrána z fronty.
- Kód příčiny RC2079 je vrácen, pokud se nevyskytne jiná chyba.

Bez této volby je vyrovnávací paměť stále zaplněna jako velká část zprávy, jak může zadržet, kód varování dokončení je vydán, ale zpracování není dokončeno. To znamená:

- Při procházení zpráv není kurzor procházení pokročilý.
- Při odebírání zpráv se zpráva neodebere z fronty.
- Kód příčiny RC2080 je vrácen, pokud se nevyskytne jiná chyba.

GMCONSOU-
Převést data zprávy.

Tato volba požaduje převedení dat aplikace ve zprávě, aby vyhovovala hodnotám *MDCSI* a *MDENC* uvedeným v parametru **MSGDSC** na volání MQGET, než se data zkopírují do parametru **BUFFER** .

Pole *MDFMT* zadané při vložení zprávy je předpokládáno procesem převodu za účelem identifikace povahy dat ve zprávě. Převod dat zprávy je správcem front pro vestavěné formáty a uživatelem napsaným výstupem pro ostatní formáty.

- Pokud je převod proveden úspěšně, pole *MDCSI* a *MDENC* zadaná v parametru **MSGDSC** se nezmění při návratu z volání MQGET.
- Pokud nelze převod provést úspěšně (ale volání MQGET se jinak dokončí bez chyby), data zprávy se vrátí nepřevedena a pole *MDCSI* a *MDENC* v souboru *MSGDSC* jsou nastaveny na hodnoty pro nekonvertované zprávy. Kód dokončení je CCWARN v tomto případě.

V obou případech tato pole popisují identifikátor znakové sady a kódování dat zprávy, která se vrací v argumentu **BUFFER** .

Seznam názvů formátů, pro které správce front provádí převod, naleznete v poli *MDFMT* , které je popsáno v “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099 .

Volby skupiny a segmentu: Následující volby se vztahují ke zpracování zpráv ve skupinách a segmentech logických zpráv. Tyto definice mohou být užitečné při pochopení možností:

Fyzická zpráva
Jedná se o nejmenší jednotku informací, které lze umístit do fronty nebo z ní odebrat; často odpovídá informacím zadaným nebo načteným při volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zprávy (MQMD). Obecně jsou fyzické zprávy rozlišeny odlišnými hodnotami identifikátoru zprávy (pole*MDMID* v MQMD), ačkoli správce front toto není vynucen.

Logická zpráva
Jedná se o jedinou jednotku informací o aplikaci. V případě neexistence systémových omezení by logická zpráva byla stejná jako fyzická zpráva. Pokud jsou však logické zprávy velké, omezení systému by mohla učinit vhodné nebo nezbytné k rozdělení logické zprávy do dvou nebo více fyzických zpráv, nazývaných segmenty.

Logická zpráva, která byla rozdělena na segmenty, se skládá ze dvou nebo více fyzických zpráv, které mají stejný identifikátor skupiny bez hodnoty null (pole*MDGID* v MQMD) a stejné pořadové číslo zprávy (pole*MDSEQ* v MQMD). Segmenty jsou odlišeny lišícími hodnotami pro offset segmentu (pole*MDOFF* v MQMD), který poskytuje odchylku dat ve fyzické zprávě od začátku dat v logické zprávě. Vzhledem k tomu, že každý segment je fyzická zpráva, segmenty v logické zprávě mají obvykle odlišné identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale jejíž segmentace byla povolena odesílající aplikací, má také neprázdný identifikátor skupiny, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s identifikátorem skupiny, pokud tato logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které má být segmentace zablokována 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 identifikátor skupiny bez hodnoty null. Logické zprávy ve skupině jsou rozlišeny odlišnými hodnotami pro pořadové číslo zprávy, což je celé číslo v rozsahu od 1 do n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentovaná, ve skupině jsou více než n fyzických zpráv.

GMLOGO
Zprávy ve skupinách a segmentech logických zpráv se vrací v logickém pořadí.

Tato volba určuje pořadí, ve kterém jsou zprávy vraceny po sobě jdoucími voláními MQGET pro manipulátor fronty. Volba musí být uvedena u každé z těchto volání, aby měla efekt.

Je-li GMLOGO uveden pro následné volání MQGET pro obsluhu fronty, zprávy ve skupinách jsou vraceny v pořadí uvedeném pořadovými čísly zpráv a segmenty logických zpráv jsou vraceny v pořadí poskytnutému jejich segmentem segmentu. Toto pořadí se může lišit od pořadí, ve kterém se tyto zprávy a segmenty vyskytují ve frontě.

Poznámka: Uvedení GMLOGO nemá žádné nepříznivé důsledky na zprávy, které nepatří do skupin a které nejsou segmenty. V důsledku toho se s takovými zprávami zachází, jako by každá patřila do skupiny zpráv skládající se pouze z jedné zprávy. Proto je naprosto bezpečné uvést GMLOGO při načítání zpráv z front, které mohou obsahovat směs zpráv ve skupinách, segmentech zpráv a nesegmentovaných zpráv, které nejsou ve skupinách.

Chcete-li zprávy vrátit v požadovaném pořadí, zachová správce front 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 to, zda jsou zprávy načítány v rámci jednotky práce. Vzhledem k tomu, že správce front uchovává tyto informace, nemusí aplikace před každým voláním MQGET nastavovat informace o skupině a segmentu. Konkrétně to znamená, že aplikace nemusí nastavovat pole *MDGID*, *MDSEQ*a *MDOFF* v MQMD. Aplikace však musí správně nastavit volbu GMSYP nebo GMNSYP na každém volání.

Když je fronta otevřena, neexistuje žádná aktuální skupina zpráv a žádná aktuální logická zpráva. Skupina zpráv se stane aktuální skupinou zpráv, je-li příkazem MQGET vrácena zpráva s příznakem MMMIG. S GMLOGO zadané v následných voláních, tato skupina zůstává 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 taková zpráva vrácena, skupina zpráv je ukončena a při úspěšném dokončení tohoto volání MQGET již neexistuje aktuální skupina. Podobně se logická zpráva stane aktuální logickou zprávou,

když se vrátí zpráva s příznakem MFSEG a tato logická zpráva je ukončena, když je vrácena zpráva, která má příznak MFLSEG.

Nejsou-li uvedena žádná kritéria výběru, za sebou následují volání MQGET (ve správném pořadí) zprávy pro první skupinu zpráv ve frontě, pak zprávy pro druhou skupinu zpráv, a tak dále, dokud nebudou k dispozici žádné další zprávy. Konkrétní skupiny zpráv lze vybrat zadáním jedné nebo více z následujících voleb do pole *GMMO* :

- MOMGI
- MODORŠTINA
- 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 703 na stránce 1078 zobrazuje hodnoty polí *MDMID*, *MDCID*, *MDGID*, *MDSEQ*a *MDOFF* , které správce front hledá při pokusu o nalezení zprávy, která má být vrácena na volání MQGET. To platí jak pro odstranění zpráv z fronty, tak pro procházení zpráv ve frontě. Sloupce v tabulce mají následující význam:

PROTOKOL
Označuje, zda je volba GMLOGO uvedená na volání.

Cur grp
Indikuje, zda před voláním existuje aktuální skupina zpráv.

Zpráva protokolu cur
Indikuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazení hodnot, které správce front hledá. "Předchozí" označuje hodnotu vrácenou pro pole v předchozí zprávě pro popisovač fronty.

Tabulka 703. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv

Volby, které uvedete	**Stav skupiny a protokolu-zpráva před voláním**		**Hodnoty, které správce front hledá**				
LOG SLOVO	**Cur grp**	**Zpráva Cur msg**	*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 korelační identifikátor	Předchozí identifikátor skupiny	1	Předchozí odchylka + předchozí délka segmentu
Ano	Ano	Ne	Libovolný identifikátor zprávy	Libovolný korelační identifikátor	Předchozí identifikátor skupiny	Předchozí pořadové číslo + 1	0
Ano	Ano	Ano	Libovolný identifikátor zprávy	Libovolný korelační identifikátor	Předchozí identifikátor skupiny	Předchozí pořadové číslo	Předchozí odchylka + předchozí délka segmentu

Tabulka 703. Volby MQGET související se zprávami ve skupinách a segmentech logických zpráv (pokračování)							
Volby, které uvedete	**Stav skupiny a protokolu-zpráva před voláním**		**Hodnoty, které správce front hledá**				
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 a které 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 zpráv 1, a posuny o 0, určují pořadí, ve kterém jsou způsobilé skupiny vráceny).

Volba GMLOGO ovlivňuje jednotky práce takto:

- Je-li 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, je-li použita stejná obsluha fronty. Avšak nemusí být načteny v rámci stejné jednotky práce. To umožňuje skupině zpráv skládající se z mnoha fyzických zpráv, které mají být rozděleny do dvou nebo více po sobě jdoucích jednotek práce pro manipulátor fronty.
- Pokud se první logická zpráva nebo segment ve skupině nenačte v rámci pracovní jednotky, žádná z ostatních logických zpráv a segmentů ve skupině nemůže být načtena v rámci pracovní jednotky, pokud se používá stejný popisovač fronty.

Nejsou-li tyto podmínky splněny, volání MQGET selže s kódem příčiny RC2245 .

Je-li uveden GMLOGO, MQGMO dodaný na volání MQGET nesmí být menší než GMVER2a MQMD nesmí být menší než MDVER2. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2256 nebo RC2257 podle potřeby.

Není-li GMLOGO uveden pro následné volání MQGET pro obsluhu fronty, jsou zprávy vraceny bez ohledu na to, zda patří do skupin zpráv, nebo zda jsou segmenty logických zpráv. To znamená, že zprávy nebo segmenty z určité skupiny nebo logické zprávy mohou být vráceny mimo pořadí, nebo mohou být intermingované 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 vrácené po sobě jdoucími voláními MQGET řízeny volbami MO* uvedenými v těchto voláních (viz pole *GMMO* popsané v části “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066 , kde jsou uvedeny podrobnosti o těchto volbách).

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy ve středu, po selhání systému. Když se systém restartuje, může aplikace nastavit pole *MDGID*, *MDSEQ*, *MDOFF*a *GMMO* na odpovídající hodnoty a pak vydat volání MQGET s GMSYP nebo GMNSYP nastavenou podle potřeby, ale bez uvedení GMLOGO. Je-li toto volání úspěšné, správce front zachová informace o skupině a segmentu a následující volání MQGET s použitím manipulátoru fronty mohou určovat GMLOGO jako normální.

Informace o skupinách a segmentech, které správce front uchovává pro volání MQGET, jsou odděleny od informace o skupině a segmentu, které si uchovává pro volání MQPUT. Kromě toho správce front uchovává samostatné informace pro:

- Volání MQGET, které odebírá zprávy z fronty.
- Volání MQGET, které prochází zprávy ve frontě.

Pro daný popisovač fronty je aplikace volná pro kombinaci volání MQGET, které určují GMLOGO s voláními MQGET, které nikoli, ale musí být zaznamenány následující body:

- Není-li GMLOGO zadán, každá úspěšná volání MQGET způsobí, že správce front nastaví informace o uložené skupině a segmentu na hodnoty odpovídající vrácené zprávě. To nahradí

existující informace o skupině a segmentech zachované správcem front pro manipulátor fronty. Upravovány jsou pouze informace odpovídající akci volání (procházení nebo odebrání).

- Není-li GMLOGO uveden, volání se nezdaří, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání však může být úspěšné s kódem dokončení CCWARN. Tabulka 704 na stránce 1080 zobrazuje různé případy, které mohou nastat. V těchto případech, je-li kód dokončení není CCOK, kód příčiny je 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 ne vstup; v těchto případech je kód dokončení vždy CCOK (za předpokladu, že nejsou žádné jiné chyby).

Tabulka 704. 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	CCFIL	CCFIL
MQGET bez GMLOGO	CCWARN	KEK
MQCLOSE s neukončené skupinou nebo logickou zprávou	CCWARN	KEK

Aplikace, které pouze chtějí načítat zprávy a segmenty v logickém pořadí, se doporučuje uvést GMLOGO, protože se jedná o nejjednodušší volbu, která se má použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupinách a segmentech, protože tyto informace spravuje správce front. Avšak specializované aplikace mohou vyžadovat více kontroly, než je poskytnuto volbou GMLOGO, a toho lze dosáhnout, když tuto volbu nezadáte. Po provedení této operace musí aplikace zajistit, aby pole *MDMID*, *MDCID*, *MDGID*, *MDSEQ*a *MDOFF* v produktu MQMD a volby MO* v produktu *GMMO* v produktu MQGMO byly nastaveny správně, před každým voláním MQGET.

Například aplikace, která chce předávat fyzické zprávy, které přijímá, bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, by neměla uvádět GMLOGO. Důvodem je to, že ve složité síti s více cestami mezi odesílajícím a přijímajícím správcem front může dojít k nedostatku fyzických zpráv v pořadí. Neuvedete-li GMLOGO a odpovídající PMLOGO na volání MQPUT, může přesměrovací aplikace načítat a předávat každou fyzickou zprávu ihned, jakmile dorazí, aniž by musela čekat na to, až se objeví další logická zpráva, která přijde.

GMLOGO lze zadat s libovolnými z dalších voleb GM* a s různými volbami MO* za odpovídajících okolností.

GMCMPM
Lze načíst pouze úplné logické zprávy.

Tato volba určuje, že se voláním MQGET může vrátit pouze úplná logická zpráva. Je-li logická zpráva segmentovaná, správce front znovu složí segmenty a vrátí k aplikaci úplnou logickou zprávu; skutečnost, že logická zpráva byla segmentována, není zřejmé, že by aplikace načítala tuto zprávu.

Poznámka: Toto je jediná volba, která způsobí, že správce front znovu sestaví segmenty zpráv. Pokud nejsou uvedeny, segmenty se vrátí jednotlivě do aplikace, pokud jsou přítomny ve frontě (a vyhovují dalším kritériím výběru uvedeným na volání MQGET). Aplikace, které nechtějí přijímat jednotlivé segmenty, by proto měly vždy uvádět GMCMPM.

Chcete-li použít tuto volbu, aplikace musí poskytovat vyrovnávací paměť, která je dostatečně velká, aby pojmula úplnou zprávu, nebo uveďte volbu GMATM.

Pokud fronta obsahuje segmentované zprávy s některými chybějícími segmenty (například proto, že byly v síti zpožděny a dosud nedorazili), uvedení GMCMPM brání načtení segmentů náležejících k neúplným logickým zprávám. Tyto segmenty zpráv však stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné obnovitelné logické zprávy, i když je *CurrentQDepth* větší než nula.

Pro trvalé zprávy může správce front znovu sestavit segmenty pouze v rámci pracovní jednotky:

- Je-li volání MQGET provozováno v rámci uživatelské jednotky práce, použije se tato jednotka práce. Pokud volání selže v rámci procesu opětovného sestavení, správce front znovu uvede všechny segmenty, které byly odebrány během opětovného sestavení. Selhání však nezabrání úspěšnému potvrzení jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelsky definovaná jednotka práce, správce front vytvoří pracovní jednotku pouze po dobu trvání hovoru. Je-li volání úspěšné, správce front automaticky potvrdí jednotku práce (aplikace ji nemusí provést). Pokud se volání nezdaří, správce front provede zálohu jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce, ale uživatelem definovaná jednotka práce existuje, správce front nemůže provést opětovné sestavení. Pokud zpráva nevyžaduje opětovnou montáž, volání může být stále úspěšné. Pokud však zpráva vyžaduje opětovnou montáž, volání selže s kódem příčiny RC2255 .

V případě přechodných zpráv správce front nevyžaduje, aby byla k dispozici jednotka práce, aby bylo možné provést opětovné sestavení.

Každá fyzická zpráva, která má segment, má svůj vlastní deskriptor zprávy. Pro segmenty tvořící jedinou logickou zprávu je většina polí v deskriptoru zpráv stejná pro všechny segmenty v logické zprávě-obvykle se jedná pouze o pole *MDMID*, *MDOFF*a *MDMFL* , která se liší mezi segmenty v logické zprávě. Je-li však segment umístěn ve frontě nedoručených zpráv ve středním správci front, načte obslužná rutina DLQ zprávu specifikující volbu GMCONV, což může mít za následek změnu znakové sady nebo kódování právě měněného segmentu. Pokud obslužná rutina DLQ úspěšně odešle segment na jeho způsob, segment může mít znakovou sadu nebo kódování, které se liší od ostatních segmentů v logické zprávě, když se segment konečně dostane do cílového správce front.

Logická zpráva skládající se ze segmentů, ve kterých se pole *MDCSI*, *MDENC*nebo obě pole nemohou znovu sestavit do jedné logické zprávy ze strany správce front. Místo toho správce front znovu sestaví a vrátí prvních několik po sobě jdoucích segmentů na začátku logické zprávy se stejnými identifikátory kódování a kódování a volání MQGET se dokončí s kódem dokončení CCWARN a kódem příčiny RC2243 nebo RC2244 podle potřeby. K tomu dojde bez ohledu na to, zda je uveden GMCONV. Chcete-li načíst zbývající segmenty, aplikace musí znovu zadat volání MQGET bez volby GMCMPM, aby načítal segmenty jednu po druhé. GMLOGO lze použít k načtení zbývajících segmentů v pořadí.

Je také možné pro aplikaci, která ukládá segmenty pro nastavení jiných polí v deskriptoru zpráv na hodnoty, které se liší mezi segmenty. K tomu však není žádná výhoda, pokud přijímající aplikace používá GMCMPM k načtení logické zprávy. Když správce front znovu složí logickou zprávu, vrací v deskriptoru zpráv hodnoty z deskriptoru zpráv pro první segment; jedinou výjimkou je pole *MDMFL* , které nastavuje správce front, aby indikoval, že opětovně sestavená zpráva je jediným segmentem.

Je-li pro zprávu sestavy uveden GMCMPM, provede správce front speciální zpracování. Správce front danou frontu zkontroluje a zjišťuje, zda jsou ve frontě všechny zprávy sestavy daného typu týkající se různých segmentů v logické zprávě. Pokud jsou, mohou být načteny jako jediná zpráva uvedením GMCMPM. Aby to bylo možné, zprávy sestavy musí být generovány správcem front nebo agentem MCA, který podporuje segmentaci, nebo musí původní aplikace vyžadovat alespoň 100 bajtů dat zprávy (to znamená, že příslušné volby RO* D nebo RO* F musí být zadány). Je-li pro segment méně než zaplněno celé množství dat aplikace, chybějící bajty se nahradí hodnotami null ve vrácené zprávě sestavy.

Je-li GMCMPM zadán s GMMUC nebo GMBRWC, musí být kurzor procházení umístěn na zprávě s polem *MDOFF* v produktu MQMD, který má hodnotu 0. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2246 .

GMCMPM znamená GMASGA, což nemusí být tedy zadáno.

GMCMPM může být zadán spolu s libovolnou z dalších voleb GM* kromě GMPSYP a s libovolnou volbou MO* kromě MOOFFS.

GMAMSA
Všechny zprávy ve skupině musí být dostupné.

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 z chybějících zpráv (například proto, že byly v síti zpožděny a ještě nedorazili), uvedení GMAMSA zabrání načtení zpráv náležejících do neúplných skupin. Tyto zprávy však stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné skupiny zpráv, které lze načíst, ačkoli **CurrentQDepth** je větší než nula. Nejsou-li k dispozici žádné další zprávy, které lze načíst, je po uplynutí zadané čekací doby (pokud existuje) vrácen kód příčiny RC2033 .

Zpracování GMAMSA závisí na tom, zda je GMLOGO uveden také:

- Pokud jsou zadány obě volby, GMAMSA ovlivní pouze v případě, že neexistuje žádná aktuální skupina nebo logická zpráva. Existuje-li aktuální skupina nebo logická zpráva, GMAMSA se ignoruje. To znamená, že GMAMSA může zůstat při zpracování zpráv v logickém pořadí zpracování.
- Je-li GMAMSA zadán bez GMLOGO, bude mít GMAMSA vždy efekt. To znamená, že po odebrání první zprávy ve skupině z fronty musí být tato volba vypnuta, 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. Avšak mějte na paměti, že jiné aplikace jsou stále schopny odebrat zprávy ze skupiny (skupina není zamknuta na aplikaci, která načte první zprávu ve skupině).

Není-li tato volba uvedena, mohou být zprávy náležící do skupin načteny i v případě, že je skupina neúplná.

GMAMSA znamená GMASGA, což nemusí být přesně uvedeno.

GMAMSA může být zadán s libovolnou z dalších voleb GM* a s libovolnou z voleb MO*.

GMASGA
Všechny segmenty v logické zprávě musí být dostupné.

Tato volba uvádí, že segmenty v logické zprávě budou k dispozici pro načtení pouze tehdy, jsou-li k dispozici všechny segmenty v logické zprávě. Pokud fronta obsahuje segmentované zprávy s některými chybějícími segmenty (snad proto, že byly v síti zpožděny a ještě nedorazili), uvedení GMASGA brání načtení segmentů náležejících k neúplným logickým zprávám. Nicméně tyto segmenty stále přispívají k hodnotě atributu fronty produktu **CurrentQDepth** ; to znamená, že mohou existovat žádné obnovitelné logické zprávy, i když je **CurrentQDepth** větší než nula. Nejsou-li k dispozici žádné další zprávy, které lze načíst, je po uplynutí zadané čekací doby (pokud existuje) vrácen kód příčiny RC2033 .

Zpracování GMASGA závisí na tom, zda je GMLOGO uveden také:

- Jsou-li obě volby zadány, GMASGA má efekt pouze v případě, že neexistuje žádná aktuální logická zpráva. Pokud se jedná o aktuální logickou zprávu, GMASGA se ignoruje. To znamená, že GMASGA může zůstat i při zpracování zpráv v logickém pořadí.
- Je-li GMASGA zadáno bez GMLOGO, GMASGA má vždy efekt. To znamená, že tato volba musí být vypnuta po odebrání prvního segmentu z logické zprávy z fronty, aby bylo možné odebrat zbývající segmenty v logické zprávě.

Není-li tato volba uvedena, lze segmenty zpráv načíst i v případě, že je logická zpráva neúplná.

Přestože GMCMPM i GMASGA vyžadují, aby všechny segmenty byly k dispozici před tím, než může být některý z nich získán, původní zpráva vrací úplnou zprávu, zatímco druhá umožňuje, aby byly segmenty načteny jeden po druhém.

Je-li pro zprávu sestavy uvedena hodnota GMASGA, provede správce front speciální zpracování. Správce front zkontroluje frontu, zda obsahuje alespoň jednu zprávu sestavy pro každý z segmentů, které tvoří úplnou logickou zprávu. Je-li tomu tak, podmínka GMASGA je splněna. Správce front však nekontroluje typ zpráv sestavy a ve zprávách sestav týkajících se segmentů logické zprávy může být ve zprávě sestavy uvedena směs typů sestav. V důsledku toho úspěch GMASGA neznamená, že GMCMPM uspěje. Je-li pro segmenty určité logické zprávy uvedena směs typů sestav, musí být tyto zprávy sestavy načteny jedním po druhém.

GMASGA lze zadat s libovolní z ostatních voleb GM* a s libovolnou z voleb MO*.

Výchozí volba: Pokud není požadována žádná z uvedených voleb, lze použít následující volbu:

GMNONE
Nejsou uvedeny žádné volby.

Tato hodnota může být použita k označení, že nebyly zadány žádné další volby; všechny volby předpokládají jejich výchozí hodnoty. Program GMNONE je definován pro dokumentaci programu podpory; není zamýšlen, že tato volba je použita spolu s jinou hodnotou, ale její hodnota je nula, takové použití nelze zjistit.

Počáteční hodnota pole *GMOPT* je GMNWT.

GMRE1 (1bajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud *GMVER* je menší než GMVER2.

GMRL (10ciferné celé číslo se znaménkem)

Délka vrácených dat zprávy (bajty).

Toto je výstupní pole, které je nastaveno správcem front na délku v bajtech dat zprávy vrácených voláním MQGET v rámci parametru **BUFFER** . Pokud správce front tuto schopnost nepodporuje, nastaví se hodnota *GMRL* na hodnotu RLUNDF.

Jsou-li zprávy převáděny mezi kódováními nebo znakovými sadami, mohou data zprávy někdy měnit velikost. Při návratu z volání MQGET:

- Pokud *GMRL* není RLMUNDF, je počet bajtů vrácených dat zpráv poskytnut *GMRL*.
- Má-li *GMRL* hodnotu RUNDF, je počet bajtů vrácených dat zprávy obvykle dán menším počtem *BUFLEN* a *DATLEN*, ale může být menší než, pokud je volání MQGET dokončeno s kódem příčiny RC2079 . Pokud k tomu dojde, jsou nevýznamné bajty v parametru **BUFFER** nastaveny na hodnoty null.

Je definována následující speciální hodnota:

RLUNDF.
Délka vrácených dat není definována.

Počáteční hodnota tohoto pole je RLMUNDF. Toto pole je ignorováno, pokud je *GMVER* menší než GMVER3.

GMRQN (48 bajtů znakového řetězce)

Vyřešený název cílové fronty.

Jedná se o výstupní pole, které je nastaveno správcem front na lokální název fronty, ze které byla zpráva načtena, jak je definováno v lokálním správci front. To se liší od názvu použitého k otevření fronty, pokud:

- Byla otevřena fronta aliasů (v takovém případě se jedná o název lokální fronty, do které je alias vrácen), nebo

- Byla otevřena modelová fronta (v takovém případě je vrácen název dynamické lokální fronty).

Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

GMRS2 (jednobajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak. Toto pole je ignorováno, pokud *GMVER* je 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
Segmentace je povolena.

Toto je výstupní pole. Počáteční hodnota tohoto pole je SEGIHB. Toto pole je ignorováno, pokud *GMVER* je menší než GMVER2.

GMSG1 (10ciferné celé číslo se znaménkem)

Signál.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

GMSG2 (10ciferné celé číslo se znaménkem)

Identifikátor signálu.

Jedná se o vyhrazené pole; jeho hodnota není významná.

GMSID (4bajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

GMSIDVERB
Identifikátor pro strukturu voleb get-message.

Toto pole je vždy vstupním polem. Počáteční hodnota tohoto pole je GMSIDV.

GMSST (jednobajtový znakový řetězec)

Příznak označující, zda je načtená zpráva segmentem logické zprávy.

Má jednu z následujících hodnot:

SSNSEG
Zpráva není segment.

SSSECHGENERICN
Zpráva je segment, ale nejedná se o poslední segment logické zprávy.

SSLSEG
Zpráva je posledním segmentem logické zprávy.

Tato hodnota je také vrácena, 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 *GMVER* je menší než GMVER2.

GMTOK (16bajtový bitový řetězec)

Token zprávy.

Jedná se o 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 LNMTOK. Počáteční hodnota tohoto pole je MTKNON. Toto pole je ignorováno, pokud je *GMVER* menší než GMVER3.

GMVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

GMVER1
Struktura volby get-message pro objekt Version-1 .

GMVER2
Struktura volby get-message pro objekt Version-2 .

GMVER3
Struktura volby get-message pro objekt Version-3 .

GMVER4
Struktura volby get-message pro objekt Version-4 .

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

GMVERC
Aktuální verze struktury voleb získání zprávy.

Toto pole je vždy vstupním polem. Počáteční hodnota tohoto pole je GMVER1.

GMVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

GMVER1
Struktura volby get-message pro objekt Version-1 .

GMVER2
Struktura volby get-message pro objekt Version-2 .

GMVER3
Struktura volby get-message pro objekt Version-3 .

GMVER4
Struktura volby get-message pro objekt Version-4 .

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

GMVERC
Aktuální verze struktury voleb získání zprávy.

Toto pole je vždy vstupním polem. Počáteční hodnota tohoto pole je GMVER1.

GMWI (10ciferné celé číslo se znaménkem)

Interval čekání.

Toto je přibližná doba, vyjádřená v milisekundách, po kterou volání MQGET čeká na příchod vhodné zprávy (tj. zpráva splňující kritéria výběru zadaná v parametru **MSGDSC** volání MQGET; další podrobnosti viz pole *MDMID* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099). Pokud po uplynutí této doby neuplyne žádná vhodná zpráva, volání skončí s CCFAIL a kódem příčiny RC2033.

GMWI se používá s volbou GMWT. Pokud tato volba není určena, je ignorována. Je-li zadána, hodnota *GMWI* musí být větší než nula nebo rovna nule nebo následující speciální hodnotu:

WIULIM
Neomezený interval čekání.

Počáteční hodnota tohoto pole je 0.

Počáteční hodnoty

Tabulka 705. Počáteční hodnoty polí v MQGMO		
Název pole	**Název konstanty**	**Hodnota konstanty**
GMSID	GMSIDVERB	'GMO¬'
GMVER	GMVER1	1
GMOPT	GMNWT.	0
GMWI	Není	0
GMSG1	Není	0
GMSG2	Není	0
GMRQN	Není	Mezery
GMMO	MOMGI + MOCOREI	3
GMGST	GSNIGA	'¬'
GMSST	SSNSEG	'¬'
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                                     00000000000000000')
     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 (záhlaví informačního obsahuIMS) v systému IBM i

Struktura MQIIH popisuje informace, které musí být přítomny na začátku zprávy odeslané do mostu IMS prostřednictvím produktu IBM MQ for z/OS.

Přehled

Název formátu: FMIMS.

Znaková sada a kódování: Speciální podmínky se vztahují na znakovou sadu a kódování použité pro strukturu MQIIH a data zprávy aplikace:

- Aplikace, které se připojují ke správci front, který vlastní frontu mostu IMS , musí poskytovat strukturu MQIIH, která se nachází ve znakové sadě a kódování správce front. Důvodem je, že převod dat struktury MQIIH se v tomto případě neprovádí.
- Aplikace, které se připojují k jiným správcům front, mohou poskytovat strukturu MQIIH, která se nachází ve všech podporovaných znakových sadách a kódováních; konverze MQIIH je prováděna přijímajícím agentem kanálu zpráv připojeným ke správci front, který vlastní frontu mostu IMS .

  Poznámka: Existuje jedna výjimka. Pokud správce front, který vlastní frontu mostu IMS , používá CICS pro distribuované řazení do fronty, musí být MQIIH ve znakové sadě a kódování správce front, který vlastní frontu mostu IMS .
- Data zprávy aplikace následující za strukturou MQIIH musí být ve stejné znakové sadě a kódování jako struktura MQIIH. Pole *IICSI* a *IIENC* ve struktuře MQIIH nemohou být použity k určení znakové sady a kódování dat zprávy aplikace.

  Uživatelská procedura pro převod dat musí být poskytnuta uživatelem za účelem převodu dat zprávy aplikace, nejsou-li data jedním z vestavěných formátů podporovaných správcem front.

- “Ověřování přístupových hesel pro aplikace mostu IMS” na stránce 1087
- “Pole” na stránce 1088
- “Počáteční hodnoty” na stránce 1091
- “Deklarace RPG” na stránce 1091

Ověřování přístupových hesel pro aplikace mostu IMS

Nyní je možné, aby administrátoři produktu IBM MQ určili název aplikace, která má být použita pro ověřovací tikety, 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 k ověření dojde stejně jako v předchozích verzích produktu IBM MQ, to znamená, že žádný název aplikace neplyne na požadavku na ověření a hodnota MVSxxxx se použije místo toho.

Hodnota 1-8 alfanumerických znaků musí odpovídat pravidlům pro názvy aplikací přístupových tiketů, jak je popsáno v příručkách RACF .

IBM MQ Administrátoři a administrátoři produktu RACF musí oba souhlasit s platnými názvy aplikací, které mají být použity. Administrátor produktu RACF musí vytvořit profil ve třídě PTKTDATA s přístupem READ k 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 má být použit pro ověření pomocí hesla.

Související informace najdete 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 lístek.

Tento parametr je volitelný; je-li zadán, použije se s ID uživatele v kontextu zabezpečení MQMD k sestavení souboru UTOKEN, který je odeslán do produktu IMS za účelem poskytnutí kontextu zabezpečení. Není-li zadán, použije se ID uživatele bez ověření. Závisí to na nastavení přepínačů RACF , které mohou vyžadovat přítomnost ověřovatele.

Tato hodnota je ignorována, pokud je první bajt prázdný nebo má hodnotu null. Mohou být použity následující speciální hodnoty:

IAUNON
Žádné ověření.

Délka tohoto pole je dána LNAUTH. Počáteční hodnota tohoto pole je IAUNON.

IICMT (jednobajtový znakový řetězec)

Režim vázaného zpracování.

Další informace o režimech potvrzování produktu IMS naleznete v příručce *OTMA Reference* . Hodnota musí být jedna z následujících:

ICMCTS
Potvrdit poté odeslání.

Tento režim implikuje dvojité řazení výstupu, ale kratší doba obsazenosti oblasti. Rychlá cesta a konverzační transakce nemohou být spuštěny s tímto režimem.

ICMSTC
Odeslat a potvrdit.

Počáteční hodnota tohoto pole je ICMCTS.

IICSI (10číslicové podepsané celé číslo)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

IIENC (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je 0.

IIFLG (desetimístné podepsané celé číslo)

Příznaky.

Hodnota musí být:

IINON
Žádné vlajky.

Počáteční hodnota tohoto pole je IINONE.

IIFMT (8bajtový znakový řetězec)

Název formátu produktu IBM MQ , který následuje za záhlavím MQIIH.

Tato hodnota určuje název formátu produktu IBM MQ pro data, která následují strukturu MQIIH.

Na základě 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 LNFMT. Počáteční hodnota tohoto pole je FMNONE.

IILEN (10ciferné celé číslo se znaménkem)

Délka struktury MQIIH.

Hodnota musí být:

IILEN1
Délka struktury záhlaví informací produktu IMS .

Počáteční hodnota tohoto pole je IILEN1.

IILTO (8bajtový znakový řetězec)

Přepsání logického terminálu.

To je umístěno v poli IO PCB. Je volitelný; pokud není zadán, je použit název TPIPE. Je ignorován, pokud je první bajt prázdný, nebo má hodnotu null.

Délka tohoto pole je dána LNLTOV. Počáteční hodnota tohoto pole je 8 prázdných znaků.

IIMMN (8bajtový znakový řetězec)

Název mapy služeb formátu zpráv.

To je umístěno v poli IO PCB. Tato položka není povinná. Na vstupu se jedná o MID, na výstupu, který představuje MOD. Je ignorován, 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)

Název formátu IBM MQ zprávy odpovědi.

Jedná se o název formátu IBM MQ zprávy odpovědi, který bude odeslán jako odpověď na aktuální zprávu. Pravidla pro kódování jsou stejná jako pravidla pro pole *MDFMT* v produktu MQMD.

Délka tohoto pole je dána LNFMT. Počáteční hodnota tohoto pole je FMNONE.

IIRSV (jednobajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole; musí být prázdné.

IISEC (jednobajtový znakový řetězec)

Rozsah zabezpečení.

Toto označuje požadované zpracování zabezpečení produktu IMS . Jsou definovány tyto hodnoty:

ISCHK
Zkontrolujte rozsah zabezpečení.

ACEE je postaven v řídicí oblasti, ale ne v závislé oblasti.

ISUJÍCÍ
Plný rozsah zabezpečení.

ACEE uložený v mezipaměti je sestavován v řídicí oblasti a ACEE bez mezipaměti je sestaven v závislé oblasti. Používáte-li ISSFUL, musíte se ujistit, že ID uživatele, pro které je produkt ACEE zabudován, má přístup k prostředkům použitým v závislé oblasti.

Nejsou-li pro toto pole zadány ISSCHK a ISSFUL, předpokládá se ISSCHK.

Počáteční hodnota tohoto pole je ISSCHK.

IISID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

IISIDV
Identifikátor pro strukturu záhlaví informací produktu IMS .

Počáteční hodnota tohoto pole je IISIDV.

IITID (16bajtový bitový řetězec)

Identifikátor instance transakce.

Toto pole je používáno výstupními zprávami z IMS , takže je ignorován na prvním vstupu. Je-li parametr *IITST* nastaven na ITSIC, musí být tento parametr poskytnut na dalším vstupu a všechny následující vstupy, aby bylo možné produkt IMS korelovat se správnou konverzací zpráv. Mohou být použity následující speciální hodnoty:

ITINON
Chybí ID instance transakce.

Délka tohoto pole je dána LNTIID. Počáteční hodnota tohoto pole je ITINON.

IITST (jednobajtový znakový řetězec)

Stav transakce.

Tento stav označuje stav konverzace produktu IMS . Tato hodnota je na prvním vstupu ignorována, protože žádná konverzace neexistuje. Na následných vstupech označuje, zda je konverzace aktivní nebo ne. Na výstupu je nastaven pomocí IMS. Hodnota musí být jedna z následujících:

ITSIC
-V rozhovoru.

ITSNICKÝ
Ne v rozhovoru.

ITSARC
Vrátit data stavu transakce ve formě architektury.

Tato hodnota se používá pouze s příkazem IMS `/DISPLAY TRAN` . Způsobuje, že data stavu transakce budou vrácena ve formě znaku IMS namísto znakového formuláře. Další podrobnosti naleznete v tématu Zápis transakčních programů IMS prostřednictvím produktu IBM MQ .

Počáteční hodnota tohoto pole je ITSNIC.

IIVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

IIVER1
Číslo verze pro strukturu záhlaví informací produktu IMS .

Následující konstanta uvádí číslo verze aktuální verze:

IIVERC
Aktuální verze struktury záhlaví informací produktu IMS .

Počáteční hodnota tohoto pole je IIVER1.

Počáteční hodnoty

Tabulka 706. Počáteční hodnoty polí 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	IINON	0
IILTO	Není	Mezery
IIMMN	Není	Mezery
IIRFM	FMNONE	Mezery
IIAUT	IAUNON	Mezery
IITID	ITINON	Hodnoty null
IITST	ITSNICKÝ	'¬'
IICMT	ICMCTS	'0'
IISEC	ISCHK	'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 (Inquire message property options) na IBM i

Struktura MQIMPO umožňuje aplikacím určit volby, které řídí, jak se mají dotazovat vlastnosti zpráv.

Přehled

Účel: Struktura je vstupním parametrem volání MQINQMP.

Znaková sada a kódování: Data ve struktuře MQIMPO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1092
- "Počáteční hodnoty" na stránce 1098
- "Deklarace RPG" na stránce 1098

Pole

Struktura MQIMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

IPOPT (10číslicové celé číslo se znaménkem)

Následující volby řídí akci MQINQMP. Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace). Kombinace voleb, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné.

Volby hodnot dat: Následující volby se vztahují ke zpracování dat hodnoty, když je vlastnost načtena ze zprávy.

IPCVAL

Tato volba vyžaduje, aby hodnota vlastnosti byla převedena tak, aby odpovídala hodnotám *IPREQCSI* a *IPREQENC* určeným před voláním MQINQMP vrací hodnotu vlastnosti v oblasti *Value* .

- Je-li konverze úspěšná, jsou pole *IPRETCSI* a *IPRETENC* nastavena na stejné hodnoty jako *IPREQCSI* a *IPREQENC* při návratu z volání MQINQMP.
- Pokud převod selže, ale volání MQINQMP se jinak dokončí bez chyby, hodnota vlastnosti se vrátí nekonvertované.

  Je-li vlastnost řetězec, jsou pole *IPRETCSI* a *IPRETENC* nastavena na znakovou sadu a kódování nepřeváděné řetězce.

  Kód dokončení je CCWARN v tomto případě, s kódem příčiny RC2466. Kurzor vlastností se zálohuje na vrácenou vlastnost.

Pokud se hodnota vlastnosti rozbalí během převodu a překročí velikost parametru **Value** , vrátí se nekonvertovaný kód s kódem dokončení CCFAIL; kód příčiny je nastaven na hodnotu RC2469.

Parametr **DataLength** volání MQINQMP vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

Tato volba také vyžaduje, aby:

- Pokud název vlastnosti obsahuje zástupný znak, a
- Pole *IPRETNAMECHRP* je inicializováno s adresou nebo offsetem pro vrácený název,

pak je vrácený název převeden tak, aby odpovídal hodnotám *IPREQCSI* a *IPREQENC* .

- Je-li konverze úspěšná, jsou pole *VSCCSID* souboru *IPRETNAMECHRP* a kódování vráceného názvu nastaveny na vstupní hodnotu *IPREQCSI* a *IPREQENC*.
- Pokud převod selže, ale volání MQINQMP se jinak dokončí bez chyby nebo varování, vrácené jméno se nekonvertuje. Kód dokončení je CCWARN v tomto případě, s kódem příčiny RC2492.

  Kurzor vlastností se zálohuje na vrácenou vlastnost. Hodnota RC2466 je vrácena, pokud nejsou obě hodnoty převedeny a název převedeny.

Pokud se vrácený název rozbalí během převodu a překročí velikost pole *VSBufsize* v poli *RequestedName*, vrácený řetězec zůstane nekonvertovaný, kód dokončení CCFAIL a kód příčiny je nastaven na RC2465.

Pole *VSLength* struktury MQCHARV vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

IPCTYP

Tato volba vyžaduje převedení hodnoty vlastnosti z aktuálního datového typu do datového typu zadaného v parametru **Type** volání MQINQMP.

- Je-li konverze úspěšná, parametr **Type** se nezmění při návratu volání MQINQMP.
- Pokud konverze selže, ale volání MQINQMP se jinak dokončí bez chyby, volání selže s příčinou RC2470. Kurzor vlastnosti se nemění.

Pokud konverze datového typu způsobí, že se hodnota během konverze rozšíří a převedená hodnota překročí velikost parametru **Value** , hodnota se vrátí nekonvertovaný, kód dokončení CCFAIL a kód příčiny je nastaven na RC2469.

Parametr **DataLength** volání MQINQMP vrací délku, kterou by hodnota vlastnosti měla převést na, aby aplikace mohla určit velikost vyrovnávací paměti, která se má použít pro umístění převedené hodnoty vlastnosti. Kurzor vlastnosti se nemění.

Není-li hodnota parametru **Type** volání MQINQMP platná, volání selže s kódem příčiny RC2473.

Není-li požadovaná konverze typu dat podporována, volání selže s příčinou RC2470. Jsou podporovány následující převody datových typů:

Tabulka 707. Podporované konverze datových typů

Datový typ vlastnosti	Podporované cílové datové typy
TYBOL	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 týkající se podporovaných převodů jsou následující:

- Hodnoty číselných vlastností lze převádět z jednoho datového typu do jiného, za předpokladu, že během převodu nebudou ztracena žádná data.

  Např. hodnota vlastnosti s datovým typem TYPI32 může být převedena na hodnotu s datovým typem TYPI64, ale nelze ji převést na hodnotu s typem dat TYPI16.
- Hodnotu vlastnosti libovolného datového typu lze převést na řetězec.
- Hodnotu vlastnosti řetězce lze převést na jakýkoli jiný typ dat za předpokladu, že je řetězec správně formátován pro převod. Pokusí-li se aplikace převést hodnotu vlastnosti řetězce, která není správně naformátována, produkt IBM MQ vrátí kód příčiny RC2472.
- Pokud se aplikace pokusí o převod, který není podporován, produkt IBM MQ vrátí kód příčiny RC2470.

Specifická pravidla pro převod hodnoty vlastnosti z jednoho datového typu do jiného jsou následující:

- Při převodu hodnoty vlastnosti TYPBOL na řetězec je hodnota TRUE převedena na řetězec "TRUE" a hodnota false se převede na řetězec "FALSE".
- Při převodu hodnoty vlastnosti TYPBOL 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 TYPBOL se řetězec "TRUE" nebo "1" převede na TRUE a řetězec "FALSE" nebo "0" se převede na FALSE.

  Všimněte si, že výrazy "TRUE" a "FALSE" nejsou citlivé na velikost písmen.

  Jakýkoli jiný řetězec nelze převést; IBM MQ vrátí 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
  Volitelné znaménko plus (+) nebo znak minus (-).

  digits
  Souvislá posloupnost číselných znaků (0-9). Musí být přítomen alespoň jeden číselný znak.

  Po pořadí znaků číslic může řetězec obsahovat i jiné znaky, které nejsou číslice, ale konverze se zastaví, jakmile je dosaženo začátku těchto znaků. Předpokládá se, že řetězec představuje desítkové celé číslo.

  IBM MQ vrátí 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
  Volitelné znaménko plus (+) nebo znak minus (-).

  digits
  Souvislá posloupnost číselných znaků (0-9). Musí být přítomen alespoň jeden číselný znak.

e_char
Exponent znak, který je buď "E" nebo "e".

e_sign
Volitelný znak plus (+) nebo znaménko minus (-) pro exponent.

e_digits
Souvislá posloupnost znaků číslic (0-9) pro exponent. Pokud řetězec obsahuje exponent exponent, musí být přítomen alespoň jeden znak číslice.

Po pořadí znaků číslic nebo volitelných znaků představujících exponent může řetězec obsahovat jiné znaky, které nejsou číslice, ale konverze se zastaví, jakmile se dosáhne první z těchto znaků. Předpokládá se, že řetězec představuje desetinné číslo s plovoucí řádovou čárkou s exponentem, který je mocninou 10.

IBM MQ vrátí kód příčiny RC2472 , pokud není řetězec správně naformátován.

- Při převodu číselné hodnoty vlastnosti na řetězec se hodnota převede na řetězcovou reprezentaci hodnoty jako dekadické číslo, nikoli řetězec obsahující znak ASCII pro tuto hodnotu. Například, celé číslo 65 je převedeno na řetězec "65", nikoli řetězec "A".
- Při převádění hodnoty vlastnosti řetězce bajtu na řetězec se každý bajt převede na dva hexadecimální znaky, které představují bajt. Příklad: 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 se nevrátí.

Je-li zadána vyrovnávací paměť *ReturnedName* , pole *VSLength* struktury MQCHARV se vyplní s délkou názvu vlastnosti. Název vlastnosti není vrácen.

Volby iterace: Následující volby se vztahují k iteraci přes vlastnosti pomocí názvu se zástupným znakem

IPINQF
Zjišťuje se první vlastnost, která odpovídá uvedenému názvu. Po tomto volání je kurzor založen na vlastnosti, která je vrácena.

Toto je výchozí hodnota.

Volba IPINQC může být následně použita s voláním MQINQMP, je-li to nutné, aby se mohla znovu dotázat na stejnou vlastnost.

Všimněte si, že existuje pouze jeden kurzor vlastnosti; proto, je-li název vlastnosti uvedený ve volání MQINQMP, změny kurzoru se resetují.

Tato volba není platná při jedné z následujících voleb:

IPINQN
IPINQC

IPINQN
Zvodí na další vlastnosti, která odpovídá uvedenému názvu, pokračuje hledání od kurzoru vlastnosti. Kurzor se přesune na vrácenou vlastnost.

Jedná-li se o první volání MQINQMP pro zadaný název, bude vrácena první vlastnost, která odpovídá zadanému názvu.

Volba IPINQC může být následně použita s voláním MQINQMP, je-li to nutné, aby se mohla znovu dotázat na stejnou vlastnost.

Pokud byla vlastnost pod kurzorem odstraněna, funkce MQINQMP vrátí následující odpovídající vlastnost za hodnotou, která byla odstraněna.

Je-li přidána vlastnost, která odpovídá zástupnému znaku, zatímco iterace probíhá, 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, zatímco iterace probíhal, není po jejím odstranění vrácena.

Tato volba není platná při jedné z následujících voleb:

IPINQF
IPINQC

IPINQC
Načtení hodnoty vlastnosti, na kterou ukazuje kurzor, který je uveden ve vlastnosti. Vlastnost, na kterou ukazuje kurzor vlastností, je ta, která byla naposledy dotazovaná, buď pomocí volby IPINQF, nebo IPINQN.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy, když je zadán popisovač zprávy v poli *MsgHandle* MQGMO na volání MQGET nebo pokud je popisovač zprávy zadán v polích *OriginalMsgHandle* nebo *NewMsgHandle* ve struktuře MQPMO na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud ustanoveno, nebo pokud byla vlastnost, na kterou ukazuje kurzor vlastností, odstraněna, volání selže s kódem dokončení CCFAIL a příčinou RC2471.

Tato volba není platná při jedné 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.

Program IPNONE opomáhá dokumentaci programu; není zamýšleno, aby tato volba byla použita s jinou, ale protože její hodnota je nula, takové použití nelze zjistit.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPINQF.

IPREQCSI (10ciferné celé číslo se znaménkem)

Znaková sada, do které se má dotazovaná hodnota vlastnosti převést, je-li hodnota znakový řetězec. Jedná se také o znakovou sadu, do které se má produkt *ReturnedName* převést, když je zadán IPCVAL nebo IPCTYP.

Počáteční hodnota tohoto pole je CSAPL.

IPREQENC (10ciferné celé číslo se znaménkem)

Jedná se o kódování, do kterého se má dotazovaná hodnota vlastnosti konvertovat, když je zadán IPCVAL nebo IPCTYP.

Počáteční hodnota tohoto pole je ENNAT.

IPRE1 (10ciferné celé číslo se znaménkem)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak.

IPRETCSI (10ciferné celé číslo se znaménkem)

Na výstupu se jedná o znakovou sadu hodnoty vrácené v případě, že parametr **Type** volání MQINQMP je TYPSTR.

Je-li volba IPCVAL zadána a konverze byla úspěšná, hodnota pole *ReturnedCCSID* při návratu je stejná jako hodnota předaná v poli.

Počáteční hodnota tohoto pole je nula.

IPRETENC (10ciferné celé číslo se znaménkem)

Na výstupu se jedná o kódování vrácené hodnoty.

Je-li volba IPCVAL zadána a konverze byla úspěšná, hodnota pole *ReturnedEncoding* při návratu je stejná jako hodnota předaná v poli.

Počáteční hodnota tohoto pole je ENNAT.

IPRETNAMCHRP (desetimístné podepsané celé číslo)

Aktuální název dotazované vlastnosti.

Na vstupu lze vyrovnávací paměť typu string předat pomocí pole *VSPtr* nebo *VSOffset* struktury MQCHARV. Délka vstupní vyrovnávací paměti řetězce je určena pomocí pole *VSBufsize* struktury MQCHARV.

Při návratu z volání MQINQMP je vyrovnávací paměť řetězce dokončena s názvem neurčené vlastnosti, za předpokladu, že vyrovnávací paměť řetězce byla dostatečně dlouhá, aby plně obsahovala název. Pole *VSLength* struktury MQCHARV se vyplní s délkou názvu vlastnosti. Pole *VSCCSID* struktury MQCHARV je vyplněno, aby byla uvedena znaková sada vráceného názvu bez ohledu na to, zda došlo k selhání převodu názvu či nikoli.

Jedná se o vstupní/výstupní pole. Počáteční hodnota tohoto pole je MQCHARV_DEFAULT.

IPSID (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor struktury. Hodnota musí být:

IPSIDV
Identifikátor pro strukturu voleb vlastností zprávy dotazu.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPSIDV.

IPTYP (10ciferné celé číslo se znaménkem)

Řetězcová reprezentace datového typu vlastnosti.

Pokud byla vlastnost zadána v záhlaví MQRFH2 a atribut MQRFH2 `dt` není rozpoznán, lze toto pole použít k určení datového typu vlastnosti. *TypeString* je vrácen v kódované znakové sadě 1208 (UTF-8) a je prvních osm bajtů hodnoty atributu `dt` vlastnosti, které se nezdařilo rozpoznat

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je řetězec s hodnotou null v programovacím jazyku C a 8 prázdných znaků v jiných programovacích jazycích.

IPVER (10číslicové podepsané celé číslo)

Jedná se o číslo verze struktury. Hodnota musí být:

IPVER1
Číslo verze pro strukturu voleb vlastností zprávy dotazu.

Následující konstanta uvádí číslo verze aktuální verze:

IPVERC
Aktuální verze struktury voleb vlastností dotazových zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je IPVER1.

Počáteční hodnoty

Tabulka 708. Počáteční hodnoty polí 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		prázdné znaky

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 (Message Descriptor) na serveru IBM i

Přehled

Účel: Struktura MQMD obsahuje řídicí informace, které jsou připojeny k datům aplikace, když se zpráva pohybuje mezi odesílající a přijímající aplikací. Struktura je vstupním/výstupním parametrem na voláních MQGET, MQPUT a MQPUT1 .

Verze: Aktuální verze MQMD je MDVER2. Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQMD, která je podporována prostředím, ale s počáteční hodnotou pole MDVER nastavenou na MDVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí nastavit pole MDVER na číslo verze požadované verze.

Deklarace pro strukturu version-1 je k dispozici s názvem MQMD1.

Znaková sada a kódování: Data ve znakové sadě MQMD musí být ve znakové sadě atributu správce front **CodedCharSetId** a kódování lokálního správce front poskytnutého ENNAT. Je-li 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, budou data v produktu MQMD převedena automaticky. Není nutné, aby aplikace převedl deskriptor MQMD.

- "Použití různých verzí produktu MQMD" na stránce 1099
- "kontext zprávy" na stránce 1099
- "Vypršení zprávy" na stránce 1100
- "Pole" na stránce 1100
- "Počáteční hodnoty" na stránce 1139
- "Deklarace RPG" na stránce 1140

Použití různých verzí produktu MQMD

version-2 MQMD je obecně ekvivalentem k použití MQMD version-1 a k určení dat zprávy se strukturou MQMDE. Pokud však mají všechny pole ve struktuře MQMDE své výchozí hodnoty, může být hodnota MQMDE vynechána. Používá se version-1 MQMD plus MQMDE, jak je popsáno dále v této sekci.

- V případě volání MQPUT a MQPUT1 , pokud aplikace poskytuje version-1 MQMD, může aplikace volitelně připojit data zprávy k datům MQMDE a nastavit pole MDFMT v MQMD na hodnotu FMMDE tak, aby označovalo, že je přítomen MQMDE. Pokud aplikace neposkytuje prostředí MQMDE, předpokládá správce front výchozí hodnoty pro pole v MQMDE.

  Poznámka: Několik polí, která existují ve version-2 MQMD, ale ne version-1 MQMD, jsou vstupní/výstupní pole na volání MQPUT a MQPUT1 . Správce front však nevrátí žádné hodnoty v ekvivalentních polích v MQMDE na výstupu z volání MQPUT a MQPUT1 ; pokud aplikace vyžaduje tyto výstupní hodnoty, musí použít version-2 MQMD.
- Pokud v rámci volání MQGET poskytuje aplikace MQMD version-1 , předpony správce front vrátí zprávu s řetězcem MQMDE, ale pouze v případě, že jedno nebo více polí v prostředí MQMDE má jinou než výchozí hodnotu. Pole MDFMT v MQMD bude mít hodnotu FMMDE, aby označilo, že je přítomen MQMDE.

Výchozí hodnoty, které používá správce front pro pole v MQMDE, jsou stejné jako počáteční hodnoty těchto polí, zobrazené v Tabulka 710 na stránce 1139.

Je-li zpráva v přenosové frontě, některá pole v produktu MQMD jsou nastavena na konkrétní hodnoty; podrobnosti viz "MQXQH (záhlaví přenosové fronty) v systému IBM i" na stránce 1232 .

kontext zprávy

Určitá pole v deskriptoru MQMD obsahují kontext zprávy. Typicky:

- *Kontext identity* souvisí s aplikací, která původně vložila zprávu
- *Výchozí kontext* se vztahuje k aplikaci, která naposledy zadala zprávu
- *Kontext uživatele* se vztahuje k aplikaci, která původně vložila zprávu.

Tyto dvě aplikace mohou být stejné aplikace, ale mohou se také jednat o různé aplikace (například, když je zpráva předána z jedné aplikace do druhé).

Ačkoli kontext identity a výchozí kontext mají obvykle význam popsaný dříve, obsah obou typů kontextových polí ve struktuře MQMD ve skutečnosti závisí na volbách PM*, které jsou zadány při vložení zprávy. V důsledku toho se kontext identity nemusí nutně vztahovat k aplikaci, která původně vložila zprávu, a kontext původu se nemusí nutně vztahovat k aplikaci, která nejnověji vložila zprávu-závisí na návrhu sady aplikací.

Existuje jedna třída aplikace, která nikdy nemění kontext zprávy, konkrétně agent kanálu zpráv (MCA). MCA, kteří přijímají zprávy ze vzdálených správců front, používají kontextový parametr PMSETA na volání MQPUT nebo MQPUT1 . To umožňuje přijímající sběrnici MCA zachovat přesně kontext zprávy, který cestoval se zprávou z odesílající sběrnice MCA. Výsledkem je však, že kontext původu se nevztahuje k aplikaci, která naposledy umístil zprávu (přijímající agent MCA), ale vztahuje se k dřívější aplikaci, která tuto zprávu vložila (pravděpodobně původní aplikace samotná).

Další informace viz téma Kontext zprávy.

Vypršení zprávy

Zprávy, jejichž platnost skončila v zavedené frontě (fronta, která byla otevřena), jsou z fronty automaticky odebrána v přiměřeném časovém intervalu po vypršení jejich platnosti. Některé další nové funkce tohoto vydání produktu IBM MQ mohou vést ke snímaným načítaným frontám, než v předchozí verzi produktu, avšak zprávy s vypršenou platností v načtených frontách budou vždy odebrány během rozumného období jejich vypršení platnosti.

Pole

Struktura MQMD obsahuje následující pole; pole jsou popsána v abecedním pořadí:

MDACC (32bajtový bitový řetězec)

Token evidence.

Tato část je součástí *kontextu identity* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

MDACC umožňuje aplikaci způsobit práci provedenou jako výsledek zprávy, která má být patřičně nabitá. Správce front považuje tyto informace za řetězec bitů a nekontroluje jeho obsah.

Když správce front vygeneruje tyto informace, je nastaven takto:

- První bajt pole je nastaven na délku účetních informací přítomných v bajtech, které následují; tato délka je v rozsahu nula až 30 a je uložena v prvním bajtu jako binární celé číslo.
- Druhý a následující bajt (jak je uvedeno v poli délky) jsou nastaveny na informace o účtování odpovídající prostředí.
  - z/OS Na z/OS jsou informace o účtování nastaveny na:
    - Pro dávkové zpracování produktu z/OS informace o účtování z karty JES JOB nebo z příkazu JES ACCT v kartě EXEC (oddělovač čárky se změní na X'FF '). Tyto informace jsou v případě potřeby zkráceny na 31 bajtů.
    - Pro TSO, číslo účtu uživatele.
    - Pro CICS, identifikátor jednotky práce LU 6.2 (UEUPOWDS) (26 bajtů).
    - Pro IMSje 8znakový název PSB zřetězený s 16znakový IMS obnoveným tokenem obnovy.
  - IBM i V systému IBM ijsou informace o účtování nastaveny na účtovací kód úlohy.

– Linux AIX V systému AIX and Linuxjsou informace o účtování nastaveny na číselný identifikátor uživatele, ve znacích ASCII.

– Windows V systému Windowsjsou informace o účtování nastaveny na Windows NT identifikátor zabezpečení (SID) v komprimovaném formátu. Identifikátor SID jednoznačně identifikuje identifikátor uživatele uložený v poli *MDUID* . Když je SID uloženo v poli *MDACC* , 6bajtová identifikační autorita (umístěná ve třetím a následujících bajtech SID) se vynechá. Například, pokud je Windows NT SID 28 bajtů dlouhý, 22 bajtů informací SID se uloží do pole *MDACC* .

- Poslední bajt je nastaven na typ účtovacího tokenu, jedna z následujících hodnot:

  ATTUCŠTINA
  CICS Identifikátor LUOW.

  ATTDOS
  Předvolený účtovací token PC DOS.

  ATTWANT
  Identifikátor zabezpečení produktu Windows .

  ATT400
  Účtovací token IBM i .

  ATTUNX
  AIX and Linux číselný identifikátor.

  ATUSR
  Uživatelem definovaný evidenční token.

  UPOZORNĚNÍ
  Neznámý typ účtovacího tokenu.

  Typ účtovacího tokenu je nastaven na explicitní hodnotu pouze v následujících prostředích:

  – AIX AIX

  – IBM i IBM i

  – Windows Windows

  a pro IBM MQ MQI clients připojené k těmto systémům.

  V jiných prostředích je typ účtovacího tokenu nastaven na hodnotu ATTUNK. V těchto prostředích lze pole MDPAT použít k odvození typu přijatého tokenu evidence.

- Všechny ostatní bajty jsou nastaveny na binární nulu.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je položka PMSETI nebo PMSETA zadána v parametru **PMO** . Není-li uvedeno PMSETI ani PMSETA, je toto pole na vstupu ignorováno a je to pole pouze pro výstup. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDACC , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDACC , která je uchována se zprávou, pokud je uchována (viz popis PMRET v souboru “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161 pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDACC , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepsání MDACC ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, je pole zcela binární nula.

Toto je výstupní pole pro volání MQGET.

Toto pole není předmětem žádného překladu založeného na znakové sadě správce front-toto pole je považováno za řetězec bitů a nikoli jako řetězec znaků.

Správce front s informacemi v tomto poli nic neudělá. Aplikace musí tyto informace interpretovat, pokud chce použít informace pro účely účetnictví.

Pro pole *MDACC* může být použita následující speciální hodnota:

ANONE
Není zadán žádný token účtování.

Hodnota je binární nula pro délku pole.

Délka tohoto pole je dána LNACCT. Počáteční hodnota tohoto pole je ACNONE.

MDAID (32bajtový znakový řetězec)

Data aplikace související s identitou.

Tato část je součástí *kontextu identity* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

MDAID jsou informace, které jsou definovány sadou aplikací a lze je použít k poskytnutí dalších informací o zprávě nebo jejím původci. Správce front považuje tyto informace za znaková data, ale nedefinuje její formát. Když správce front vygeneruje tyto informace, je zcela prázdný.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je položka PMSETI nebo PMSETA zadána v parametru **PMO** . Je-li přítomen znak null, správce front převede znak null a všechny následující znaky na mezery. Není-li uvedeno PMSETI ani PMSETA, je toto pole na vstupu ignorováno a je to pole pouze pro výstup. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDAID , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDAID , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDAID , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDAID ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána LNAIDD. Počáteční hodnota tohoto pole je 32 prázdných znaků.

MDAOD (4bajtový znakový řetězec)

Údaje o žádosti vztahující se k původu.

Toto je část *kontextu původu* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

MDAOD jsou informace, které jsou definovány sadou aplikací, které lze použít k poskytnutí dalších informací o původu zprávy. Například by mohly být nastaveny aplikacemi, které jsou spuštěny s odpovídajícím oprávněním uživatele, aby označovaly, zda jsou data identity důvěryhodná.

Správce front považuje tyto informace za znaková data, ale nedefinuje její formát. Když správce front vygeneruje tyto informace, je zcela prázdný.

Pro volání MQPUT a MQPUT1 je to vstupní/výstupní pole, je-li položka PMSETA zadána v parametru **PMO** . Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny na mezery. Není-li položka PMSETA uvedena, je toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDAOD , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDAOD , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDAOD , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDAOD ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána LNAORD. Počáteční hodnota tohoto pole je 4 prázdné znaky.

MDBOC (10číslicové celé číslo se znaménkem)

Čítač k vrácení.

Jedná se o počet případů, kdy byla zpráva již dříve vrácena voláním MQGET jako součást pracovní jednotky a následně vrácena. Je poskytnuta jako pomůka pro aplikaci při zjišťování chyb zpracování, které jsou založeny na obsahu zprávy. Počet vylučuje volání MQGET, která specifikovanou některou z voleb GMBRW*.

Přesnost tohoto počtu je ovlivněna atributem fronty **HardenGetBackout** ; viz “Atributy pro fronty” na stránce 1353.

Toto je výstupní pole pro volání MQGET. Pro volání MQPUT a MQPUT1 je ignorována. Počáteční hodnota tohoto pole je 0.

MDCID (24bajtový bitový řetězec)

Identifikátor korelace.

Jedná se o bajtový řetězec, který může aplikace použít ke vztažení jedné zprávy k jiné, nebo ke vztažení zprávy k jiné práci, kterou aplikace provádí. Identifikátor korelace je trvalou vlastností zprávy a uchovává se po restartu správce front. Vzhledem k tomu, že identifikátor korelace je bajtový řetězec a nikoli znakový řetězec, není korelační identifikátor převeden mezi znakovými sadami, když se tok zpráv z jednoho správce front do jiného správce front.

Pro volání MQPUT a MQPUT1 může aplikace určit libovolnou hodnotu. Správce front tuto hodnotu přenáší se zprávou a doručuje ji aplikaci, která vydá požadavek na získání pro zprávu.

Pokud aplikace uvádí PMNCID, správce front vygeneruje jedinečný korelační identifikátor, který se odešle se zprávou a také se vrátí do odesílající aplikace na výstupu z volání MQPUT nebo MQPUT1 .

Tento generovaný korelační identifikátor je uložen se zprávou, pokud je uchován a je použit jako identifikátor korelace, když je zpráva odeslána jako publikace odběratelům, kteří specifikují CINONE v poli SDCID v MQSD předávaném na volání MQSUB.

Další podrobnosti o zachovaných příručkách naleznete v příručce “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161 .

Když správce front nebo agent kanálu zpráv vygeneruje zprávu s hlášením, nastaví pole MDCID tak, jak je určeno polem MDREP původní zprávy, buď ROCMTC nebo ROPCI. Aplikace, které generují zprávy sestav, by to měly provést také.

Pro volání MQGET je MDCID jedním z pěti polí, které lze použít k výběru konkrétní zprávy, která má být načtena z fronty. Podrobné informace o tom, jak určit hodnoty pro toto pole, najdete v popisu pole MDMID .

Zadání CINONE jako korelačního identifikátoru má stejný účinek jako neuvedení MOCORI, to znamená, že jakýkoli korelační identifikátor se bude shodovat.

Je-li v parametru **GMO** na volání MQGET zadána volba GMMUC, je toto pole ignorováno.

Při návratu z volání MQGET je pole MDCID nastaveno na identifikátor korelace vrácené zprávy (je-li k dispozici).

Mohou být použity následující speciální hodnoty:

CINNE
Není uveden žádný korelační identifikátor.

Hodnota je binární nula pro délku pole.

KINETY
Zpráva je začátkem nové relace.

Tato hodnota je rozpoznána produktem CICS bridge jako označení začátku nové relace, tj. začátek nové posloupnosti zpráv.

U volání MQGET se jedná o vstupní/výstupní pole. Pro volání MQPUT a MQPUT1 je toto vstupní pole, není-li zadáno PMNCID, a výstupní pole, je-li uvedeno PMNCID. Délka tohoto pole je dána hodnotou LNCID. Počáteční hodnota tohoto pole je CINONE.

MDCSI (10ciferné celé číslo se znaménkem)

Uvádí identifikátor znakové sady pro znaková data ve zprávě.

Poznámka: Znaková data v MQMD a ostatních datových strukturách IBM MQ , které jsou parametry na voláních, musí být ve znakové sadě správce front. Tento atribut je definován atributem **CodedCharSetId** správce front; podrobnosti o tomto atributu viz "Atributy pro správce front v systému IBM i" na stránce 1384 .

Mohou být použity následující speciální hodnoty:

CSQM
Identifikátor znakové sady správce front.

Znaková data ve zprávě jsou uvedena ve znakové sadě správce front.

Na základě volání MQPUT a MQPUT1 změní správce front tuto hodnotu v deskriptoru MQMD odeslanou se zprávou na skutečný identifikátor znakové sady správce front. V důsledku toho není hodnota CSQM nikdy vrácena voláním MQGET.

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data ve zprávě se nacházejí ve stejné znakové sadě jako v této struktuře. 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 se zprávou na skutečný identifikátor znakové sady MQMD. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole MDPAT v MQMD je ATBRKR.

CSEMBD.
Identifikátor vložené znakové sady.

Znaková data ve zprávě se nacházejí ve znakové sadě s identifikátorem, který je obsažen v samotných datech zprávy. V datech zprávy může být libovolný počet identifikátorů znakových sad, který se vztahuje 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 v rámci volání MQPUT a MQPUT1 . Je-li zadán na volání MQGET, brání převodu zprávy.

Na základě volání MQPUT a MQPUT1 změní správce front hodnoty CSQM a CSINHT v MQMD odeslanou se zprávou, jak bylo popsáno výše, ale nezmění MQMD, které je určeno v rámci volání MQPUT nebo MQPUT1 . Na zadané hodnotě není provedena žádná další kontrola.

Aplikace, které načítají zprávy, by měly porovnat toto pole s hodnotou, kterou aplikace očekává; pokud se hodnoty liší, může aplikace vyžadovat převod znakových dat ve zprávě.

Je-li ve volání MQGET zadána volba GMCONV, je toto pole vstupní/výstupní pole. Hodnota uvedená v aplikaci je identifikátor kódované znakové sady, do kterého by měla být data zprávy v případě potřeby převedena. Je-li konverze úspěšná nebo zbytečná, hodnota se nezmění (kromě toho, že hodnota CSQM nebo CSINHT se převede na skutečnou hodnotu). Pokud je konverze neúspěšná, 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 (10ciferné celé číslo se znaménkem)

Číselné kódování dat zprávy.

Určuje číselné kódování číselných dat ve zprávě. Nevztahuje se na číselná data ve struktuře MQMD jako takové. Numerické kódování definuje znázornění použité pro binární celá čísla, packed-decimální celá čísla a čísla s pohyblivou řádovou čárkou.

Na základě 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
Kódování nativního 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 záhlaví, makra, COPY nebo INCLUDE souborů odpovídajících prostředí, ve kterém bude aplikace spuštěna.

Aplikace, které vložila zprávy, by normálně měly uvádět ENNAT. Aplikace, které načítají zprávy, by měly porovnat toto pole s hodnotou ENNAT; pokud se hodnoty liší, aplikace může potřebovat převést numerická data ve zprávě. Volbu GMCONV lze použít k vyžádání správce front pro převod zprávy v rámci zpracování volání MQGET.

Je-li ve volání MQGET zadána volba GMCONV, je toto pole vstupní/výstupní pole. Hodnota uvedená aplikací je kódování, do kterého by měla být data zprávy převedena, je-li to nutné. Je-li konverze úspěšná nebo zbytečná, hodnota se nezmění. Pokud je konverze neúspěšná, 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 (10ciferné 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 se sníží tak, aby odrážela dobu, kterou zpráva stráví na cílové frontě, a také na všech intermediačních přenosových frontách, pokud je vložení do vzdálené fronty. Může být také snížena podle agentů kanálů zpráv, aby odrážely časy přenosu, jsou-li tyto údaje významné. Podobně může i aplikace přeposílání této zprávy do jiné fronty snížit hodnotu, je-li to nutné, pokud si ji zprávu uchovala po významnou dobu. Avšak čas vypršení platnosti je považován za přibližný a hodnota nemusí být snížena, aby odrážela malé časové intervaly.

Když je zpráva načtena aplikací pomocí volání MQGET, pole MDEXP představuje velikost původní doby vypršení platnosti, která stále zůstává.

Po uplynutí doby vypršení platnosti zprávy bude možné, že správce front bude vyřazen z ukončení. V aktuálních implementacích je zpráva vyřazena v případě, že dojde k volání příkazu MQGET při procházení nebo při procházení, které by vrátilo zprávu, protože již platnost zprávy dosud nevypršela. Například volání MQGET bez procházení s polem GMMO v MQGMO nastaveným na hodnotu MONONE ve čtení z fronty seřazených FIFO způsobí, že všechny zprávy s vypršenou platností budou zahozeny až do první zprávy bez vypršení platnosti. Při použití fronty s prioritou bude stejné volání vyřazeno vypršelé zprávy s vyšší prioritou a zprávami stejné priority, které dorazily do fronty před první zprávou bez vypršení platnosti.

Platnost zprávy, jejíž platnost vypršela, se nikdy nevrací do aplikace (buď pomocí procházení nebo při volání MQGET bez procházení), takže hodnota v poli MDEXP deskriptoru zpráv po úspěšném volání MQGET je buď větší než nula, nebo speciální hodnota EIULIM.

Je-li zpráva vložena do vzdálené fronty, zpráva může vypršet (a být vyřazena), zatímco se nachází ve střední přenosové frontě, než se zpráva dostane do cílové fronty.

Sestava se vygeneruje, když je zahozena zpráva s vypršenou platností, pokud byla zpráva uvedena jako jedna z voleb sestavy ROEXP*. Není-li zadána žádná z těchto voleb, nebude vygenerována žádná

taková sestava. Předpokládá se, že zpráva již není relevantní po uplynutí této doby (možná proto, že ji později nahradila novější zpráva).

Jakýkoliv jiný program, který vyřadí zprávy na základě doby platnosti, musí také odeslat odpovídající zprávu, pokud byla požadována.

Poznámka:

1. Je-li zpráva vložena s hodnotou MDEXP nula, volání MQPUT nebo MQPUT1 selže s kódem příčiny RC2013; v tomto případě se nevygeneruje žádná zpráva sestavy.
2. Vzhledem k tomu, že zpráva s uplynulou dobou platnosti nemusí být skutečně vyřazena, mohou být zprávy ve frontě, které prošly jejich dobou platnosti, a které proto nejsou způsobilé pro načtení. Tyto zprávy se však započítávají do počtu zpráv ve frontě pro všechny účely, včetně spuštění hloubky.
3. Je-li požadována zpráva o vypršení platnosti zprávy, je vygenerována zpráva o vypršení platnosti, nikoli v případě, že je tato zpráva považována za způsobilou pro vyřazení.
4. Vyřazení zprávy s ukončenou platností a generování sestavy vypršení platnosti, je-li požadováno, nejsou nikdy součástí pracovní jednotky aplikace, i když byla zpráva naplánována k vyřazení v důsledku volání MQGET v rámci pracovní jednotky.
5. Je-li zpráva s téměř skončenou platností načtena voláním MQGET v rámci pracovní jednotky a jednotka práce je následně vrácena zpět, může být zpráva považována za způsobilou k vyřazení, než bude možné ji znovu načíst.
6. Je-li zpráva s téměř skončenou platností zamknuta pomocí volání MQGET s GMLK, může být tato zpráva vyřazena dříve, než bude načtena pomocí volání 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, může aplikace provést jednu z následujících akcí, když odešle zprávu odpovědi:

   - Zkopírujte zbývající dobu vypršení platnosti ze zprávy požadavku do zprávy odpovědi.
   - Nastavte čas vypršení platnosti ve zprávě odpovědi na explicitní hodnotu větší než nula.
   - Nastavte dobu vypršení platnosti ve zprávě odpovědi na EIULIM.

   Akce, která se má provést, závisí na návrhu sady aplikací. Avšak výchozí akce pro vložení zpráv do fronty nedoručených zpráv by měla být zachována zbývající doba vypršení platnosti zprávy a její snížení bude pokračovat.
8. Zprávy triggeru jsou vždy generovány s EIULIM.
9. Zpráva (obvykle v přenosové frontě), která má název MDFMT FMXQH, má druhý deskriptor zprávy v rámci MQXQH. Má proto k sobě přidružená dvě pole MDEXP . V tomto případě by měly být zaznamenány následující dodatečné body:

   - Když aplikace vloží zprávu do vzdálené fronty, umístí správce front zprávu na počátku do lokální přenosové fronty a předpony dat aplikační zprávy se strukturou MQXQH. Správce front nastaví hodnoty dvou polí MDEXP tak, aby byly shodné s hodnotami zadanými v aplikaci.

     Pokud aplikace vloží zprávu přímo do lokální přenosové fronty, musí data zprávy již začínat strukturou MQXQH a název formátu musí být FMXQH (ale správce front toto nevynucuje). V tomto případě aplikace nemusí nastavit hodnoty těchto dvou polí MDEXP tak, aby byla stejná. (Správce front nekontroluje, zda pole MDEXP v rámci MQXQH obsahuje platnou hodnotu, nebo dokonce že data zprávy jsou dostatečně dlouhá, aby mohla být zahrnuta.)
   - Když je načtena zpráva s názvem MDFMT FMXQH z fronty (ať už se jedná o normální nebo přenosovou frontu), správce front sníží obě tato pole MDEXP s časem stráveným čekáním na frontu. Pokud data zprávy nejsou dostatečně dlouhá, aby zahrnula pole MDEXP do pole MQXQH, žádná chyba se neobjevuje.
   - Správce front používá pole MDEXP v odděleném deskriptoru zprávy (to znamená, že ne test v deskriptoru zprávy vloženého do struktury MQXQH), aby otestuje, zda je zpráva vhodná pro vyřazení.

- Pokud byly počáteční hodnoty těchto dvou polí MDEXP odlišné, je tedy možné, aby byl čas MDEXP v odděleném deskriptoru zpráv, když je zpráva načítána tak, aby byla větší než nula (takže zpráva není způsobilá pro zrušení), zatímco doba podle pole MDEXP v MQXQH již uplynula. V tomto případě je pole MDEXP v MQXQH nastaveno na nulu.

Je rozpoznána následující speciální hodnota:

EIULIM
Neomezená životnost.

Zpráva má neomezenou dobu platnosti.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je EIULIM.

MDFB (10ciferné celé číslo se znaménkem)

Zpětná vazba nebo kód příčiny.

Tato hodnota se používá se zprávou typu MTRPRT k označení povahy sestavy a je smysluplná pouze s daným typem zprávy. Pole může obsahovat jednu z hodnot FB*, nebo jednu z hodnot RC*. Kódy zpětné vazby jsou seskupeny následujícím způsobem:

FBNONE
Nebyla poskytnuta žádná zpětná vazba.

FBSFST.
Nejnižší hodnota pro zpětnou vazbu generovanou systémem.

FBSLST
Nejvyšší hodnota zpětné vazby generované systémem.

Rozsah systémových zpětnovazebních kódů FBSFFST pomocí FBSLST zahrnuje obecné kódy zpětné vazby uvedené dále v této sekci (FB*) a také kódy příčiny (RC*), které se mohou vyskytnout, když nelze zprávu umístit do cílové fronty.

FBAFST
Nejnižší hodnota pro zpětnou vazbu generovaná aplikací.

BLÁH
Nejvyšší hodnota zpětné vazby generované aplikací.

Aplikace, které generují zprávy sestav, by neměly používat kódy zpětné vazby v rozsahu systému (jiné než FBQUIT), pokud chtějí simulovat zprávy sestavy generované správcem front nebo agentem oznamovacího kanálu.

Na základě volání MQPUT nebo MQPUT1 musí být zadaná hodnota buď FBNONE, nebo musí být v rozsahu systému nebo rozsahu aplikace. Tato hodnota je zkontrolována bez ohledu na hodnotu parametru MDMT.

Obecné kódy zpětné vazby:

FBCOA
Potvrzení přijetí do cílové fronty (viz ROCOA).

TRESKA OBECNÁ
Potvrzení o doručení přijímajícímu podání (viz ROCOD).

FBEXP
Platnost zprávy vypršela.

Zpráva byla zahozena, protože nebyla odebrána z cílové fronty před uplynutím jeho doby vypršení platnosti.

SKOŘEPINA
Pozitivní opatření na akci (viz ROPAN).

FBNANCITY
Negativní upozornění na akci (viz RONAN).

FBQUIT
Aplikace by měla skončit.

To může použít program plánování pracovní zátěže k řízení počtu instancí aplikačního programu, které jsou spuštěny. Při odeslání zprávy MTRPRT s tímto kódem zpětné vazby na instanci aplikačního programu bude tato instance indikovat, že by měla zastavit zpracování. Dodržování této konvence je však záležitostí pro aplikaci; správce front jej nevynucuje.

IMS-bridge feedback codes: Když most IMS obdrží nenulový kód chyby IMS-OTMA, most IMS převede chybový kód z hexadecimálního formátu na desítková, přidá hodnotu FBIERR (300) a umístí výsledek do pole MDFB zprávy odpovědi. Výsledkem je kód zpětné vazby, který má hodnotu v rozsahu FBIFST (301) prostřednictvím FBILST (399), když se vyskytla chyba IMS-OTMA.

Most IMS může generovat následující kódy zpětné vazby:

FBDLZ
Nulová délka dat.

Délka segmentu byla nula v datech aplikace zprávy.

FBDLN
Záporná délka dat.

Délka segmentu byla záporná v datech aplikace zprávy.

FBDLTB
Délka dat je příliš velká.

Délka segmentu byla příliš velká v datech aplikace zprávy.

FBBUFO
Přetečení vyrovnávací paměti.

Hodnota jednoho z polí s délkou by způsobila přetečení vyrovnávací paměti zpráv.

FBLOB1
Chybná délka v jedné chybě.

Hodnota jednoho z polí délky byla jeden bajt příliš krátký.

FBIIH.
Struktura MQIIH není platná nebo chybí.

Pole MDFMT v deskriptoru MQMD uvádí FMIMS, ale zpráva nezačíná platnou strukturou MQIIH.

FBNAFI.
ID uživatele není autorizováno pro použití v produktu IMS.

ID uživatele obsažené v deskriptoru zpráv MQMD nebo heslo obsažené v poli IIAUT ve struktuře MQIIH selhalo při ověřování, které provedl most IMS . V důsledku toho nebyla zpráva předána produktu IMS.

FBIERR
IMSvrátila neočekávanou chybu.

IMSvrátila neočekávanou chybu. Další informace o chybě naleznete v protokolu chyb produktu IBM MQ v systému, na kterém je umístěn most systému IMS .

FBIFST
Nejnižší hodnota pro zpětnou vazbu generovaná produktem IMS.

IMS-generované kódy zpětné vazby obsazují rozsah FBIFST (300) přes FBILST (399). Samotný chybový kód IMS-OTMA je MDFB minus FBIERR.

FBISTOVÁ
Nejvyšší hodnota zpětné vazby generované produktem IMS.

CICS-bridge feedback codes: Rozhraní CICS bridgemůže generovat následující kódy zpětné vazby:

FBCAAB
Aplikace byla ukončena.

Aplikační program uvedený ve zprávě byl ukončen. Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

PUSTY
Aplikaci nelze spustit.

EXEC CICS LINK pro aplikační program uvedený ve zprávě selhal. Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

FBCBRF
CICS bridge byl nestandardně ukončen bez dokončení normálního zpracování chyb.

FBCCSE.
Identifikátor znakové sady není platný.

FBCIHE.
Struktura záhlaví informačního obsahu produktu CICS chybí nebo není platná.

FBCCAECH
Délka CICS commarea není platná.

FBCCIE
Identifikátor korelace není platný.

FBCDLQ.
Fronta nedoručených zpráv není k dispozici.

Úloha CICS bridge nebyla schopna zkopírovat odpověď na tento požadavek do fronty nedoručených zpráv. Požadavek byl zálohován.

FBCENA
Kódování není platné.

PRASATA
V produktu CICS bridge došlo k neočekávané chybě.

Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

FBCNTA
Identifikátor uživatele není autorizován nebo heslo není platné.

Tento kód zpětné vazby se vyskytuje pouze v poli DLREA struktury MQDLH.

FUBCUBŠTINA
Jednotka z práce byla odvolána.

Pracovní jednotka byla zálohována, z jednoho z následujících důvodů:

- Bylo zjištěno selhání během zpracování jiného požadavku v rámci stejné jednotky práce.
- Došlo k nestandardkonci CICS , zatímco jednotka práce právě probíhá.

FUBCUWE.
Pole řízení počtu pracovních jednotek CIUOW není platné.

MQ kódy příčin: Pro zprávy o výjimce obsahuje produkt MDFB kód příčiny MQ . Mezi možné kódy příčiny patří:

RC2051
(2051, X'803 ') Volání s blokováno pro frontu.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2035
(2035, X'7F3') Chybí autorizace pro přístup.

RC2056
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

RC2048
(2048, X'800 ') Fronta nepodporuje trvalé zprávy.

RC2031
(2031, X'7EF') Délka zprávy je větší než maximum pro správce front.

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.

Jedná se o jméno, které může odesilatel zprávy použít k označení povahy dat ve zprávě příjemci. Jakékoli znaky, které jsou ve znakové sadě správce front, lze pro daný název zadat, ale doporučuje se, aby název byl omezen na následující:

- Velká písmena A až Z
- Číselné číslice 0 až 9

Jsou-li použity jiné znaky, nemusí být možné přeložit název mezi znakové sady odesílajícího a přijímajícího správce front.

Název by měl být doplněn mezerami do délky pole nebo znak null použitý k ukončení názvu před koncem pole; hodnota null a všechny následné znaky jsou považovány za mezery. Neuvádějte jméno s úvodními nebo vloženými mezerami. Pro volání MQGET vrátí správce front název doplněný mezerami do délky pole.

Správce front nekontroluje, zda je daný název v souladu s dříve popsanými doporučeními.

Názvy začínající řetězcem "MQ" v horním, dolním a smíšeném případě mají význam, který definuje správce front; neměli byste používat názvy začínající těmito písmeny pro vlastní formáty. Vestavěné formáty správce front jsou:

FMNONE
Chybí název formátu.

Povaha dat není definována. To znamená, že data nelze konvertovat, když je zpráva načtena z fronty pomocí volby GMCONV.

Je-li parametr GMCONV zadán ve volání MQGET a znaková sada nebo kódování dat ve zprávě se liší od hodnoty zadané argumentem **MSGDSC** , vrátí se zpráva s následujícím kódem dokončení a s kódem příčiny (za předpokladu, že nejsou uvedeny žádné další chyby):

- Kód dokončení CCWARN a kód příčiny RC2110 , jsou-li 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 struktur záhlaví MQ). Struktury záhlaví MQ se převedou na požadovanou znakovou sadu a kódování v tomto případě.

FMADMN
Zpráva s požadavkem/odpovědí příkazového serveru.

Jedná se o požadavek na příkaz-server nebo zprávu odpovědi ve formátu PCF (Programmable command Format). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV. Další informace o používání programovatelných zpráv ve formátu příkazu najdete v tématu Použití programu Programmable Command Formats.

FMCICS
CICS .

Data zprávy začínají záhlavím informací produktu CICS MQCIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem CIFMT ve struktuře MQCIH.

FMCMD1
Zpráva odpovědi příkazu typu 1.

Zpráva je zprávou příkazu MQSC příkazu-server, obsahující počet objektů, kód dokončení a kód příčiny. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMCMD2
Zadejte zprávu s odpovědí příkazu typu 2.

Zpráva je zpráva příkazu MQSC, která obsahuje informace o požadovaném objektu (objektech). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMDLH
Hlavička nedoručitelného dopisu.

Data zprávy začínají záhlaví nedoručených zpráv MQDLH. Data z původní zprávy bezprostředně následují za strukturou MQDLH. Název formátu původních dat zprávy je dán polem DLFMT ve struktuře MQDLH. Podrobnosti o této struktuře viz "MQDLH (záhlaví nedoručených zpráv) v systému IBM i" na stránce 1054 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

Sestavy COA a COD se nevygenerují pro zprávy, které mají MDFMT FMDLH.

FMDH/
Hlavička rozdělovníku.

Data zprávy začínají záhlavím MQDH záhlaví distribučního seznamu, což zahrnuje pole záznamů MQOR a MQPMR. Za záhlavím rozdělovníku může následovat další data. Formát dalších dat (pokud existuje) je dán polem DHFMT ve struktuře MQDH. Podrobnosti o této struktuře viz "MQDH (záhlaví distribuce) v systému IBM i" na stránce 1050 . Zprávy s formátem FMDH lze převést, je-li na volání MQGET zadána volba GMCONV.

FMEVNT
Zpráva o události.

Zpráva je zpráva události MQ , která hlásí událost, která se vyskytla. Zprávy událostí mají stejnou strukturu jako programovatelné příkazy; další informace o této struktuře najdete v tématu Struktury pro příkazy a odpovědi. Informace o událostech naleznete v tématu Monitorování událostí.

Zprávy událostí Version-1 lze převést, je-li volba GMCONV zadána na volání MQGET.

FMIMS
IMS .

Data zprávy začínají záhlavím informací produktu IMS MQIIH, za nímž následují data aplikace. Název formátu dat aplikace je uveden v poli *IIFMT* ve struktuře MQIIH. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMIMVS
Řetězec proměnné IMS .

Zpráva je řetězec proměnné IMS , který je řetězcem ve tvaru `llzzccc`, kde:

ll
je 2bajtová délka pole uvádějící celkovou délku položky řetězce proměnné IMS . Tato délka se rovná délce `ll` (2 bajty) a délce `zz` (2 bajtů) a délky samotného znakového řetězce. `ll` je 2bajtové binární celé číslo v kódování zadaném v poli MDENC .

zz
je 2bajtové pole obsahující příznaky, které jsou významné pro IMS. `zz` je bajtový řetězec skládající se ze dvou 1bajtových bitových řetězcových polí a přenáší se bez změny od odesílatele k příjemci (to znamená, že `zz` není předmětem žádné konverze).

ccc
je řetězec znaků s proměnnou délkou obsahující `ll-4` znaků. `ccc` je ve znakové sadě zadané v poli MDCSI .

Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMMUDE
Rozšíření deskriptoru zpráv.

Data zprávy začínají na rozšíření deskriptoru zpráv MQMDE a volitelně jsou následována jinými daty (obvykle data zprávy aplikace). Název formátu, znaková sada a kódování dat, která následuje MQMDE, je dána poli MEFMT, MECSIa MEENC v MQMDE. Podrobnosti o této struktuře viz “MQMDE (rozšíření deskriptoru zpráv) na IBM i” na stránce 1141 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMPCF
Uživatelsky definovaná zpráva v programovatelným formátu příkazu (PCF).

Zpráva je uživatelem definovaná zpráva, která odpovídá struktuře zprávy PCF (Programmable command format). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV. Další informace o používání uživatelem programovatelných zpráv ve formátu příkazu najdete v tématu Použití programů Programmable Command Formats .

FMRMHCACH
Referenční záhlaví zprávy.

Data zprávy začínají odkazem na záhlaví MQRMH a volitelně jsou následována jinými daty. Název formátu, znaková sada a kódování dat jsou dány poli RMFMT, RMCSIa RMENC v MQRMH. Podrobnosti o této struktuře viz “MQRMH (Referenční záhlaví zprávy) v systému IBM i” na stránce 1187 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMRFH
Pravidla a formátovací záhlaví.

Data zprávy začínají na pravidla a formátovací záhlaví MQRFH a volitelně jsou následována jinými daty. Název formátu, znaková sada a kódování dat (pokud existuje) je dána poli RFFMT, RFCSIa RFENC v aplikaci MQRFH. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMRFH2
Pravidla a formátovací záhlaví verze 2.

Data zprávy začínají s pravidly version-2 a formátováním záhlaví MQRFH2a volitelně jsou následována jinými daty. Název formátu, znaková sada a kódování volitelných dat (pokud existuje) je dána poli RF2FMT, RF2CSIa RF2ENC ve struktuře MQRFH2. Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMSTR
Zpráva sestávající pouze ze znaků.

Data zprávy aplikace mohou být buď řetězec SBCS (jednobajtová znaková sada), nebo řetězec DBCS (dvojbajtová znaková sada). Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMTM
Zpráva spouštěče.

Zpráva je zpráva spouštěče, která je popsána strukturou MQTM. Podrobnosti o této struktuře viz “MQTM-Zpráva spouštěče” na stránce 1222 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba GMCONV.

FMWIUFN
Záhlaví informací o práci.

Data zprávy začínají záhlavím MQWIH s informacemi o práci, za nímž následují data aplikace. Název formátu dat aplikace je dán polem WIFMT ve struktuře MQWIH.

FMXQH
Záhlaví přenosové fronty.

Data zprávy začínají s hlavičkou přenosové fronty MQXQH. Data z původní zprávy bezprostředně následují za strukturou MQXQH. Název formátu původních dat zprávy je dán polem MDFMT v rámci struktury MQMD, která je součástí záhlaví MQXQH přenosové fronty. Podrobnosti o této struktuře viz “MQXQH (záhlaví přenosové fronty) v systému IBM i” na stránce 1232 .

Sestavy COA a COD se nevygenerují 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, do níž náleží fyzická zpráva. MDGID se také používá, pokud je pro zprávu povoleno segmentace. Ve všech těchto případech má MDGID hodnotu jinou než null a v poli MDMFL je nastaven jeden nebo více následujících parametrů:

- MFMIG
- MFLMIG
- MFSEG
- MFLSEG
- MFSEGA

Není-li nastaven žádný z těchto parametrů, má MDGID speciální hodnotu null GINONE.

Toto pole nemusí být nastaveno aplikací v rámci volání MQPUT nebo MQGET, pokud:

- Na volání MQPUT je uveden PMLOGO.
- Na volání MQGET není příkaz MOGRPI zadán.

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 produkt MDGID nastaven na příslušnou hodnotu.

Skupiny zpráv a segmenty mohou být zpracovány správně pouze tehdy, je-li 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 zadáno 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 této procedury.
- Není-li PMLOGO uvedeno, aplikace by měla požadovat, aby správce front generoval identifikátor skupiny, nastavením MDGID na GINONE na první volání MQPUT nebo MQPUT1 pro zprávu ve skupině nebo segmentu logické zprávy. Identifikátor skupiny vrácený správcem front na výstupu z tohoto volání by pak měl být použit pro zbývající zprávy ve skupině nebo segmentech logické zprávy. Pokud skupina zpráv obsahuje segmentované zprávy, musí být použit stejný identifikátor skupiny pro všechny segmenty a zprávy ve skupině.

  Není-li PMLOGO uveden, zprávy ve skupinách a segmentech logických zpráv lze vložit do libovolného pořadí (například v opačném pořadí), ale identifikátor skupiny musí být alokován prvním voláním MQPUT nebo MQPUT1 , které bylo vydáno pro některou z těchto zpráv.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v poli PMOPT. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána se zprávou, pokud je otevřený objekt jedinou frontou a nikoli distribučními seznamy, ale ponechá ji nezměněnou, pokud je objekt otevřený distribučnímu seznamu. V případě, že aplikace potřebuje znát generované identifikátory skupin, musí v případě potřeby poskytnout záznamy MQPMR obsahující pole PRGID .

Na vstupu do volání MQGET používá správce front hodnotu popsanou v tabulce Tabulka 1. Na výstupu z volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Je definována následující speciální hodnota:

GINON
Není uveden žádný identifikátor skupiny.

Hodnota je binární nula pro délku pole. Toto je hodnota, která se používá pro zprávy, které nejsou ve skupinách, ne segmenty logických zpráv a pro které segmentaci není povoleno.

Délka tohoto pole je dána LNGID. Počáteční hodnota tohoto pole je GINONE. Toto pole je ignorováno, pokud MDVER je menší než MDVER2.

MDMFL (10ciferné číslicové celé číslo)

Příznaky zpráv.

Jedná se o příznaky, které určují atributy zprávy, nebo řídí jejich zpracování. Příznaky jsou rozděleny do následujících kategorií:

- Příznak segmentace
- Příznaky stavu

Ty jsou popsány v řadě.

Příznaky segmentace: Je-li zpráva příliš velká pro frontu, pokus o vložení zprávy do fronty se obvykle nezdaří. Segmentace je technika, pomocí níž správce front nebo aplikace rozdělí zprávu na menší části, které se nazývají segmenty, a umístí každý segment do fronty jako samostatnou fyzickou zprávu. Aplikace, která načítá zprávu, může buď načíst segmenty jednu po druhé, nebo požádat správce front, aby znovu složl segmenty do jediné zprávy vrácené voláním MQGET. Toho je dosaženo uvedením volby GMCMPM na volání MQGET a dodání vyrovnávací paměti, která je dostatečně velká, aby pojmula úplnou zprávu. (Podrobnosti o volbě GMCMPM viz “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066 .) Segmentace zprávy se může vyskytnout u odesílajícího správce front v intermediačních správci front nebo v cílovém správci front.

Chcete-li řídit segmentaci zprávy, můžete určit jednu z následujících možností:

MFSEGŠTINA
Segmentace je blokována.

Tato volba zabraňuje tomu, aby byla zpráva rozdělena do segmentů správcem front. Je-li pro zprávu, která je již segmentem, zadána, zabrání tomu, aby segment byl rozdělen do menších segmentů.

Hodnota tohoto parametru je binární nula. Toto nastavení je výchozí.

MFSEGA
Segmentace je povolena.

Tato volba umožňuje rozdělení zprávy do segmentů prostřednictvím správce front. Je-li pro zprávu, která je již segmentem, zadána, tato volba umožňuje rozdělení segmentu do menších segmentů. MFSEGA lze nastavit bez nastaveného objektu MFSEG nebo MFLSEG.

Když správce front segmentuje zprávu, aktivuje správce front příznak MFSEG v kopii MQMD, který je odeslán s každým segmentem, ale nemění nastavení těchto parametrů v deskriptoru MQMD, který je poskytován aplikací v rámci volání MQPUT nebo MQPUT1 . Pro poslední segment v logické zprávě správce front zapíná také příznak MFLSEG v deskriptoru MQMD, který je odeslán se segmentem.

Poznámka: Je třeba dbát na to, aby byly zprávy vloženy do MFSEGA, ale bez PMLOGO. Je-li zpráva:

- ne segment a
- Není ve skupině a
- Nepředává se,

Aplikace musí pamatovat na nastavení pole MDGID na hodnotu GINONE před každým voláním MQPUT nebo MQPUT1 za účelem vytvoření jedinečného identifikátoru skupiny, který má být vygenerován správcem front pro každou zprávu. Není-li tomu tak, nesouvisející zprávy by mohly nechtěně skončit se stejným identifikátorem skupiny, což může vést k následnému chybnému zpracování. Chcete-li získat více informací o tom, kdy musí být pole MDGID resetováno, prohlédněte si popis pole MDGID a volby PMLOGO.

Správce front rozdělí zprávy do segmentů podle potřeby, aby se zajistilo, že segmenty (plus jakákoli data záhlaví, která mohou být požadována) se vejdou do fronty. Pro velikost segmentu generovaného správcem front však existuje nižší mezní hodnota a pouze poslední segment vytvořený ze zprávy může být menší než tento limit. (Spodní limit velikosti segmentu generovaný aplikací je jeden bajt.) Segmenty generované správcem front mohou mít nestejnou délku. Správce front zpracovává zprávu následujícím způsobem:

- Uživatelsky definované formáty jsou rozděleny na hranicích, které jsou násobky 16 bajtů. To znamená, že správce front negeneruje segmenty, které jsou menší než 16 bajtů (jiné než poslední segment).
- Vestavěné jiné formáty, než je FMSTR, jsou rozděleny v bodech odpovídajících povaze přítomná data. Správce front však nikdy nerozdělí zprávu uprostřed struktury záhlaví produktu MQ . To znamená, že segment obsahující jednu strukturu záhlaví MQ nemůže být dále rozdělen správcem front, a výsledkem je minimální možná velikost segmentu pro tuto zprávu větší než 16 bajtů.

  Druhý nebo pozdější segment generovaný správcem front bude začínat jedním z následujících:

  – Struktura záhlaví MQ
  – Začátek dat zprávy aplikace
  – Částečná cesta prostřednictvím dat zprávy aplikace
- FMSTR je rozdělen bez ohledu na charakter přítomného data (SBCS, DBCS, nebo smíšených SBCS/DBCS). Je-li řetězec DBCS nebo smíšený SBCS/DBCS, může dojít k převedení segmentů, které nelze převést z jedné znakové sady na jinou. Správce front nikdy nerozděluje zprávy FMSTR do segmentů menších než 16 bajtů (jiných než posledního segmentu).
- Pole MDFMT, MDCSIa MDENC v deskriptoru MQMD každého segmentu jsou nastaveny správcem front k tomu, aby správně popisovala data přítomná na začátku segmentu; název formátu bude buď název vestavěného formátu, nebo název uživatelsky definovaného formátu.
- Pole MDREP v deskriptoru MQMD se segmenty 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 žádný z prvních 100 bajtů uživatelských dat (tj. data následující po všech strukturách záhlaví MQ , které mohou být přítomny), bude volba sestavy změněna na RO*.

Správce front postupuje podle předchozích pravidel, ale jinak rozděluje zprávy nepředvídatelně; nevytvářet hypotézy o tom, kam je zpráva rozdělena.

Pro trvalé zprávy může správce front provádět segmentaci pouze v rámci pracovní jednotky:

- Je-li volání MQPUT nebo MQPUT1 funkční v rámci uživatelské jednotky práce, použije se jednotka práce. Pokud se volání nezdaří prostřednictvím procesu segmentace, odebere správce front všechny segmenty, které byly umístěny do fronty, jako výsledek selhání volání. Selhání však nezabrání úspěšnému potvrzení jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce a neexistuje žádná uživatelsky definovaná jednotka práce, správce front vytvoří pracovní jednotku pouze po dobu trvání hovoru. Je-li volání úspěšné, správce front automaticky potvrdí jednotku práce (aplikace ji nemusí provést). Pokud se volání nezdaří, správce front provede zálohu jednotky práce.
- Pokud je volání mimo uživatelem definovanou jednotku práce, ale uživatelem definovaná jednotka práce existuje, správce front nemůže provést segmentaci. Pokud zpráva nevyžaduje segmentaci, může být volání přesto úspěšné. Pokud však zpráva vyžaduje segmentaci, volání selže s kódem příčiny RC2255.

Pro přechodné zprávy správce front nevyžaduje, aby byla k dispozici jednotka práce, aby bylo možné provést segmentaci.

Zvláštní pozornost musí být věnována převodu zpráv, které mohou být rozděleny na segmenty:

- Je-li převod dat prováděn pouze přijímající aplikací na volání MQGET a aplikace uvádí volbu GMCMPM, předání dat výstupního bodu předání bude předáno úplnou zprávu pro ukončení převodu a skutečnost, že zpráva byla segmentována, nebude pro ukončení zřejmé.
- Pokud přijímající aplikace načte v daném okamžiku jeden segment, bude k převodu jednoho segmentu v daném okamžiku vyvolána uživatelská procedura konverze dat. Výjezd musí být proto schopen převést data v segmentu nezávisle na datech v jiných segmentech.

  Je-li povaha dat ve zprávě taková, že libovolná segmentace dat na šestnáctibajtových okrajích může vyústit v segmenty, které nelze převést uživatelskou procedurou, nebo formát je FMSTR a znaková sada je DBCS nebo smíšená SBCS/DBCS, odesílající aplikace by měla sama vytvořit a vložit segmenty a uvést MFSEGI k potlačení další segmentace. Tímto způsobem odesílající aplikace může zajistit, aby každý segment obsahoval dostatečné informace pro umožnění úspěšného převodu segmentu na výstupu konverze dat.
- Je-li pro odesílajícího agenta MCA (Message Channel Agent) určena konverze odesílatele, program MCA převádí pouze zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí o převod zpráv, které jsou segmenty.

Tento příznak je vstupní příznak volání MQPUT a MQPUT1 a výstupní příznak pro volání MQGET. Při druhém volání správce front také zobrazí hodnotu příznaku pro pole GMSEG v produktu MQGMO.

Počáteční hodnota tohoto parametru je MFSEGI.

Příznaky stavu: Jedná se o příznaky, které označují, zda fyzická zpráva patří do skupiny zpráv, je segment logické zprávy, obojí, nebo ani jedno. Na volání MQPUT nebo MQPUT1 může být určena jedna nebo více z následujících možností, nebo je vrácena pomocí volání MQGET:

MFMIG
Zpráva je členem skupiny.

MFLMIG
Zpráva je poslední logickou zprávou ve skupině.

Je-li tento příznak nastaven, správce front zapne MFMIG v kopii MQMD, která se odešle se zprávou, ale nezmění nastavení těchto parametrů v deskriptoru MQMD, které poskytuje aplikace v rámci volání MQPUT nebo MQPUT1 .

Je platný pro skupinu, která se má skládat pouze z jedné logické zprávy. Pokud se jedná o tento případ, MFLMIG je nastaven, ale pole MDSEQ má hodnotu jedna.

MFSEG
Zpráva je segmentem logické zprávy.

Je-li MFSEG zadáno bez MFLSEG, musí být délka dat zprávy aplikace v segmentu (kromě délek všech struktur záhlaví MQ , které se mohou vyskytovat), alespoň jedna. Je-li délka nulová, volání MQPUT nebo MQPUT1 selže s kódem příčiny RC2253.

MFLSEG
Zpráva je posledním segmentem logické zprávy.

Je-li tento příznak nastaven, správce front zapne MFSEG v kopii MQMD, která se odešle se zprávou, ale nezmění nastavení těchto parametrů v deskriptoru MQMD, které poskytuje aplikace v rámci volání MQPUT nebo MQPUT1 .

Je platné pro logickou zprávu, která se skládá pouze z jednoho segmentu. Je-li tomu tak, je hodnota MFLSEG nastavena, ale pole MDOFF má hodnotu nula.

Je-li zadáno MFLSEG, je přípustné pro délku dat zprávy aplikace v segmentu (kromě délek všech struktur záhlaví, které mohou být přítomny), které mají být nulové.

Aplikace musí zajistit správné nastavení těchto parametrů při vkládání zpráv. Je-li PMLOGO uveden nebo byl zadán v předchozím volání MQPUT pro manipulátor fronty, musí být nastavení příznaků

konzistentní s informacemi o skupině a segmentu uchované správcem front pro obsluhu fronty. Následující podmínky se vztahují na opakované volání MQPUT pro popisovač fronty, je-li zadáno PMLOGO:

- Pokud neexistuje žádná aktuální skupina nebo logická zpráva, všechny tyto příznaky (a jejich kombinace) jsou platné.
- Byl-li zadán MFMIG, musí zůstat zapnutý, dokud nebude zadáno MFLMIG. Volání selže s kódem příčiny RC2241 , pokud tato podmínka není splněna.
- Jakmile je uvedeno MFSEG, musí zůstat zapnuto, dokud nebude uvedeno MFLSEG. Volání selže s kódem příčiny RC2242 , pokud tato podmínka není splněna.
- Jakmile je MFSEG uvedeno bez MFMIG, MFMIG musí zůstat až po uvedení MFLSEG do doby, než bude zadáno MFLSEG. Volání selže s kódem příčiny RC2242 , pokud tato podmínka není splněna.

Tabulka 1 zobrazuje platné kombinace příznaků a hodnoty použité pro různá pole.

Tyto příznaky jsou vstupní příznaky na volání MQPUT a MQPUT1 a výstupní příznaky na volání MQGET. Při druhém volání správce front také odráží hodnoty parametrů pro pole GMGST a GMSST v MQGMO.

Výchozí příznaky: Uvedou se následující informace, které označují, že zpráva má výchozí atributy:

MFNONE
Žádné příznaky zpráv (výchozí atributy zpráv).

To inhibuje segmentaci a označuje, že zpráva není ve skupině a není segmentem logické zprávy. MFNONE je definována pro dokumentaci programu podpory. Není určeno, aby tento parametr byl použit spolu s jiným, ale jako jeho hodnota je nula, takové použití nelze detekovat.

Pole MDMFL je rozděleno na dílčí pole, kde jsou podrobnosti viz “Volby sestav a příznaky zpráv v systému IBM i” na stránce 1418.

Počáteční hodnota tohoto pole je MFNONE. Toto pole je ignorováno, pokud MDVER je menší než MDVER2.

MDMID (24bajtový bitový řetězec)

Identifikátor zprávy.

Jedná se o bajtový řetězec, který se používá k rozlišení jedné zprávy od druhé. Obecně platí, že žádné dvě zprávy by neměly mít stejný identifikátor zprávy, ačkoli správce front tento stav nezakázal. Identifikátor zprávy je trvalou vlastností zprávy a uchovává se přes restarty správce front. Protože identifikátor zprávy je bajtový řetězec a ne znakový řetězec, identifikátor zprávy se nekonvertuje mezi znakovými sadami, když se tok zpráv z jednoho správce front do jiného správce front.

Pro volání MQPUT a MQPUT1 , je-li aplikace MINONE nebo PMNMID určena aplikací, správce front vygeneruje jedinečný identifikátor zprávy, když je zpráva vložena, a umístí ji do deskriptoru zprávy odeslaného se zprávou. Správce front také vrátí tento identifikátor zprávy v deskriptoru zpráv, který patří do odesílající aplikace. Aplikace může tuto hodnotu použít k zaznamenání informací o konkrétních zprávách a k odpovědi 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) a za nímž následuje implementace jedinečného řetězce specifické pro produkt producl-specific. V IBM MQ toto obsahuje prvních 12 znaků názvu správce front a hodnoty odvozené ze systémových hodin. Všichni správci front, kteří mohou vzájemně komunikovat, musí mít proto názvy, které se liší od prvních 12 znaků, aby se zajistilo, že identifikátory zpráv jsou jedinečné. Schopnost generovat jedinečný řetězec také závisí na tom, že systémové hodiny se neměnily zpět. Aby se vyloučila možnost identifikátoru zprávy generovaného správcem front, který duplikuje jeden generovaný aplikací, aplikace by se měla vyvarovat generování identifikátorů s počátečními znaky v rozsahu A až I v ASCII nebo EBCDIC (X'41 'až X'49' a X'C1'až X'C9'). Aplikace však není bráněno v generování identifikátorů s počátečními znaky v těchto rozsazích.

Je-li zpráva vložena do tématu, správce front generuje jedinečné identifikátory zpráv, které jsou nezbytné pro každou publikovanou zprávu. Pokud aplikace zadá PMNMID, správce front vygeneruje jedinečný identifikátor zprávy, který se vrátí na výstup. Je-li hodnota parametru MINONE určena aplikací, hodnota pole MDMID v deskriptoru MQMD se při návratu z volání nezmění.

Další informace o zachovaných příručkách naleznete v popisu PMRET v souboru PMOPT .

Pokud je zpráva vložena do distribučního seznamu, správce front generuje podle potřeby jedinečné identifikátory zpráv, ale hodnota pole MDMID v produktu MQMD se nezmění při návratu z volání, i když byla zadána hodnota MINONE nebo PMNMTID. Pokud aplikace potřebuje znát identifikátory zpráv generované správcem front, musí aplikace poskytnout záznamy MQPMR obsahující pole PRMID .

Odesílající aplikace může také určit konkrétní hodnotu pro identifikátor zprávy, jiné než MINONE; tím se zastaví správce front, který generuje jedinečný identifikátor zprávy. Aplikace, která přeposílá zprávu, může tuto poskytovanou službu využít k šíření identifikátoru zprávy původní zprávy.

Správce front sám o sobě nepoužívá žádné použití tohoto pole kromě následujících:

- Generovat jedinečnou hodnotu, je-li požadována, jak je popsáno výše
- Doručí 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 s hlášením, nastaví pole MDMID tak, jak je určeno polem MDREP původní zprávy, buď RONMI nebo ROPMI. Aplikace, které generují zprávy sestav, by také měly dělat toto.

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 vrátí další zprávu ve frontě, ale pokud je požadována určitá zpráva, lze ji získat zadáním jednoho nebo více pěti výběrových kritérií v libovolné kombinaci; tato pole jsou:

- MDMID
- MDCID
- MDGID
- MDSEQ
- MDOFF

Aplikace nastaví jeden nebo více těchto polí na požadované hodnoty a poté nastaví odpovídající volby MO* v poli GMMO v produktu MQGMO, aby označovaly, že tato pole by měla být použita jako kritéria výběru. Pouze zprávy, které mají uvedené hodnoty v těchto polích, jsou kandidáty na načtení. Předvolba pro pole GMMO (pokud není změněna aplikací) má odpovídat jak identifikátoru zprávy, tak i identifikátoru korelace.

Za normálních okolností je vrácena první zpráva ve frontě, která splňuje kritéria výběru. Je-li však uveden GMBRWN, vrácená zpráva je další zpráva, která splní kritéria výběru; skenování pro tuto zprávu začíná zprávou za aktuální pozicí kurzoru.

Poznámka: Fronta je skenována sekvenčně pro zprávu, která odpovídá kritériím výběru, takže časy načtení budou pomalejší, než když nejsou uvedena žádná kritéria výběru, zvláště pokud se má před nalezenou vhodnou zprávou vyhledat mnoho zpráv.

Viz Tabulka 1 , kde získáte další informace o tom, jak jsou kritéria výběru použita v různých situacích.

Uvedení parametru MINONE jako identifikátoru zprávy má stejný účinek jako neuvedení MOMSGI, to znamená, že jakýkoli identifikátor zprávy se bude shodovat.

Toto pole je ignorováno, pokud je volba GMMUC zadána v parametru **GMO** na volání MQGET.

Při návratu z volání MQGET je pole MDMID nastaveno na identifikátor zprávy vrácené zprávy (je-li k dispozici).

Mohou být použity následující speciální hodnoty:

MINON
Není uveden žádný identifikátor zprávy.

Hodnota je binární nula pro délku pole.

Jedná se o vstupní/výstupní pole pro volání MQGET, MQPUT a MQPUT1 . Délka tohoto pole je dána LNMID. Počáteční hodnota tohoto pole je MINONE.

MDMT (10ciferné celé číslo se znaménkem)

Typ zprávy.

Označuje typ zprávy. Typy zpráv jsou seskupeny následujícím způsobem:

MTSFST
Nejnižší hodnota pro systémem definované typy zpráv.

MTSLST
Nejvyšší hodnota pro typy zpráv definované systémem.

V rozsahu systému jsou momentálně definovány následující hodnoty:

MTDGRM
Zpráva nevyžadující odpověď.

Zpráva je taková, která nevyžaduje odpověď.

MTRQST
Zpráva vyžadující odpověď.

Zpráva je taková, která vyžaduje odpověď.

Název fronty, do které má být odeslána odpověď, musí být zadán v poli MDRQ . Pole MDREP udává, jak mají být nastaveny hodnoty MDMID a MDCID odpovědi.

MEZE
Odpovězte na předchozí zprávu požadavku.

Zpráva je odpovědí na předchozí zprávu požadavku (MTRQST). Zpráva by měla být odeslána do fronty uvedené v poli MDRQ zprávy požadavku. Pole MDREP požadavku by mělo být použito pro řízení toho, jak jsou nastaveny MDMID a MDCID odpovědi.

Poznámka: Správce front nevynucuje vztah požadavek-odezva. Jedná se o zodpovědnost aplikace.

MTRPRT
Zpráva sestavy.

Zpráva se hlásí k očekávanému nebo neočekávanému výskytu, obvykle souvisí s nějakou jinou zprávou (například byla přijata zpráva požadavku, která obsahovala neplatná data). Zpráva by měla být odeslána do fronty uvedené v poli MDRQ deskriptoru zprávy původní zprávy. Pole MDFB by mělo být nastaveno tak, aby určovalo povahu sestavy. Pole MDREP původní zprávy lze použít k určení způsobu nastavení MDMID a MDCID zprávy sestavy.

Zprávy sestav generované správcem front nebo agentem oznamovacího kanálu jsou vždy odesílány do fronty MDRQ s dříve popsanými poli MDFB a MDCID .

Další hodnoty v rozsahu systému mohou být definovány v budoucích verzích rozhraní MQI a jsou přijímány voláními MQPUT a MQPUT1 bez chyby.

Lze také použít hodnoty definované aplikací. Musí být v následujícím rozsahu:

MTAFST
Nejnižší hodnota pro typy zpráv definované aplikací.

MTALST
Nejvyšší hodnota pro typy zpráv definované aplikací.

V případě volání MQPUT a MQPUT1 musí být hodnota MDMT buď v rozsahu definovaném systémem, nebo v rozsahu definovaném aplikací; pokud tomu tak 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 (desetimístné podepsané celé číslo)

Posunutí dat ve fyzické zprávě od začátku logické zprávy.

Toto je posun v bajtech dat ve fyzické zprávě od začátku logické zprávy, z níž jsou data součástí. Tato data se nazývají *segment*. Posunutí je v rozsahu od 0 do 999 999 999. Fyzická zpráva, která není segmentem logické zprávy, má posun nula.

Toto pole nemusí být nastaveno aplikací v rámci volání MQPUT nebo MQGET, pokud:

- Na volání MQPUT je uveden PMLOGO.
- Na volání MQGET není parametr MOOFFS určen.

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 produkt MDOFF nastaven na příslušnou hodnotu.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v tabulce Tabulka 1. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou.

Pro hlášení zpráv sestavy v segmentu logické zprávy je pole MDOLN (za předpokladu, že není OLUNDF) použito pro aktualizaci offsetu v informacích o segmentu uchovaných správcem front.

Na vstupu do volání MQGET používá správce front hodnotu popsanou v tabulce Tabulka 1. Na výstupu z volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je nula. Toto pole je ignorováno, pokud MDVER je menší než MDVER2.

MDOLN (10ciferné celé číslo se znaménkem)

Délka původní zprávy.

Toto pole má význam pouze u zpráv sestav, které jsou segmenty. Určuje délku segmentu zprávy, k němuž se zpráva sestavy vztahuje; neudává délku logické zprávy, jejíž část tvoří část formuláře, ani délku dat ve zprávě sestavy.

Poznámka: Při generování zprávy sestavy pro zprávu, která je segmentem, se kopie správce front a agent kanálu zpráv do MQMD pro zprávu hlásí do polí MDGID, MDSEQ, MDOFFa *MDMFL*, v polích z původní zprávy. V důsledku toho je zpráva zprávy také segmentem. Aplikace, které generují zprávy sestav, se doporučují stejné, a aby se zajistilo, že pole MDOLN je nastaveno správně.

Je definována následující speciální hodnota:

OLUNDFE.
Původní délka zprávy není definována.

MDOLN je vstupní pole pro volání MQPUT a MQPUT1 , ale hodnota poskytnutá aplikací je přijata pouze za určitých okolností:

- Je-li odesílaná zpráva segmentem a je také zprávou sestavy, přijme správce front zadanou hodnotu. Hodnota musí být:
  - Větší než nula, pokud segment není posledním segmentem
  - Ne méně než nula, je-li segment posledním segmentem
  - Ne méně než délka dat přítomných ve zprávě

  Nejsou-li tyto podmínky splněny, volání selže s kódem příčiny RC2252.
- Je-li odesílaná zpráva segment, ale ne zpráva sestavy, správce front ignoruje pole a použije místo toho délku dat zprávy aplikace.
- Ve všech ostatních případech správce front ignoruje pole a místo toho použije hodnotu OLUNDF.

Jedná se o výstupní pole ve volání MQGET.

Počáteční hodnota tohoto pole je OLUNDF. Toto pole je ignorováno, pokud MDVER je menší než MDVER2.

MDPAN (28bajtový znakový řetězec)

Název aplikace, která vložila zprávu.

Toto je část *kontextu původu* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Formát hodnoty MDPAN závisí na hodnotě MDPAT.

Je-li toto pole nastaveno správcem front (tj. pro všechny volby kromě PMSETA), je nastaven na hodnotu, která je určena prostředím:

- z/OS V systému z/OSpoužívá správce front následující:
  - Pro dávku produktu z/OS jde o 8znakový název úlohy z karty JES JOB
  - Pro TSO se jedná o 7znakový identifikátor uživatele TSO.
  - Pro CICSje osmiznakový identifikátor Applid následován čtyřmístným tranID
  - Pro IMS, 8znakový identifikátor systému IMS , následovaný 8místným názvem PSB
  - Pro XCF, 8znakový název skupiny XCF následovaný 16znakovým názvem člena XCF
  - Pro zprávu vygenerovanou správcem front je prvních 28 znaků názvu správce front
  - Pro distribuované ukládání do fronty bez CICSje osmiznakový název úlohy inicializátoru kanálu následován osmiznakovým názvem modulu, který vkládá do fronty nedoručených zpráv, za nímž následuje 8znakový identifikátor úlohy.
  - For MQSeries Java language bindings processing with IBM MQ for z/OS the 8-character jobname of the address space created for the z/OS UNIX System Services environment. Obvykle se jedná o identifikátor uživatele TSO s připojeným jedním numerickým znakem.

  Název nebo názvy jsou doplněny mezerami vpravo s mezerami, stejně jako každý prostor ve zbytku pole. Pokud existuje více než jedno jméno, mezi nimi není oddělovač.
- Windows V systémech DOS v PC a Windows používá správce front:
  - Pro aplikaci CICS se název transakce CICS
  - Pro aplikaci, která není typuCICS , je nejvíce 28 znaků plně kvalifikovaného názvu spustitelného souboru.
- IBM i V systému IBM isprávce front používá plně kvalifikované jméno úlohy.
- Linux AIX V systému AIX and Linuxpoužívá správce front následující:
  - Pro aplikaci CICS se název transakce CICS
  - Pro aplikaci, která není typuCICS , má nejvíce 14 znaků plně kvalifikovaného názvu spustitelného souboru, je-li k dispozici pro správce front, a mezery jinak (například na AIX)
- V systému VSE/ESAsprávce front používá osmiznakový identifikátor Applid následovaný 4místným transidem.

Pro volání MQPUT a MQPUT1 je to vstupní/výstupní pole, je-li položka PMSETA zadána v parametru **PMO** . Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny na mezery. Není-li položka PMSETA uvedena, je toto pole na vstupu ignorováno a je to 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 (10číslicové podepsané celé číslo)

Typ aplikace, která vložila zprávu.

Toto je část **kontextu původu** zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

MDPAT může mít jeden z následujících standardních typů. Mohou být také použity uživatelsky definované typy, ale měly by být omezeny na hodnoty v rozsahu ATUFST přes ATULST.

ATAIX
Aplikace AIX (stejná hodnota jako ATUNIX).

ATBRKR
Broker.

ATTIKA
CICS .

ATCICBA
CICS bridge.

ATVSE
CICS/VSE .

ATDOSI
IBM MQ MQI client na PC DOS.

ATDQM
Distribuovaný agent správce front.

ATGUAR
Aplikace Tandem Guardian (stejná hodnota jako ATNSK).

ATIMÁTY
IMS .

ATIMB
Most IMS .

ATJAVA.
Java.

FUNKCE ATMVS
Aplikace MVS nebo TSO (stejná hodnota jako ATZOS).

ATNOTE
Lotus Notes Agent.

ATNSCITY
Aplikace jádra Tandem NonStop .

AT390
Aplikace OS/390 (stejná hodnota jako ATZOS).

AT400
IBM i .

ATQM.
Správce front.

ATUNIX
UNIX .

ATVOS
Aplikace Stratus VOS.

ATWIN
16bitová aplikace Windows .

ATWINT
32bitovou aplikaci Windows .

ATXCF
XCF.

AZOS
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 prostředí.

SKÁČ
Neznámý typ aplikace.

Tato hodnota může být použita k označení, že typ aplikace je neznámý, i když jsou přítomné jiné informace o kontextu.

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULSTCITY
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 v případě, že je zpráva vložena bez kontextu (tj. je zadána volba kontextu PMNOC).

Když je zpráva načtena, lze pro tuto hodnotu testovat MDPAT , aby se rozhodlo, zda má zpráva kontext (doporučuje se, že MDPAT není nikdy nastaven na ATNCON, a to aplikací pomocí PMSETA, je-li nějaká z ostatních kontextových polí neprázdná).

ATSIB
Označuje, že zpráva pochází z jiného produktu systému zpráv produktu IBM MQ a byla doručena prostřednictvím mostu SIB (Service Integration Bus).

Když správce front vygeneruje tyto informace v důsledku vložení aplikace, je pole nastaveno na hodnotu určenou prostředím.

IBM i Všimněte si, že v IBM ije pole nastaveno na AT400; správce front nikdy nepoužívá ATCICS na IBM i.

Pro volání MQPUT a MQPUT1 je to vstupní/výstupní pole, je-li položka PMSETA zadána v parametru **PMO** . Není-li položka PMSETA uvedena, je toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDPAT , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDPAT , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDPAT , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDPAT ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, je pole nastaveno na hodnotu 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 *kontextu původu* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Formát použitý pro datum, kdy je toto pole generováno správcem front, je:

- YYYYMMDD

kde znaky představují:

yyyy
rok (čtyři číselné číslice)

MM
měsíc v roce (01 až 12)

DD
den v měsíci (01 až 31)

Čas GMT (Greenwich Mean Time) se používá pro pole MDPD a MDPT , přičemž se použijí systémové hodiny přesně nastavené na GMT.

Pokud byla zpráva vložena jako součást pracovní jednotky, datum je datum, kdy byla zpráva vložena, a nikoli datum, kdy byla transakce potvrzena.

Pro volání MQPUT a MQPUT1 je to vstupní/výstupní pole, je-li položka PMSETA zadána v parametru **PMO** . Obsah pole nekontroluje správce front, s tím rozdílem, že všechny informace, které následují za znakem null uvnitř pole, jsou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny na mezery. Není-li položka PMSETA uvedena, je toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDPD , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDPD , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDPD , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDPD ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána LNPDAT. Počáteční hodnota tohoto pole je 8 prázdných znaků.

MDPER (10ciferné celé číslo se znaménkem)

Perzistence zpráv.

Označuje, zda zpráva přežije selhání systému a restartuje správce front. Pro volání MQPUT a MQPUT1 musí být hodnota jedna z následujících:

PÍPČ
Zpráva je trvalá.

To znamená, že zpráva přečká selhání systému a restartuje správce front. Jakmile je zpráva vložena a jednotka práce s putter (je-li zpráva vložena jako součást pracovní jednotky), je zpráva uchována v pomocné paměti. Zůstane tam, dokud nebude zpráva odebrána z fronty a jednotka procesu getter (je-li zpráva načtena jako část pracovní jednotky).

Když se do vzdálené fronty odešle trvalá zpráva, použije se k uchování zprávy v každém správci front v každém správci front místo určení, dokud není známo, že dorazila do dalšího správce front, dokud není známo, že se zpráva dostala do dalšího správce front.

Trvalé zprávy nelze umístit na:

- Dočasné dynamické fronty
- Sdílené fronty, v nichž je úroveň struktury prostředku Coupling Facility menší než tři, nebo struktura prostředku Coupling Facility není obnovitelná.

Trvalé zprávy lze umístit do trvalých dynamických front, předdefinovaných front a sdílených front, kde úroveň struktury prostředku Coupling Facility je 3, a prostředek Coupling Facility je obnovitelný.

PENPER
Zpráva není trvalá.

To znamená, že zpráva normálně nepřežije selhání systému nebo restartuje správce front. To platí i v případě, že se během restartu správce front nachází neporušená kopie zprávy v pomocné paměti.

Ve speciálním případě sdílených front přežijí přechodné zprávy *do* restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility použitého k ukládání zpráv ve sdílených frontách.

DEFINICE PEQDEF
Zpráva má výchozí trvání.

- Je-li fronta fronta klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného v cílovém správci front, který vlastní danou instanci fronty, na které je zpráva umístěna. Obvykle mají všechny instance fronty klastru stejnou hodnotu atributu **DefPersistence** , i když to není nařízeno.

  Při umístění zprávy do cílové fronty je hodnota parametru **DefPersistence** zkopírována do pole *MDPER* . Je-li produkt **DefPersistence** později změněn, nebudou ovlivněny zprávy, které již byly umístěny do fronty.
- Není-li fronta fronta klastru, je perzistence zprávy převzata z atributu **DefPersistence** definovaného v lokálním správci front, i když je správce cílové fronty vzdálený.

  Je-li v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude použita výchozí perzistence z hodnoty tohoto atributu v první definici v cestě. To může být:

  – Fronta aliasů
  – Lokální fronta
  – Lokální definice vzdálené fronty
  – Alias správce front
  – Přenosová fronta (například fronta `DefXmitQName`)

  Hodnota parametru **DefPersistence** se při vložení zprávy zkopíruje do pole `MDPER` . Pokud je produkt **DefPersistence** později změněn, zprávy, které již byly vloženy, nejsou ovlivněny.

Trvalé i přechodné zprávy mohou existovat ve stejné frontě.

Při odpovídání na zprávu by aplikace měly normálně používat pro odpověď zprávu trvalost zprávy vzniklé při zpracování požadavku.

Pro volání MQGET je vrácena hodnota 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 (10ciferné celé číslo se znaménkem)

Priorita zprávy.

Pro volání MQPUT a MQPUT1 musí být hodnota větší než nula nebo rovna nule; hodnota nula je nejnižší priorita. Je možné použít také následující speciální hodnotu:

DEFINICE PRQDEF
Výchozí priorita pro frontu.

- Je-li fronta fronta klastru, je priorita zprávy převzata z atributu **DefPriority** , jak je definováno na cílovém správci front, který vlastní danou instanci fronty, na které je zpráva umístěna. Obvykle mají všechny instance fronty klastru stejnou hodnotu atributu **DefPriority** , i když to není nařízeno.

  Při umístění zprávy do cílové fronty je hodnota parametru **DefPriority** zkopírována do pole `MDPRI` . Je-li produkt **DefPriority** později změněn, nebudou ovlivněny zprávy, které již byly umístěny do fronty.
- Pokud fronta není fronta klastru, je priorita zprávy převzata z atributu **DefPriority** podle definice v lokálním správci front, a to i v případě, že je správce cílové fronty vzdálený.

  Je-li v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude z hodnoty tohoto atributu použita výchozí priorita v první definici v cestě. To může být:

- Fronta aliasů
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta DefXmitQName)

Hodnota parametru **DefPriority** se při vložení zprávy zkopíruje do pole MDPRI . Pokud je produkt **DefPriority** později změněn, zprávy, které již byly vloženy, nejsou ovlivněny.

Hodnota vrácená voláním MQGET je vždy větší než nebo rovna nule; hodnota PRQDEF není nikdy vrácena.

Je-li zpráva vložena s prioritou vyšší, než je maximum podporované lokálním správcem front (toto maximum je přiděleno atributem správce front produktu **MaxPriority**), zpráva je přijata správcem front, ale zařazena do fronty v maximální prioritě správce front; volání MQPUT nebo MQPUT1 je dokončeno s CCWARN a kódem příčiny RC2049. V poli MDPRI je však zachována hodnota zadaná aplikací, která zprávu vložila.

Při odpovídání na zprávu by aplikace měly normálně používat pro odpověď zprávy prioritu zprávy požadavku. V jiných situacích umožňuje uvedení PRQDEF, aby bylo ladění prováděno 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 **kontextu původu** zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Formát použitý pro čas, kdy je toto pole generováno správcem front, je:

- HHMMSTH

kde znaky představují (v pořadí):

HH
hodin (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: Je-li časová základna systému synchronizována s velmi přesným časovým standardem, je možné ve vzácných případech vrátit hodnotu 60 nebo 61 po dobu sekund v produktu MDPT. To se stane, když se do globálního časového standardu vloží přestupné sekundy.

Čas GMT (Greenwich Mean Time) se používá pro pole MDPD a MDPT , přičemž se použijí systémové hodiny přesně nastavené na GMT.

Pokud byla zpráva vložena jako část pracovní jednotky, je čas, kdy byla zpráva vložena, a nikoli čas, kdy byla transakce potvrzena.

Pro volání MQPUT a MQPUT1 je to vstupní/výstupní pole, je-li položka PMSETA zadána v parametru **PMO** . Obsah pole nekontroluje správce front, s tím rozdílem, že všechny informace, které následují za znakem null uvnitř pole, jsou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny

na mezery. Není-li položka PMSETA uvedena, je toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat hodnotu MDPT , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDPT , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných publikacích), ale nepoužívá se jako MDPT , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDPT ve všech publikačních publikacích, které se na ně posílají. 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 (10ciferné celé číslo se znaménkem)

Volby pro zprávy sestav.

Zpráva sestavy je zpráva o jiné zprávě, která se používá k informování aplikace o očekávaných nebo neočekávaných událostech, které se vztahují k původní zprávě. Pole MDREP umožňuje aplikaci odesláním původní zprávy určit, které zprávy sestavy jsou povinné, zda mají být data zprávy aplikace zahrnuta do nich, a také (pro sestavy i odpovědi), jak mají být nastaveny zprávy a identifikátory korelace v sestavě nebo zprávě odpovědi. Je možné požadovat libovolný nebo žádný (nebo žádný) z následujících typů zpráv sestavy:

- Výjimka
- Konec platnosti
- Potvrdit při příchodu (COA)
- Potvrdit při doručení (COD)
- Pozitivní upozornění na akci (PAN)
- Negativní upozornění na akci (NAN)

Je-li vyžadována více než jeden typ zprávy sestavy nebo jsou potřebné jiné volby sestavy, lze tyto hodnoty přidat společně (nepřidávat stejnou konstantu víckrát než jednou).

Aplikace, která přijímá zprávu sestavy, může určit příčinu, proč byla sestava generována, tak, že prozkoumáte pole MDFB v MQMD; další podrobnosti viz pole MDFB .

Použití voleb sestavy při vkládání zprávy do tématu může způsobit generování zprávy nebo generování zpráv sestav a odeslání zprávy do aplikace. Důvodem je skutečnost, že zpráva o publikování může být odeslána na nulu, jedna nebo více odebírajících aplikací.

Volby výjimky: Můžete uvést jednu z následujících možností, abyste požádali o zprávu hlášení výjimek.

ROAKTIVITA

Vyžadované sestavy aktivity

Tato volba sestavy umožňuje generování sestavy aktivity, kdykoli je zpracována zpráva s touto sadou voleb sestavy podporou aplikací.

Zprávy s touto sadou voleb sestavy musí být akceptovány kterýchkoli správcem front, a to i v případě, že nerozumí této volbě. To umožňuje nastavení volby sestavy na libovolné uživatelské zprávě, i když jsou zpracovány předchozími správci front. K dosažení tohoto cíle je volba sestavy umístěna do podpole ROAUM.

Pokud proces (správce front nebo uživatelský proces) provádí aktivitu na zprávě se sadou ROACT, může se rozhodnout vygenerovat a vložit sestavu aktivity.

Volba sestavy o aktivitě umožňuje trasování libovolné zprávy v rámci sítě správce front. Volba sestavy může být uvedena na libovolné aktuální zprávě uživatele a okamžitě může začít spočítat trasu zprávy přes síť. Pokud aplikace, která generuje zprávu, nemůže povolit generování sestavy o aktivitě, lze ji povolit pomocí výstupního bodu rozhraní API dodaného administrátory správce front.

Pro sestavy aktivit lze použít několik podmínek:

1. Přenosová cesta bude méně podrobná, pokud je v síti méně správců front, které mohou generovat sestavy o aktivitě.
2. Sestavy aktivit nemusí být snadno 'objednatelné', aby bylo možné určit trasu, která byla přijata.
3. Zprávy o činnosti nemusí být schopny najít trasu k požadovanému cíli.

ROEXC
Požadována hlášení výjimek.

Tento typ sestavy může generovat agent kanálu zpráv při odeslání zprávy do jiného správce front a tuto zprávu nelze doručit do zadané cílové fronty. Například cílová fronta nebo intermediační přenosová fronta může být plná, nebo může být zpráva příliš velká pro frontu.

Generování zprávy o výjimce závisí na perzistenci původní zprávy a na rychlosti kanálu zpráv (normální nebo rychlé), přes kterou se původní zpráva pohybuje:

- Pro všechny trvalé zprávy a pro přechodné zprávy, které cestují prostřednictvím běžných kanálů zpráv, se sestava výjimek generuje pouze v případě, že akce určená odesílající aplikací pro chybový stav může být úspěšně dokončena. Odesílající aplikace může určit jednu z následujících akcí k řízení dispozice původní zprávy, když dojde k chybovému stavu:
  - RODLQ (to způsobí umístění původní zprávy do fronty nedoručených zpráv).
  - RODISC (to způsobí vyřazení původní zprávy).

  Pokud nemůže být akce určená odesílající aplikací úspěšně dokončena, bude původní zpráva ponechána na přenosové frontě a nebude vygenerována žádná zpráva o výjimce.
- V případě přechodných zpráv, které cestují prostřednictvím rychlých kanálů zpráv, je původní zpráva odebrána z přenosové fronty a vygenerovaná zpráva o výjimce i v případě, že zadaná akce pro chybový stav nemůže být úspěšně dokončena. Je-li například uveden parametr RODLQ, ale původní zprávu nelze umístit do fronty nedoručených zpráv, protože je tato fronta plná, vygeneruje se zpráva výjimky a původní zpráva byla vyřazena.

  Další informace o normálních a rychlých kanálech zpráv najdete v tématu Perzistence zpráv .

Sestava výjimek se negeneruje, pokud aplikace, která vložila původní zprávu, může být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 .

Aplikace mohou také odesílat zprávy o výjimkách, což znamená, že zpráva, kterou přijal, nemůže být zpracována (například proto, že se jedná o debetní transakci, která by způsobila, že účet překročí svůj úvěrový limit).

Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z ROEXC, ROEXCD a ROEXCF.

REXCD
Požadují se zprávy s údaji o výjimkách.

To je stejné jako ROEXC, kromě toho, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jeden z ROEXC, ROEXCD a ROEXCF.

VÝMLUVU
Sestavy výjimek s úplnými požadovanými daty.

To je stejné jako ROEXC, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jeden z ROEXC, ROEXCD a ROEXCF.

Volby ukončení platnosti: Můžete uvést jednu z následujících voleb pro vyžádání zprávy hlášení o vypršení platnosti.

VÝMĚNNÉ
Povinné sestavy vypršení platnosti.

Tento typ sestavy je generován správcem front, pokud je zpráva vyřazena před doručením do aplikace, protože uplynul její čas ukončení platnosti (viz pole MDEXP). Není-li tato volba nastavena, nebude generována žádná zpráva sestavy, je-li 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 jedno z ROEXP, ROEXPD a ROEXPF.

ROEXPD
Zprávy o vypršení platnosti s požadovanými daty.

To je stejné jako hodnota 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 produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jedno z ROEXP, ROEXPD a ROEXPF.

ROEXPF
Zprávy o vypršení platnosti s požadovanými úplnými daty.

To je stejné jako hodnota ROEXP, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jedno z ROEXP, ROEXPD a ROEXPF.

Volby potvrzení při příjmu: Můžete uvést jednu z následujících voleb pro vyžádání zprávy hlášení potvrzení o příjmu.

ROCOA
Vyžaduje se potvrzení o přijetí sestav.

Tento typ sestavy je generován správcem front, který vlastní cílovou frontu, je-li 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 frontou, bude zpráva COA vygenerovaná správcem front k dispozici pro načtení pouze tehdy, je-li jednotka práce potvrzena.

Sestava COA se nevygeneruje, pokud je pole MDFMT v deskriptoru zprávy FMXQH nebo FMDLH. Zabráníte tak vygenerování sestavy COA, pokud je zpráva vložena do přenosové fronty nebo je nedoručitelná a vložena do fronty nedoručených zpráv.

Neuvádějte více než jednu z hodnot ROCOA, ROCOAD a ROCOAF.

ROCOAD
Vyžaduje se potvrzení o přijetí dat s požadovanými daty.

To je stejné jako ROCOA, kromě toho, že první 100 bajtů dat zprávy aplikace z původní zprávy je zahrnuto do zprávy sestavy. Pokud původní zpráva obsahuje jednu nebo více struktur záhlaví produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Neuvádějte více než jednu z hodnot ROCOA, ROCOAD a ROCOAF.

ROCOAF
Hlášení o potvrzení-on-arrival s úplnými požadovanými daty.

Je to stejné jako ROCOA, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Neuvádějte více než jednu z hodnot ROCOA, ROCOAD a ROCOAF.

Volby vyřazení a vypršení platnosti: Můžete zadat následující volbu, chcete-li nastavit dobu vypršení platnosti a příznak vyřazení pro zprávy sestav.

JELENOVITÍ
Nastavit dobu vypršení platnosti zprávy sestavy a příznak vyřazení.

Tato volba zajišťuje, že zprávy sestavy a zprávy odpovědí dědí čas vypršení platnosti a příznak zahození (zda mají být zahozena či nikoli), z původních zpráv. Pomocí této sady voleb zprávy a zprávy s odpovědí:

1. Zdědit příznak RODISC (je-li nastaven).
2. Zdědit zbývající dobu vypršení platnosti zprávy, pokud zpráva není hlášením o vypršení platnosti. Je-li zpráva o vypršení platnosti, je doba vypršení platnosti nastavena na 60 sekund.

Při použití této sady voleb platí následující pravidla:

Poznámka:

1. Zprávy a zprávy s odpovědí jsou generovány s příznakem vyřazení a hodnotou vypršení platnosti a nemohou zůstat v systému.
2. Zprávy přenosové cesty se nemohou dostat do cílových front na správcích front s povolenou přenosovou cestou, než je trasování přenosové cesty.
3. Fronty jsou zabráněné vyplněným sestavami, které nelze doručit, jsou-li přerušeny komunikační vazby.
4. Odpovědi na příkazový server dědí zbývající vypršení platnosti požadavku.

Volby potvrzení při doručení: Můžete zadat jednu z následujících voleb pro vyžádání zprávy sestavy potvrzení o doručení.

TRESKA OBECNÁ
Vyžaduje se povinné hlášení o doručení.

Tento typ sestavy je generován správcem front v případě, že aplikace načte zprávu z cílové fronty způsobem, který způsobí odstranění zprávy z fronty. Data zprávy z původní zprávy nejsou zahrnuta do zprávy sestavy.

Je-li zpráva načtena jako součást pracovní jednotky, vygeneruje se zpráva sestavy v rámci stejné pracovní jednotky, takže sestava nebude k dispozici, dokud nebude potvrzena jednotka práce. Je-li jednotka práce zálohována, sestava se neodešle.

Sestava COD není generována, je-li pole MDFMT v deskriptoru zprávy FMDLH. Zabráníte tak vygenerování sestavy COD, pokud je zpráva nedoručitelná a vložena do fronty nedoručených zpráv.

ROCOD není platný, je-li cílová fronta frontou XCF.

Neuvádějte více než jeden ROCOD, ROCODD a ROCODF.

RODOKD
Potvrzení o doručení s údaji vyžadovaným pro doručení.

To 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 produktu MQ , jsou obsaženy ve zprávě sestavy spolu s údaji o velikosti 100 bajtů dat aplikace.

Je-li GMATM zadán na volání MQGET pro původní zprávu a načtená zpráva je zkrácena, množství dat zprávy aplikace umístěné ve zprávě sestavy je minimem pro:

- Délka původní zprávy
- 100 bajtů.

Hodnota ROCODD není platná, je-li cílová fronta frontou XCF.

Neuvádějte více než jeden ROCOD, ROCODD a ROCODF.

RODCODF
Zprávy Confirm-on-delivery s požadovanými úplnými údaji.

To je stejné jako ROCOD, kromě toho, že všechna data zprávy aplikace z původní zprávy jsou zahrnuta do zprávy sestavy.

Identifikátor ROCODF není platný, je-li cílová fronta frontou XCF.

Neuvádějte více než jeden ROCOD, ROCODD a ROCODF.

Volby oznámení akce: Můžete zadat jednu nebo obě následující volby, chcete-li požádat, aby přijímající aplikace odeslala zprávu s kladnou akcí nebo se zprávou s negativním výsledkem.

ROPAN
Povinné sestavy upozornění na akci jsou povinné.

Tento typ sestavy je generován aplikací, která danou zprávu načte a jedná s ním. Zpráva označuje, že akce požadovaná ve zprávě byla úspěšně provedena. Aplikace, která generuje sestavu, určuje, zda má být nějaká data zahrnuta do sestavy.

Kromě odeslání tohoto požadavku do aplikace při načítání zprávy správce front nepodnikává žádnou akci založenou na této volbě. Je-li to vhodné, odpovídá za načtení zprávy za účelem získání sestavy.

RONAN
Požadují se sestavy upozornění na negativní akci.

Tento typ sestavy je generován aplikací, která danou zprávu načte a jedná s ním. Znamená to, že akce požadovaná ve zprávě nebyla úspěšně provedena. Aplikace, která generuje sestavu, určuje, zda má být nějaká data zahrnuta do sestavy. Může být například žádoucí zahrnout data označující, proč požadavek nemohl být proveden.

Kromě odeslání tohoto požadavku do aplikace při načítání zprávy správce front nepodnikává žádnou akci založenou na této volbě. Je-li to vhodné, odpovídá za načtení zprávy za účelem získání sestavy.

Určení, které podmínky odpovídají kladnému účinku a které odpovídají negativním opatřením, je odpovědnost za žádost. Doporučuje se však, aby byl-li požadavek proveden pouze částečně, měla by být vygenerována zpráva NAN spíše než zpráva PAN, pokud se o to požádá. Doporučuje se také, aby každá možná podmínka měla odpovídat buď kladné akci, nebo záporné akci, ale ne oběma.

Volby identifikátoru zprávy: Můžete určit jednu z následujících voleb, které řídí, jak se má nastavit `MDMID` zprávy sestavy (nebo zprávy odpovědi).

RONMI
Identifikátor nové zprávy.

Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď generována jako výsledek této zprávy, vygeneruje se nová `MDMID` pro zprávu nebo zprávu odpovědi.

ROPMI
Předat identifikátor zprávy.

Je-li zpráva nebo odpověď generována jako výsledek této zprávy, `MDMID` této zprávy se má zkopírovat do `MDMID` sestavy nebo zprávy odpovědi.

`MsgId` publikační zprávy bude pro každého odběratele, který obdrží kopii publikace, jinak, a proto se `MsgId` zkopírovaný do sestavy nebo zprávy odpovědi bude pro každou z nich lišit.

Není-li tato volba zadána, předpokládá se RONMI.

Volby identifikátoru korelace: Můžete určit jednu z následujících možností, jak určit, jak má být nastavena `MDCID` zprávy sestavy (nebo zprávy odpovědi).

ROMTMTCA
Kopírovat identifikátor zprávy do identifikátoru korelace.

Jedná se o výchozí akci a označuje, že pokud je sestava nebo odpověď generována jako výsledek této zprávy, `MDMID` této zprávy se má zkopírovat do sestavy `MDCID` sestavy nebo zprávy odpovědi.

Pro každého odběratele, který obdrží kopii publikace, se bude pro každého odběratele lišit `MsgId` , a proto se `MsgId` kopie souboru sestavy nebo zprávy odpovědi do sestavy `CorrelId` bude lišit pro každou z nich.

ROPCI
Předat identifikátor korelace.

Je-li zpráva nebo odpověď generována jako výsledek této zprávy, MDCID této zprávy se má zkopírovat do MDCID sestavy nebo zprávy odpovědi.

MDCID publikační zprávy bude specifické pro odběratele, pokud nepoužije volbu SOSCID a nastaví pole SCDIC v MQSD na CINONE. Proto je možné, že se MDCID zkopírovaný do sestavy MDCID sestavy nebo zprávy odpovědi bude pro každou z nich lišit.

Není-li tato volba zadána, předpokládá se hodnota ROCMTC.

Servery odpovídání na požadavky nebo generování zpráv sestav se doporučuje zkontrolovat, zda byly volby ROPMI nebo ROPCI nastaveny v původní zprávě. Pokud by byly, měly by servery provést akci popsanou pro tyto volby. Není-li nastaven ani jeden z nich, servery by měly provést odpovídající výchozí akci.

: Můžete určit jednu z následujících voleb pro řízení odebrání původní zprávy, pokud ji nelze doručit do cílové fronty. Tyto volby se týkají pouze situací, které by vedly ke generování zprávy hlášení výjimek, pokud by byla jedna požadována odesílající aplikací. Aplikace může nastavit volby odebrání nezávisle na požadování sestav výjimek.

RODLQ
Umístit zprávu do fronty nedoručených zpráv.

Jedná se o výchozí akci a označuje, že zpráva by měla být umístěna do fronty nedoručených zpráv, pokud tuto zprávu nelze doručit do cílové fronty. K tomu dojde v následujících situacích:

- Když aplikace, která zadala původní zprávu, nemůže být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Vygeneruje se zpráva hlášení o výjimce, pokud ji někdo požadoval odesilatel.
- Když byla aplikace, která vložila původní zprávu, do tématu vložena

Bude vygenerována zpráva o výjimce, pokud byla vyžádána odesílatelem.

RODISK
Zahodit zprávu.

Tato zpráva informuje o tom, že zpráva by měla být vyřazena, pokud ji nelze doručit do cílové fronty. K tomu dojde v následujících situacích:

- Když aplikace, která zadala původní zprávu, nemůže být synchronně oznámena problému prostřednictvím kódu příčiny vráceného voláním MQPUT nebo MQPUT1 . Vygeneruje se zpráva hlášení o výjimce, pokud ji někdo požadoval odesilatel.
- Když byla aplikace, která vložila původní zprávu, do tématu vložena

Bude vygenerována zpráva o výjimce, pokud byla vyžádána odesílatelem.

Je-li požadováno vrácení původní zprávy odesílateli, aniž by byla do fronty nedoručených zpráv vložena původní zpráva, měl by odesílatel určit RODISC s ROEXCF.

Výchozí volba: Můžete uvést následující, pokud nejsou požadovány žádné volby sestavy:

RONAN
Nejsou vyžadovány žádné sestavy.

Tato hodnota může být použita k označení, že nebyly zadány žádné další volby. Hodnota RONONE je definována pro dokumentaci programu podpory. Není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze detekovat.

Obecné informace:

1. Všechny požadované typy sestav musí být výslovně vyžádány aplikací, která odesílá původní zprávu. Je-li například požadována zpráva COA, ale sestava výjimek není, vygeneruje se zpráva COA, když je zpráva umístěna do cílové fronty, ale pokud je fronta cíle zaplněna, jakmile zpráva dorazí, nebude vygenerována žádná sestava výjimek. Nejsou-li nastaveny žádné volby obslužného programu MDREP , správce front nebo agent kanálu zpráv (MCA) negeneruje žádné zprávy sestavy.

Některé volby sestavy lze zadat i v případě, že lokální správce front je nerozpoznal; je užitečný v případě, že má být tato volba zpracována cílovým správcem front. Další podrobnosti viz "Volby sestav a příznaky zpráv v systému IBM i" na stránce 1418.

Je-li požadována zpráva sestavy, musí být název fronty, do které má být sestava odeslána, uvedena v poli MDRQ . Když je přijata zpráva sestavy, charakter sestavy lze určit prozkoumáním pole MDFB v deskriptoru zprávy.

2. Pokud správce front nebo MCA, který generuje zprávu sestavy, nemůže vložit zprávu se sestavou do fronty odpovědí (například, protože fronta odpovědí nebo přenosová fronta je plná), zpráva sestavy bude umístěna místo fronty nedoručených zpráv. Pokud také selže nebo ve frontě není žádná fronta nedoručených zpráv, bude akce záviset na typu zprávy hlášení:

   - Je-li zpráva hlášení o výjimce, zpráva, která způsobila vygenerování sestavy výjimek, je ponechána na přenosové frontě, což zajišťuje, že zpráva nebude ztracena.
   - Pro všechny ostatní typy sestav je zpráva sestavy vyřazena a zpracování bude normálně pokračovat. Důvodem je to, že původní zpráva již byla doručena bezpečně (zprávy sestav COA nebo COD) nebo již není o žádný zájem (pro zprávu o vypršení platnosti zprávy).

   Jakmile byla zpráva sestavy úspěšně umístěna do fronty (cílová fronta nebo mezilehlá přenosová fronta), zpráva již není předmětem speciálního zpracování; zachází se stejně jako s jakoukoli jinou zprávou.
3. Když je sestava generována, je otevřena fronta MDRQ a zpráva sestavy nabyla pomocí oprávnění MDUID v MQMD zprávy způsobující tuto sestavu, s výjimkou následujících případů:

   - Zprávy výjimek generované přijímajícím agentem MCA jsou při pokusu o vložení zprávy způsobující vložení zprávy použity bez ohledu na to, jakou má agent MCA práci. Atribut kanálu CDPA určuje použitý identifikátor uživatele.
   - Sestavy COA generované správcem front byly použity bez ohledu na to, zda byla zpráva při generování sestavy vložena do správce front, který byl použit. Například, pokud byla zpráva vložena přijímajícím agentem MCA pomocí identifikátoru uživatele MCA, umístí správce front zprávu COA pomocí identifikátoru uživatele MCA.

   Aplikace generující sestavy by normálně měly používat stejné oprávnění jako ty, které se použily při generování odpovědi; to by mělo být obvykle oprávnění identifikátoru uživatele v původní zprávě.

   Má-li sestava cestovat do vzdáleného cíle, odesilatelé a příjemci se mohou rozhodnout, zda ji přijmou, stejně jako pro jiné zprávy.
4. Je-li požadována zpráva hlášení s daty, postupujte takto:

   - Zpráva sestavy se vždy vygeneruje s množstvím dat požadovaných odesílatelem původní zprávy. Je-li zpráva zprávy příliš velká pro frontu odpovědí, zpracování popsané dříve se vyskytne; zpráva sestavy není nikdy zkrácena, aby se vešla do fronty odpovědí.
   - Je-li MDFMT původní zprávy FMXQH, data obsažená v sestavě nezahrnují MQXQH. Data sestavy začínají prvním bajtem dat nad rámec MQXQH v původní zprávě. Dochází k tomu, zda je fronta přenosovou frontou.
5. Je-li ve frontě odpovědí přijata zpráva COA, COD nebo Zpráva o vypršení platnosti, je zaručeno, že byla doručena původní zpráva, byla doručena nebo vypršela její platnost, podle situace. Je-li však jedna nebo více z těchto zpráv sestav požadováno a není obdržená, nelze předpokládat, že by se mohlo jednat o jednu z následujících možností:

   a. Zpráva sestavy je zadržena, protože odkaz je mimo provoz.
   b. Zpráva sestavy je zadržena, protože blokující podmínka existuje ve střední přenosové frontě nebo ve frontě odpovědí (například plná nebo zablokovaná fronta pro vložení)).
   c. Zpráva sestavy se nachází ve frontě nedoručených zpráv.
   d. Když se správce front pokusil vygenerovat zprávu sestavy, nemohl ji zařadit do příslušné fronty a nemohl ji zařadit do fronty nedoručených zpráv, takže zprávu sestavy nelze vygenerovat.
   e. Došlo k selhání správce front mezi hlášenou akcí (přijetí, doručení nebo vypršení platnosti) a generováním odpovídající zprávy sestavy. (To se nestane pro zprávy COD, pokud aplikace

načte původní zprávu v rámci pracovní jednotky, protože zpráva hlášení COD je generována v rámci stejné pracovní jednotky.)

Zprávy o výjimkách mohou být uchovávány stejným způsobem z důvodů 1, 2 a 3 dříve. Pokud však program MCA nemůže generovat zprávu s hlášením o výjimce (zprávu sestavy nelze vložit do fronty odpovědí nebo do fronty nedoručených zpráv), zůstane původní zpráva v přenosové frontě na odesílateli a kanál je uzavřen. K tomu dojde bez ohledu na to, zda byla zpráva sestavy generována při odesílání nebo na přijímajícím konci kanálu.

6. Je-li původní zpráva dočasně zablokována (výsledkem je generování zprávy o výjimce a původní zpráva byla vložena do fronty nedoručených zpráv), ale blokace je vymazána a aplikace pak přečte původní zprávu z fronty nedoručených zpráv a znovu ji umístí do místa určení, může dojít k následujícím:
   - I když byla vygenerována zpráva o výjimce, bude původní zpráva nakonec úspěšně doručena do místa určení.
   - Pro jednu původní zprávu je vygenerována více než jedna zpráva o výjimce, 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 na téma, které může být nula, jedna nebo více. To je třeba vzít v úvahu při výběru možnosti použití voleb sestavy, protože mnoho zpráv sestav může být generováno jako výsledek.
2. Při vkládání zprávy do tématu může být k dispozici mnoho cílových front, které mají být předány kopie zprávy. Mají-li některé z těchto cílových front problém, jako je například zaplnění fronty, závisí úspěšné dokončení příkazu MQPUT na nastavení NPMSGDLV nebo PMSGDLV (v závislosti na trvání zprávy). Pokud je nastavení takové, že doručení zprávy do cílové fronty musí být úspěšné (například, že se jedná o trvalou zprávu na trvalém odběrateli a PMSGDLV je nastaveno na ALL nebo ALLDUR), pak je úspěch definován jako jedno z následujících kritérií:
   - Úspěšné vložení do fronty odběratele
   - Použití parametru RODLQ a úspěšného vložení do fronty nedoručených zpráv, pokud fronta odběratele nemůže převzít zprávu.
   - Použití RODISC, pokud fronta odběratele nemůže převzít zprávu.

Hlásit zprávy pro segmenty zpráv:

1. Zprávy sestavy mohou být požadovány pro zprávy, které mají povolenou segmentaci (viz popis parametru MFSEGA). Pokud správce front zjistí, že je nutné zprávu segmentovat, může být vygenerována zpráva sestavy pro každý z segmentů, který následně zjistí příslušnou podmínku. Aplikace by proto měly být připraveny přijímat více zpráv sestav pro každý typ požadované zprávy sestavy. Pole `MDGID` ve zprávě sestavy může být použito ke korelaci více sestav se identifikátorem skupiny původní zprávy a pole `MDFB` , které se používá k identifikaci typu každé zprávy sestavy.
2. Pokud se GMLOGO používá k načítání zpráv sestav pro segmenty, buďte si vědomi toho, že sestavy různých typů mohou být vráceny po sobě jdoucími voláními MQGET. Je-li například požadována zpráva COA i COD pro zprávu segmentovanou správcem front, mohou zprávy COA a COD vracet zprávy hlášení 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 vztahující se k původní zprávě a druhé volání MQGET by mohlo znovu sestavit všechny zprávy COD. Který je znovu sestavený jako první závisí na tom, jaký typ zprávy hlášení se bude dít první ve frontě.
3. Aplikace, které samy umístí segmenty, mohou uvádět různé volby sestavy pro každý segment. Je však třeba poznamenat následující skutečnosti:
   - Pokud jsou segmenty načítány pomocí volby GMCMPM, budou správcem front uznány 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 ne, nebude možné použít volbu GMCMPM k načtení zpráv sestavy

s jedním voláním MQGET, nebo použít volbu GMASGA pro zjištění, zda byly obdrženy všechny zprávy sestavy.

4. V síti MQ je možné, aby správci front měli různé schopnosti. Je-li zpráva sestavy pro segment generována správcem front nebo agentem MCA, který nepodporuje segmentaci, správce front nebo MCA standardně nebude obsahovat nezbytné informace o segmentech ve zprávě sestavy a může být obtížné identifikovat původní zprávu, která způsobila vygenerování sestavy. Této složitosti se lze vyhnout tak, že požádáte o údaje se zprávou o zprávě, to znamená uvedením vhodných možností RO* D nebo RO* F. Uvědomte si však, že je-li zadána hodnota RO* D, může být vráceno méně než 100 bajtů dat aplikace zprávy do aplikace, která načte zprávu sestavy, pokud je zpráva sestavy generována správcem front nebo agentem MCA, který nepodporuje segmentaci.

Obsah deskriptoru zpráv pro zprávu sestavy: Pokud správce front nebo agent kanálu zpráv (MCA) vygeneruje zprávu s hlášením, nastaví pole v deskriptoru zpráv na následující hodnoty a poté vloží zprávu normálním způsobem.

Tabulka 709. Hodnoty použité pro pole MQMD, je-li generována zpráva sestavy

Pole v MQMD	Použitá hodnota
MDSID	MDSIDROVSKÁ
MDVER	MDVER2
MDREP	RONAN
MDMT	MTRPRT
MDEXP	EIULIM
MDFB	Podle potřeby pro povahu zprávy (FBSCOA, FBCOD, FBEXP nebo hodnota 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 uvedeno ve volbách sestavy v původním deskriptoru zpráv
MDCID	Jak je uvedeno ve volbách sestavy v původním deskriptoru zpráv
MDBOC	0
MDRQ	Mezery
MDRM	Název správce front
MDUID	Jako sada podle volby PMPASI
MDACC	Jako sada podle volby PMPASI
MDAID	Jako sada podle volby PMPASI
MDPAT	ATQM nebo odpovídající pro agenta kanálu zpráv
MDPAN	Prvních 28 bajtů názvu správce front nebo názvu agenta kanálu zpráv. Pro zprávy sestav generované mostem IMS toto pole obsahuje název skupiny XCF a název člena XCF systému IMS , kterého se zpráva týká.
MDPD	Datum, kdy se odešle zpráva hlášení
MDPT	Čas odeslání zprávy sestavy
MDAOD	Mezery
MDGID	Zkopírováno z původního deskriptoru zprávy

Tabulka 709. Hodnoty použité pro pole MQMD, je-li generována zpráva sestavy (pokračování)

Pole v MQMD	Použitá hodnota
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 nastaví se na délku dat původní zprávy jinak

Aplikace generující sestavu je doporučována pro nastavení podobných hodnot, s výjimkou následujících:

- Pole MDRM může být nastaveno na prázdné místo (správce front to změní na název lokálního správce front, když je zpráva vložena).
- Pole kontextu by měla být nastavena pomocí volby, která by byla použita pro odpověď, obvykle PMPASI.

Analýza pole sestavy: Pole MDREP obsahuje podpole; z tohoto důvodu aplikace, které potřebují zkontrolovat, zda odesilatel zprávy vyžádal určitou sestavu, by měl použít jednu z technik popsaných v "Analýza pole sestavy v systému IBM i" na stránce 1419.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je RONONE.

MDRM (48-bajtový znakový řetězec)

Název správce front odpovědí.

Jedná se o název správce front, do kterého má být odeslána zpráva odpovědi nebo zpráva sestavy. MDRQ je lokální název fronty, která je definovaná na tomto správci front.

Je-li pole MDRM prázdné, správce lokální fronty vyhledá ve svých definicích front název **MDRQ** . Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota **MDRM** v přenesené zprávě je nahrazena hodnotou atributu **RemoteQMgrName** z definice vzdálené fronty a tato hodnota bude vrácena v deskriptoru zpráv, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, MDRM přenášený se zprávou je název lokálního správce front.

Je-li jméno uvedeno, může obsahovat koncové mezery; první znak null a znaky za ním jsou považovány za mezery. Jinak se však nekontroluje, zda název odpovídá pravidlům pojmenování pro správce front, nebo že tento název je známý odesílajícímu správci front; pro předaný název je to také pravda, pokud je **MDRM** nahrazeno v přenesené zprávě.

Není-li vyžadována odpověď na frontu, doporučuje se (i když toto není zaškrtnuto), že pole MDRM by mělo být nastaveno na mezery; pole by nemělo být ponecháno neinicializováno.

U volání MQGET správce front vždy vrátí název doplněný 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 LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

MDRQ (48-bajtový znakový řetězec)

Název fronty odpovědí.

Toto je jméno fronty zpráv, do které by měla aplikace, která vydala požadavek na získání pro zprávu, poslat zprávy MTRPLY a MTRPRT. Název je lokální název fronty, který je definován ve správci front identifikovaném příkazem MDRM. Tato fronta by neměla být modelovou frontou, ačkoli odesílající správce front toto neověří, když je zpráva vložena.

U volání MQPUT a MQPUT1 nesmí být toto pole prázdné, pokud má pole MDMT hodnotu MTRQST, nebo pokud pole MDREP vyžaduje nějaké zprávy sestavy. Zadaná hodnota (nebo náhrada) se však předává aplikaci, která vydala požadavek na získání pro zprávu, bez ohledu na typ zprávy.

Je-li pole MDRM prázdné, vyhledá lokální správce front název MDRQ ve svých vlastních definicích front. Pokud existuje lokální definice vzdálené fronty s tímto názvem, hodnota MDRQ v přenesené zprávě je nahrazena hodnotou atributu **RemoteQName** z definice vzdálené fronty a tato hodnota bude vrácena v deskriptoru zpráv, když přijímající aplikace vydá pro zprávu volání MQGET. Pokud lokální definice vzdálené fronty neexistuje, MDRQ se nemění.

Je-li jméno uvedeno, může obsahovat koncové mezery; první znak null a znaky za ním jsou považovány za mezery. Jinak se však nekontroluje, zda název splňuje pravidla pojmenování pro fronty; to je také pravda pro přenesený název, pokud je MDRQ nahrazen v přenesené zprávě. Jediná kontrola je, že jméno bylo uvedeno, pokud to okolnosti vyžadují.

Není-li vyžadována odpověď na frontu, doporučuje se (i když toto není zaškrtnuto), že pole MDRQ by mělo být nastaveno na mezery; pole by nemělo být ponecháno neinicializováno.

U volání MQGET správce front vždy vrátí název doplněný mezerami na délku pole.

Pokud nelze doručit zprávu, která vyžaduje zprávu sestavy, a zpráva sestavy také nemůže být doručena do zadané fronty, původní zpráva i zpráva sestavy jdou do fronty nedoručených zpráv (undelivered-message). Viz atribut **DeadLetterQName** popsaný v tématu "Atributy pro správce front v systému IBM i" na stránce 1384.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

MDSEQ (10ciferné celé číslo se znaménkem)

Pořadové číslo logické zprávy ve skupině.

Pořadová čísla začínají hodnotou 1 a u každé nové logické zprávy ve skupině se zvyšují o 1 až do maximální hodnoty 999 999 999. Fyzická zpráva, která není ve skupině, má pořadové číslo 1.

Toto pole nemusí být nastaveno aplikací v rámci volání MQPUT nebo MQGET, pokud:

- Na volání MQPUT je uveden PMLOGO.
- Na základě volání MQGET není uvedena hodnota 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 produkt MDSEQ nastaven na příslušnou hodnotu.

Ve vstupu do volání MQPUT a MQPUT1 používá správce front hodnotu popsanou v tabulce Tabulka 1. Na výstupu z volání MQPUT a MQPUT1 nastaví správce front toto pole na hodnotu, která byla odeslána spolu se zprávou.

Na vstupu do volání MQGET používá správce front hodnotu popsanou v tabulce Tabulka 1. Na výstupu z volání MQGET nastaví správce front toto pole na hodnotu pro načtenou zprávu.

Počáteční hodnota tohoto pole je jedna. Toto pole je ignorováno, pokud MDVER je menší než MDVER2.

MDSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

MDSIDROVSKÁ
Identifikátor pro strukturu deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MDSIDV.

MDUID (12bajtový znakový řetězec)

Identifikátor uživatele.

Tato část je součástí *kontextu identity* zprávy. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

MDUID uvádí identifikátor uživatele aplikace, která je původcem zprávy. Správce front považuje tyto informace za znaková data, ale nedefinuje její formát.

Po přijetí zprávy lze produkt MDUID použít v poli ODAU parametru **OBJDSC** u následných volání MQOPEN nebo MQPUT1 , takže kontrola autorizace bude provedena pro uživatele produktu MDUID namísto toho, že aplikace bude otevřená.

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 následující:
  - Pro dávku, identifikátor uživatele z karty JES JOB nebo spuštěnou úlohu
  - Pro TSO, protokol na identifikátoru uživatele
  - Pro CICSje identifikátor uživatele přidružený k úloze
  - Pro produkt IMSzávisí identifikátor uživatele na typu aplikace:
    - Počet:
      - Regiony BMP bez zpráv
      - Nezpráva IFP regionů
      - Zpráva BMP a zprávy IFP zprávy, které nevydaly úspěšné volání GU

      správce front používá identifikátor uživatele z karty JES JOB nebo z identifikátoru uživatele TSO. Jsou-li tyto hodnoty prázdné nebo mají hodnotu null, použije název bloku specifikace programu (PSB).
    - Počet:
      - Zpráva BMP a zprávy IFP zprávy, které vydaly úspěšné volání GU
      - Oblasti MPP

      správce front používá jednu z následujících možností:
      - Identifikátor přihlášeného uživatele přidružený ke zprávě
      - Název logického terminálu (LTERM)
      - Identifikátor uživatele z karty JES JOB
      - Identifikátor uživatele TSO
      - Název PSB
- IBM i V systému IBM isprávce front používá název profilu uživatele přidruženého k úloze aplikace.
- Linux AIX V systému AIX and Linuxpoužívá správce front následující:
  - Přihlašovací jméno aplikace
  - Efektivní identifikátor uživatele procesu, pokud není k dispozici přihlášení
  - Identifikátor uživatele přidružený k transakci, pokud je aplikací transakce CICS .
- V systému VSE/ESAse jedná o vyhrazené pole.
- Windows V systému Windowssprávce front používá prvních 12 znaků jména přihlášeného uživatele.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, pokud je položka PMSETI nebo PMSETA zadána v parametru **PMO** . Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny na mezery. Není-li položka PMSETI nebo PMSETA uvedena, je toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat MDUID , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota MDUID , která je uchována se zprávou, pokud je uchována (viz popis PMRET pro více podrobností o zachovaných

publikacích), ale nepoužívá se jako MDUID , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu pro přepis MDUID ve všech publikačních publikacích, které se na ně posílají. Pokud zpráva nemá žádný kontext, pole je zcela prázdné.

Toto je výstupní pole pro volání MQGET. Délka tohoto pole je dána LNUID. Počáteční hodnota tohoto pole je 12 prázdných znaků.

MDVER (10číslicové celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

MDVER1
Struktura deskriptoru zpráv Version-1 .

MDVER2
Struktura deskriptoru zpráv Version-2 .

Poznámka: Při použití version-2 MQMD provádí správce front 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 poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

MDVERC
Aktuální verze struktury deskriptoru zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MDVER1.

Počáteční hodnoty

Tabulka 710. Počáteční hodnoty polí v produktu MQMD

Název pole	Název konstanty	Hodnota konstanty
MDSID	MDSIDROVSKÁ	'MD¬¬'
MDVER	MDVER1	1
MDREP	RONAN	0
MDMT	MTDGRM	8
MDEXP	EIULIM	-1
MDFB	FBNONE	0
MDENC	ENNAT	Závisí na prostředí
MDCSI	CSQM	0
MDFMT	FMNONE	Mezery
MDPRI	DEFINICE PRQDEF	-1
MDPER	DEFINICE PEQDEF	2
MDMID	MINON	Hodnoty null
MDCID	CINNE	Hodnoty null
MDBOC	Není	0
MDRQ	Není	Mezery
MDRM	Není	Mezery

Tabulka 710. Počáteční hodnoty polí v produktu MQMD (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
MDUID	Není	Mezery
MDACC	ANONE	Hodnoty null
MDAID	Není	Mezery
MDPAT	ATNCON	0
MDPAN	Není	Mezery
MDPD	Není	Mezery
MDPT	Není	Mezery
MDAOD	Není	Mezery
MDGID	GINON	Hodnoty null
MDSEQ	Není	1
MDOFF	Není	0
MDMFL	MFNONE	0
MDOLN	OLUNDFE.	-1
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                                     0000000000000000000000-
     D                                     0000000000000000000000-
     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                                     0000000000000000000000-
     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 taková pole MQMD, která existují v version-2 MQMD, ale ne v version-1 MQMD.

Název formátu: FMMDE.

Znaková sada a kódování: Data v MQMDE musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT pro programovací jazyk C.

Znaková sada a kódování MQMDE musí být nastaveny na pole *MDCSI* a *MDENC* v:

- MQMD (je-li struktura MQMDE na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQMDE (všechny ostatní případy).

Pokud se MQMDE nenachází ve znakové sadě a kódování správce front, je MQMDE přijat, ale není dodržen, to znamená, že MQMDE je považován za data zprávy.

Použití: Normální aplikace by měly používat version-2 MQMD, v takovém případě se nezobrazí ve struktuře MQMDE. Avšak specializované aplikace a aplikace, které i nadále používají version-1 MQMD, se mohou v některých situacích setkat s MQMDE. Struktura MQMDE se může vyskytnout za následujících okolností:

- Určeno na základě volání MQPUT a MQPUT1
- Vráceno voláním MQGET
- Ve zprávách v přenosových frontách

- "Hodnota MQMDE zadaná v rámci volání MQPUT a MQPUT1" na stránce 1142
- "MQMDE vrácený voláním MQGET" na stránce 1142
- "MQMDE ve zprávách v přenosových frontách" na stránce 1143
- "Pole" na stránce 1143

- “Počáteční hodnoty” na stránce 1145
- “Deklarace RPG” na stránce 1145

Hodnota MQMDE zadaná v rámci volání MQPUT a MQPUT1

V případě volání MQPUT a MQPUT1 , pokud aplikace poskytuje version-1 MQMD, může aplikace volitelně připojit data zprávy k datům MQMDE a nastavit pole *MDFMT* v MQMD na hodnotu FMMDE tak, aby označovalo, že je přítomen MQMDE. Pokud aplikace neposkytuje prostředí MQMDE, předpokládá správce front výchozí hodnoty pro pole v MQMDE. Výchozí hodnoty, které správce front používá, jsou stejné jako počáteční hodnoty pro strukturu-viz Tabulka 712 na stránce 1145.

Pokud aplikace poskytuje version-2 MQMD *and* předpony dat zprávy aplikace s MQMDE, struktury se zpracují tak, jak jsou zobrazeny v Tabulka 711 na stránce 1142.

Tabulka 711. Akce správce front, je-li hodnota MQMDE zadána v MQPUT nebo MQPUT1

Verze MQMD	Hodnoty polí version-2	Hodnoty odpovídajících polí v MQMDE	Akce provedená správcem front
1	-	Platný	MQMDE je poctěn
2	Výchozí	Platný	MQMDE je poctěn
2	Není výchozí	Platný	MQMDE je považován za data zprávy
1 nebo 2	Libovolný	Neplatný	Volání selže s příslušným kódem příčiny
1 nebo 2	Libovolný	MQMDE je v nesprávné znakové sadě nebo kódování, nebo se jedná o nepodporovanou verzi	MQMDE je považován za data zprávy

Je tu jeden speciální případ. Pokud aplikace používá version-2 MQMD k vložení zprávy, která je segmentem (tj. je nastaven příznak MFSEG nebo MFLSEG) a název formátu v MQMD je FMDLH, správce front vygeneruje strukturu MQMDE a vloží ji *mezi* strukturou MQDLH a daty, která za ní následují.
V deskriptoru MQMD, který správce front zachovává se zprávou, jsou pole version-2 nastavena na jejich výchozí hodnoty.

Několik polí, která existují ve version-2 MQMD, ale ne version-1 MQMD jsou vstupní/výstupní pole MQPUT a MQPUT1. Správce front však nevrátí žádné hodnoty v ekvivalentních polích v MQMDE na výstupu z volání MQPUT a MQPUT1 ; pokud aplikace vyžaduje tyto výstupní hodnoty, musí použít version-2 MQMD.

MQMDE vrácený voláním MQGET

Pokud v rámci volání MQGET poskytuje aplikace MQMD version-1 , předpony správce front vrátí zprávu s hodnotou MQMDE, ale pouze v případě, že jedno nebo více polí v MQMDE má nevýchozí hodnotu. Správce front nastaví pole *MDFMT* v MQMD na hodnotu FMMDE, aby indikovala, že je přítomen prvek MQMDE.

Pokud aplikace poskytuje prostředí MQMDE na začátku parametru **BUFFER** , hodnota MQMDE se ignoruje. Při návratu z volání MQGET je tato zpráva nahrazena hodnotou MQMDE pro zprávu (je-li vyžadována) nebo je přepsána daty zprávy aplikace (pokud není MQMDE potřeba).

Pokud je MQMDE vrácen voláním MQGET, jsou data v MQMDE obvykle ve znakové sadě a kódování správce front. Nicméně MQMDE může být v nějaké jiné znakové sadě a kódování, pokud:

- Objekt MQMDE byl zpracován jako data na volání MQPUT nebo MQPUT1 (viz Tabulka 711 na stránce 1142 , kde jsou uvedeny okolnosti, které mohou být příčinou).
- Zpráva byla přijata ze vzdáleného správce front připojeného pomocí připojení TCP a přijímací agent kanálu zpráv (MCA) nebyl správně nastaven (další informace naleznete v tématu Zabezpečení objektů IBM MQ for IBM i).

MQMDE ve zprávách v přenosových frontách

Zprávy v přenosových frontách mají předponu struktury MQXQH, která obsahuje v něm version-1 MQMD. Objekt MQMDE může být také přítomen, umístěn mezi strukturou MQXQH a daty zprávy aplikace, ale obvykle se bude prezentovat pouze tehdy, pokud jedno nebo více polí v MQMDE má nevýchozí hodnotu.

Další struktury záhlaví IBM MQ se mohou také vyskytnout mezi strukturou MQXQH a daty zprávy aplikace. Je-li například přítomen záhlaví dead-letter MQDLH a zpráva není segmentem, objednávka je následující:

- MQXQH (obsahující version-1 MQMD)
- MQMDE
- MQDLH
- Data zprávy aplikace

Pole

Struktura MQMDE obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

MECSI (10číslicové podepsané celé číslo)

Identifikátor znakové sady dat, který následuje MQMDE.

Uvádí identifikátor znakové sady dat, která se řídí strukturou MQMDE; nevztahuje se na znaková data v samotné struktuře MQMDE.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Správce front nekontroluje, zda je toto pole platné. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

MEENC (10ciferné celé číslo se znaménkem)
MEENC (10ciferné celé číslo se znaménkem)

Uvádí číselné kódování dat, která se řídí strukturou MQMDE; nevztahuje se na číselná data ve struktuře MQMDE.

Na základě 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 najdete v poli *MDENC* , které popisuje téma “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099 .

Počáteční hodnota tohoto pole je ENNAT.

MEGFLG (10ciferné celé číslo se znaménkem)

Obecné příznaky.

Lze zadat následující příznak:

MEFNON
Žádné vlajky.

Počáteční hodnota tohoto pole je MEFNON.

MEMFMT (8bajtový znakový řetězec)

Název formátu dat, která následuje MQMDE.

Uvádí název formátu dat, která se řídí strukturou MQMDE.

Na základě 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 (Message Descriptor) na serveru IBM i” na stránce 1099 .

Počáteční hodnota tohoto pole je FMNONE.

MEGID (24bajtový bitový řetězec)

Identifikátor skupiny.

Viz pole *MDGID* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Počáteční hodnota tohoto pole je GINONE.

MELEN (10ciferné 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.

MEMFG (10ciferné celé číslo se znaménkem)

Příznaky zpráv.

Viz pole *MDMFL* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Počáteční hodnota tohoto pole je MFNONE.

MEGP (10ciferné celé číslo se znaménkem)

Posunutí dat ve fyzické zprávě od začátku logické zprávy.

Viz pole *MDOFF* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Počáteční hodnota tohoto pole je 0.

MEOLN (10ciferné celé číslo se znaménkem)

Délka původní zprávy.

Viz pole *MDOLN* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Počáteční hodnota tohoto pole je OLUNDF.

MESEQ (10ciferné celé číslo se znaménkem)

Pořadové číslo logické zprávy ve skupině.

Viz pole *MDSEQ* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Počáteční hodnota tohoto pole je 1.

MESID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

MESIDV
Identifikátor pro strukturu rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MESIDV.

MEVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

MEVER2
Struktura rozšíření deskriptoru zpráv Version-2 .

Následující konstanta uvádí číslo verze aktuální verze:

MEVERC
Aktuální verze struktury rozšíření deskriptoru zpráv.

Počáteční hodnota tohoto pole je MEVER2.

Počáteční hodnoty

Tabulka 712. Počáteční hodnoty polí v 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	GINON	Hodnoty null
MESEQ	Není	1
MEOFF	Není	0
MEMFL	MFNONE	0
MEOLN	OLUNDFE.	-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                                     00000000000000000000-
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 (zpracování zpráv pro volby vyrovnávací paměti) v systému IBM i

Struktura definující popisovač zprávy pro volby vyrovnávací paměti

Přehled

Účel: Struktura MQMHBO umožňuje aplikacím zadávat volby, které řídí způsob, jakým jsou vyrovnávací paměti vytvářeny z manipulátorů zpráv. Struktura je vstupním parametrem na volání MQMHBUF.

Znaková sada a kódování: Data v objektu MQMHBO musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1146
- "Počáteční hodnoty" na stránce 1147
- "Deklarace RPG" na stránce 1147

Pole

Struktura MQMHBO obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

MBOPT (10ciferné celé číslo se znaménkem)

Struktura voleb pro strukturu voleb vyrovnávací paměti-pole MBOPT.

Tyto volby řídí akci MQMHBUF.

Je třeba určit následující volbu:

MBPRRF
Při převádění vlastností z manipulátorů zpráv do vyrovnávací paměti je převeďte do formátu MQRFH2 .

Volitelně můžete také zadat následující volbu. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

MBDLPR
Vlastnosti, které jsou přidány do vyrovnávací paměti, se odstraní z popisovače zprávy. Pokud se nezdaří volání, nebudou odstraněny žádné vlastnosti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MBPRRF.

MBSID (10ciferné celé číslo se znaménkem)

Struktura ID voleb vyrovnávací paměti-pole MBSID.

Jedná se o identifikátor struktury. Hodnota musí být:

MBSIDV
Identifikátor pro popisovač zprávy pro strukturu voleb vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole isMBSIDV.

MBVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury. Hodnota musí být:

MBVER1
Číslo verze pro popisovač zprávy pro strukturu voleb vyrovnávací paměti.

Následující konstanta uvádí číslo verze aktuální verze:

MBVERC
Aktuální verze obslužné rutiny zpráv pro strukturu voleb vyrovnávací paměti.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MBVER1.

Počáteční hodnoty

Tabulka 713. Počáteční hodnoty polí v MQMHBO		
Název pole	**Název konstanty**	**Hodnota konstanty**
MVSID	MBSIDV	'MHBO'
MBVER	MBVER1	1
MBOPT	MBPRRF	

Notes:

1. Hodnota null string nebo mezery označuje 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 1148
- "Počáteční hodnoty" na stránce 1155
- "Deklarace RPG" na stránce 1155

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 1152).

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 714. Počáteční hodnoty polí v MQOD

Název pole	Název konstanty	Hodnota konstanty
ODSID	ODSIDV	'OD¬¬'
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                                     000000000000000000000000-
     D                                     000000000000000000000000-
     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 (Object record) v systému IBM i

Struktura MQOR se používá k určení názvu fronty a názvu správce front jedné cílové fronty.

Přehled

Účel: MQOR je vstupní struktura pro volání MQOPEN a MQPUT1 .

Znaková sada a kódování: Data v MQOR musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur v rámci volání MQOPEN je možné otevřít seznam front; tento seznam se nazývá *distribuční seznam*. Pokud byla fronta úspěšně otevřena, každá zpráva používající obslužnou rutinu fronty vrácenou tímto voláním MQOPEN je umístěna do každé z front v seznamu.

- "Pole" na stránce 1157
- "Počáteční hodnoty" na stránce 1158
- "Deklarace RPG" na stránce 1158

Pole

Struktura MQOR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

ORMN (48-bajtový znakový řetězec)

Název správce front objektu.

To 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 (48-bajtový znakový řetězec)

Název objektu.

To je stejné jako pole *ODON* ve struktuře MQOD (podrobnosti viz MQOD), kromě následujících:

- Musí se jednat o 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 715. Počáteční hodnoty polí 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

Prostor **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 se musí nacházet ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1158
- "Počáteční hodnoty" na stránce 1161
- "Deklarace RPG" na stránce 1161

Pole

Struktura MQPD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

PDCT (10ciferné celé číslo se znaménkem)

Tato vlastnost popisuje kontext zprávy, do níž daná vlastnost patří.

Obdrží-li správce front zprávu obsahující vlastnost definované produktem IBM MQ, kterou správce front rozpozná jako nesprávnou. Správce front opraví hodnotu pole *PDCT* .

Je možné zadat následující volbu:

PRASC
Vlastnost je přidružena ke kontextu uživatele.

K nastavení vlastnosti přidružené k kontextu uživatele pomocí volání MQSETMP není vyžadována žádná speciální autorizace.

Ve správci front produktu IBM WebSphere MQ 7.0 je vlastnost přidružená ke kontextu uživatele uložena, jak je popsáno pro OOSAVA. Volání MQPUT s uvedeným PMASA způsobí, že se vlastnost zkopíruje z uloženého kontextu do nové zprávy.

Není-li dříve popsaná volba vyžadována, 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 z volání MQINQMP. Počáteční hodnota tohoto pole je PDNOC.

PDCPYOPT (10ciferné celé číslo se znaménkem)

Popisuje, do kterého typu zpráv má být vlastnost zkopírována.

Toto je pouze výstupní pole pro rozeznávané vlastnosti definované IBM MQ; IBM MQ nastavuje příslušnou hodnotu.

Obdrží-li správce front zprávu obsahující vlastnost definované produktem IBM MQ, kterou správce front rozpozná jako nesprávnou. Správce front opraví hodnotu pole *CopyOptions* .

Můžete uvést jednu nebo více z těchto voleb. Chcete-li zadat více než jednu volbu, buď přidejte hodnoty dohromady (nepřidávejte stejnou konstantu víckrát než jednou), nebo zkombinujte hodnoty pomocí bitové operace OR (pokud programovací jazyk podporuje bitové operace).

KOPFOR
Tato vlastnost je zkopírována do předávané zprávy.

COPPUB
Tato vlastnost se okopíruje do zprávy přijaté odběratelem při publikování zprávy.

KOPREP
Tato vlastnost je zkopírována do zprávy odpovědi.

KOPRP
Tato vlastnost je zkopírována do zprávy sestavy.

KOPL.
Tato vlastnost se zkopíruje do všech typů následujících zpráv.

COPNON
Tato vlastnost se nekopíruje do zprávy.

Výchozí volba: Pro dodání výchozí sady voleb kopírování lze zadat následující volbu:

KOPEDEF
Tato vlastnost se okopíruje do zprávy, která se předá, do zprávy sestavy nebo do zprávy přijaté odběratelem při publikování zprávy.

To je stejné jako uvedení kombinace voleb COPFOR, plus COPRP plus COPPUB.

Pokud žádná z výše popsaných voleb není povinná, použijte následující volbu:

COPNON
Použijte tuto hodnotu, chcete-li označit, že nebyly zadány žádné jiné volby kopírování; mezi touto vlastností a následujícími zprávami však neexistuje žádná relace. Tato hodnota je vždy vrácena pro vlastnosti deskriptoru zpráv.

Jedná se o vstupní/výstupní pole pro volání MQSETMP a výstupní pole z volání MQINQMP. Počáteční hodnota tohoto pole je COPDEF.

PDOPT (10ciferné celé číslo se znaménkem)

Hodnota musí být:

PDNONE
Nejsou zadány žádné volby

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PDNONE.

PDSID (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor struktury; hodnota musí být:

PSIDV
Identifikátor pro strukturu deskriptoru vlastností.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **PSIDV**.

PDSUP (10ciferné celé číslo se znaménkem)

Toto pole popisuje, jaká úroveň podpory pro vlastnost zprávy je vyžadována správce front, aby byla zpráva obsahující tuto vlastnost vložena do fronty. Toto platí pouze pro vlastnosti definované IBM MQ; podpora pro všechny ostatní vlastnosti je volitelná.

Pole je automaticky nastaveno na správnou hodnotu, je-li správce front znám vlastnost definovaná uživatelem IBM MQ. Není-li vlastnost rozpoznána, je přiřazena PDSUPO. Obdrží-li správce front zprávu obsahující vlastnost definované produktem IBM MQ, kterou správce front rozpozná jako nesprávnou. Správce front opraví hodnotu pole *PDSUP* .

Při nastavení definované vlastnosti IBM MQpomocí volání MQSETMP na obslužné rutiny zprávy, kde byla nastavena volba CMNOVA, se *PDSUP* stane vstupním polem. To umožňuje aplikaci umístit vlastnost definované IBM MQs správnou hodnotou, kde tato vlastnost není podporována připojeným správcem front, ale kde je zpráva určena ke zpracování v jiném správci front.

Hodnota PDSUPO je vždy přidružena k vlastnostem, které nejsou definované IBM MQvlastností.

Pokud správce front produktu IBM WebSphere MQ 7.0 , který podporuje vlastnosti zprávy, obdrží vlastnost obsahující nerozpoznanou hodnotu *PDSUP* , bude s touto vlastností zacházeno jako s následujícím způsobem:

- PDSUPR bylo uvedeno, pokud byla některá z nerozpoznaných hodnot obsažena v PDRUM.
- PDSUPL byl zadán, pokud jsou některé z nerozpoznaných hodnot obsaženy v PDAUXM
- Hodnota PDSUPO byla zadána jinak.

Je vrácena jedna z následujících hodnot volání MQINQMP nebo jedna z hodnot může být zadána při použití volání MQSETMP na obslužné rutiny zprávy, kde je nastavena volba CMNAVA:

PDSUPO
Vlastnost je přijata správcem front, i když není podporována. Vlastnost může být vyřazena, aby byla zpráva přetékat do správce front, který nepodporuje vlastnosti zprávy. Tato hodnota je také přiřazena vlastnostem, které nejsou IBM MQdefinované.

PDSUPR
Podpora pro vlastnost je povinná. Zpráva je odmítnuta správcem front, který nepodporuje vlastnost definovaná uživatelem IBM MQ. Volání MQPUT nebo MQPUT1 selže s kódem dokončení CCFAIL a kódem příčiny RC2490.

PDSUPL
Zpráva je odmítnuta správcem front, který nepodporuje vlastnost definovaná uživatelem IBM MQ, je-li zpráva určena pro lokální frontu. Volání MQPUT nebo MQPUT1 selže s kódem dokončení CCFAIL a kódem příčiny RC2490.

Volání MQPUT nebo MQPUT1 je úspěšné, pokud je zpráva určena pro vzdáleného správce front.

Toto je výstupní pole v rámci volání MQINQMP a vstupní pole pro volání MQSETMP, pokud byl popisovač zprávy vytvořen s nastavenou volbou CMNOVA. Počáteční hodnota tohoto pole je PDSUPO.

PDVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

PDVER1
Struktura deskriptoru vlastností Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

PDVERC
Aktuální verze struktury deskriptoru vlastností.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **PDVER1**.

Počáteční hodnoty

Tabulka 716. Počáteční hodnoty polí v MQPD		
Název pole	**Název konstanty**	**Hodnota konstanty**
PDSID	PDSIDV	'PD'
PDVER	PDVER1	1
PDOPT	PDNONE	0
PDSUP	PDSUPO	0
PDCT	PDNOC	0
PDCPYOPT	KOPEDEF	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 (volby vkládání zpráv) v systému IBM i

Struktura MQPMO umožňuje aplikaci určit volby, které řídí způsob vkládání zpráv do front nebo publikování do témat.

Přehled

Účel

Struktura je vstupním/výstupním parametrem na volání MQPUT a MQPUT1 .

Verze

Aktuální verze MQPMO je PMVER2. Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech, které následují.

Poskytnutý soubor COPY obsahuje nejnovější verzi MQPMO, která je podporována prostředím, ale s počáteční hodnotou pole *PMVER* nastavenou na PMVER1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí nastavit pole *PMVER* na číslo verze požadované verze.

Znaková sada a kódování

Data v MQPMO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

- "Pole" na stránce 1161
- "Počáteční hodnoty" na stránce 1175
- "Deklarace RPG" na stránce 1175

Pole

Struktura MQPMO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

PMCT (10ciferné celé číslo se znaménkem)

Popisovač objektu vstupní fronty.

Je-li uvedeno PMPASI nebo PPMASA, musí toto pole obsahovat popisovač vstupní fronty, z níž jsou informace o kontextu přidružené ke zprávě, která má být vložena.

Nejsou-li PMPASI a PMPASA zadány, bude toto pole ignorováno.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

PMIDC (Celé číslo se znaménkem 10 číslic)

Počet zpráv, které nebylo možné odeslat.

Jedná se o počet zpráv, které nebylo možné odeslat do front v seznamu distribuce. Tento počet zahrnuje fronty, které se nepodařilo otevřít, a fronty, které byly úspěšně otevřeny, ale pro které došlo k selhání operace vložení. Toto pole je také nastaveno při vkládání zprávy do jedné fronty, která není v rozdělovníku.

Poznámka: Toto pole je nastaveno pouze v případě, že je parametr **CMPCOD** u volání MQPUT nebo MQPUT1 nastaven na hodnotu CCOK nebo CCWARN; není nastavena, je-li parametr **CMPCOD** CCFAIL.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole není nastaveno, pokud *PMVER* je menší než PMVER2.

PMKDC (10 číslic se znaménkem celého čísla)

Počet zpráv odeslaných úspěšně do lokálních front.

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v seznamu distribuce, které jsou lokálními frontami. Tento počet nezahrnuje zprávy odeslané do front, které se interpretují do vzdálených front (ačkoli lokální přenosová fronta je na počátku použita 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 *PMVER* je menší než PMVER2.

PMOPT (10ciferné celé číslo se znaménkem)

Volby, které řídí akce MQPUT a MQPUT1.

Může být uvedena jakákoli nebo žádná z následujících možností. Pokud je požadováno více než jedna, lze tyto hodnoty přidat (nepřidávat stejnou konstantu víckrát než jednou). Kombinace, které nejsou platné, jsou zaznamenány; jakékoli jiné kombinace jsou platné.

Volby publikování: Následující volby řídí způsob, jakým jsou zprávy publikovány do tématu.

PMSRTO

Veškeré informace vyplněné v polích MDRQ a MDRM deskriptoru MQMD této publikace nejsou předány odběratelům. Je-li tato volba použita s volbou sestavy, která vyžaduje odpověď ReplyToQ, volání selže s RC2027 .

PMRET

Odeslaná publikování má být uchována správcem front. To umožňuje odběrateli požádat o kopii této publikace po jejím publikování pomocí volání MQSUBRQ. Umožňuje také odeslání publikování aplikacím, které učiní jejich odběr po datu, kdy byla tato publikace vytvořena, pokud se nerozhodnou neodesílat jej pomocí volby SONEWP. Je-li aplikace odeslána publikování, která byla uchována, je označena vlastností zprávy mq.IsRetained této publikace.

V každém uzlu stromu témat může být zachováno pouze jedno publikování. To znamená, že pokud již existuje zachované publikování pro toto téma publikováno jakoukoli jinou aplikací, je tato publikace nahrazena touto publikací. Je proto lepší vyhnout se tomu, aby zprávy uchovaly více než jeden vydavatel v rámci stejného tématu.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr obsahovat zástupný znak v tématu, v takovém případě může být u žádající aplikace zaslán počet zachovaných publikování (v různých uzlech stromu témat) a může být odesláno několik příruček. Další podrobnosti naleznete v popisu volání příkazu "MQSUBRQ-Požadavek na odběr" na stránce 788 .

Je-li tato volba použita a publikování nelze zadržet, zpráva se nepublikuje a volání selže s RC2479 .

Volby synchronizačních bodů: Následující volby se vztahují k účasti volání MQPUT nebo MQPUT1 v rámci transakce:

PMSYP
Vložit zprávu s řízením synchronizačního bodu.

Požadavek má fungovat v rámci běžných protokolů jednotky práce. Zpráva není viditelná mimo pracovní jednotku, dokud se jednotka práce nepotvrdí. Je-li jednotka práce zálohována, zpráva se odstraní.

Pokud tato volba a PMNSYP nejsou uvedeny, požadavek na vložení není v rámci pracovní jednotky.

PMSYP nesmí být zadán společně s PMNSYP.

PMNSYP
Vložit zprávu bez řízení synchronizačního bodu.

Požadavek má fungovat mimo běžné protokoly jednotek práce. Zpráva je okamžitě k dispozici a nelze ji odstranit tím, že zazálohujete jednotku práce.

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 korelačního identifikátoru: Následující volby vyžadují, aby správce front vygeneroval nový identifikátor zprávy nebo identifikátor korelace:

PMNMID
Generujte nový identifikátor zprávy.

Tato volba způsobí, že správce front nahradí obsah pole *MDMID* v produktu MQMD novým identifikátorem zprávy. Tento identifikátor zprávy je odeslán se zprávou a vrácen do aplikace na výstupu z volání MQPUT nebo MQPUT1 .

Tuto volbu lze zadat také v případě, že je zpráva vložena do distribučního seznamu; podrobnosti naleznete v popisu pole *PRMID* ve struktuře MQPMR.

Použití této volby zmírňuje aplikaci nutnosti resetovat pole *MDMID* na INONE před každým voláním MQPUT nebo MQPUT1 .

ID DOKUMENTU
Vygenerujte nový korelační identifikátor.

Tato volba způsobí, že správce front nahradí obsah pole *MDCID* v produktu MQMD novým identifikátorem korelace. Tento korelační identifikátor se odešle se zprávou a vrátí se aplikaci na výstup z volání MQPUT nebo MQPUT1 .

Tuto volbu lze zadat také v případě, že je zpráva vložena do distribučního seznamu; podrobnosti naleznete v popisu pole *PRCID* ve struktuře MQPMR.

PMNCID je užitečné v situacích, kdy aplikace vyžaduje jedinečný korelační identifikátor.

Volby skupiny a segmentu: Následující volba souvisí se zpracováním zpráv ve skupinách a segmentech logických zpráv. Tyto definice mohou být užitečné při seznámení s volbou:

Fyzická zpráva
Jedná se o nejmenší jednotku informací, které lze umístit do fronty nebo z ní odebrat; často odpovídá informacím zadaným nebo načteným při volání MQPUT, MQPUT1nebo MQGET. Každá fyzická zpráva má svůj vlastní deskriptor zprávy (MQMD). Obecně jsou fyzické zprávy rozlišeny

odlišnými hodnotami identifikátoru zprávy (pole*MDMID* v MQMD), ačkoli správce front toto není vynucen.

Logická zpráva
Jedná se o jedinou jednotku informací o aplikaci. V případě neexistence systémových omezení by logická zpráva byla stejná jako fyzická zpráva. Pokud však logické zprávy jsou velké, omezení systému mohou učinit vhodné nebo nezbytné k rozdělení logické zprávy do dvou nebo více fyzických zpráv, které se nazývají *segmenty*.

Logická zpráva, která byla rozdělena na segmenty, se skládá ze dvou nebo více fyzických zpráv, které mají stejný identifikátor skupiny bez hodnoty null (pole*MDGID* v MQMD) a stejné pořadové číslo zprávy (pole*MDSEQ* v MQMD). Segmenty jsou odlišeny lišícími hodnotami pro offset segmentu (pole*MDOFF* v MQMD), který poskytuje odchylku dat ve fyzické zprávě od začátku dat v logické zprávě. Vzhledem k tomu, že každý segment je fyzická zpráva, segmenty v logické zprávě mají obvykle odlišné identifikátory zpráv.

Logická zpráva, která nebyla segmentována, ale jejíž segmentace byla povolena odesílající aplikací, má také neprázdný identifikátor skupiny, ačkoli v tomto případě existuje pouze jedna fyzická zpráva s identifikátorem skupiny, pokud tato logická zpráva nepatří do skupiny zpráv. Logické zprávy, pro které má být segmentace zablokována 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 identifikátor skupiny bez hodnoty null. Logické zprávy ve skupině jsou rozlišeny odlišnými hodnotami pro pořadové číslo zprávy, což je celé číslo v rozsahu od 1 do n, kde n je počet logických zpráv ve skupině. Je-li jedna nebo více logických zpráv segmentovaná, ve skupině jsou více než n fyzických zpráv.

PMLOGO
Zprávy ve skupinách a segmentech logických zpráv jsou uvedeny v logickém pořadí.

Tato volba sděluje správci front, jak aplikace vkládá zprávy do skupin a segmentů logických zpráv. Může být zadán pouze na volání MQPUT; není platný na volání MQPUT1 .

Je-li uveden PMLOGO, znamená to, že aplikace používá po sobě jdoucí volání MQPUT k:

- 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.

Tento příkaz se nazývá "logické pořadí".

Vzhledem k tomu, že aplikace sdělila správci front, jak vkládá zprávy do skupin a segmentů logických zpráv, aplikace nemusí udržovat a aktualizovat informace o skupinách a segmentech o každém volání MQPUT, protože správce front toto provádí. Konkrétně to znamená, že aplikace nemusí nastavovat pole *MDGID*, *MDSEQ*a *MDOFF* v MQMD, protože správce front tyto hodnoty nastavuje na příslušné hodnoty. Aplikace potřebuje nastavit pouze pole *MDMFL* v produktu MQMD, aby označilo, kdy zprávy patří do skupin nebo jsou segmenty logických zpráv, a označují poslední zprávu ve skupině nebo posledním segmentu logické zprávy.

Po spuštění skupiny zpráv nebo logické zprávy musí následná volání MQPUT uvádět příslušné příznaky MF* v produktu *MDMFL* v produktu MQMD. Pokud se aplikace pokusí vložit zprávu do skupiny, když existuje neukončená skupina zpráv, nebo vložit zprávu, která není segmentem, když se jedná o neukončené logické zprávy, volání selže s kódem příčiny RC2241 nebo RC2242 . Správce front však uchovává informace o aktuální skupině zpráv nebo aktuální logické zprávě a aplikace je může ukončit odesláním zprávy (případně bez dat zprávy aplikace) zadáním parametru MFLMIG nebo MFLSEG, než znovu odešlete volání MQPUT tak, aby vložila zprávu, která není ve skupině, nebo se nejedná o segment.

Příkaz Tabulka 717 na stránce 1165 zobrazuje kombinace voleb a příznaků, které jsou platné, a hodnoty polí *MDGID*, *MDSEQ*a *MDOFF* , které správce front používá v každém případě. Kombinace voleb a příznaků, které nejsou zobrazeny v tabulce, jsou neplatné. Sloupce v tabulce mají následující význam:

PROTOKOL
Označuje, zda je volba PMLOGO uvedena na volání.

MIGOCITY
Označuje, zda je ve volání zadána volba MFMIG nebo MFLMIG.

SEGG
Označuje, zda je volba MFSEG nebo MFLSEG uvedená na volání.

SEG OK
Označuje, zda je volba MFSEGA uvedená na volání.

Cur grp
Indikuje, zda před voláním existuje aktuální skupina zpráv.

Zpráva protokolu cur
Indikuje, zda před voláním existuje aktuální logická zpráva.

Ostatní sloupce
Zobrazení hodnot, které správce front používá. "Předchozí" označuje hodnotu použitou pro pole v předchozí zprávě pro popisovač fronty.

PMRLOC
Určuje, že PMRQN ve struktuře MQPMO musí být dokončen s názvem lokální fronty, do které se zpráva skutečně dostane. Název ResolvedQMgrje podobně doplněn názvem lokálního správce front, který je hostitelem lokální fronty. Viz OORLOQ pro to, co to znamená. Je-li uživatel oprávněn pro vložení do fronty, mají oprávnění k uvedení tohoto příznaku v rámci volání MQPUT. Není třeba žádné zvláštní oprávnění.

Tabulka 717. Volby MQPUT související se zprávami ve skupinách a segmentech logických zpráv

Volby, které uvedete				**Stav skupiny a protokolu-zpráva před voláním**		**Hodnoty, které správce front používá**		
LOG SLOVO	**MIG**	**SEG**	**SEG OK**	**Cur grp**	**Zpráva Cur msg**	*MDGID*	*MDSEQ*	*MDOFF*
Ano	Ne	Ne	Ne	Ne	Ne	GINON	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	Předchozí ID skupiny	1	Předchozí odchylka + 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	Předchozí ID skupiny	Předchozí pořadové číslo + 1	0

Tabulka 717. Volby MQPUT související se zprávami ve skupinách a segmentech logických zpráv (pokračování)

Volby, které uvedete				**Stav skupiny a protokolu-zpráva před voláním**		**Hodnoty, které správce front používá**		
Ano	Ano	Ano	YES nebo NO	Ano	Ano	Předchozí ID skupiny	Předchozí pořadové číslo	Předchozí odchylka + předchozí délka segmentu
Ne	Ne	Ne	Ne	YES nebo NO	YES nebo NO	GINON	1	0
Ne	Ne	Ne	Ano	YES nebo NO	YES nebo NO	Nové ID skupiny, je-li GINONE, hodnota else v poli	1	0
Ne	Ne	Ano	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, je-li GINONE, hodnota else v poli	1	Hodnota v poli
Ne	Ano	Ne	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, je-li GINONE, hodnota else v poli	Hodnota v poli	0
Ne	Ano	Ano	YES nebo NO	YES nebo NO	YES nebo NO	Nové ID skupiny, je-li GINONE, hodnota else v poli	Hodnota v poli	Hodnota v poli

Poznámka:

- PMLOGO není platný na volání MQPUT1 .
- Pro pole *MDMID* správce front vygeneruje nový identifikátor zprávy, pokud je zadáno PMNMID nebo MINONE, a použije hodnotu v poli jinak.
- Pro pole *MDCID* správce front vygeneruje nový korelační identifikátor, je-li zadáno PMNCID, a použije hodnotu v poli jinak.

Je-li uvedeno PMLOGO, správce front vyžaduje, aby všechny zprávy ve skupině a segmenty v logické zprávě byly vloženy se stejnou hodnotou do pole *MDPER* v MQMD, tj. všechny 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, jak je uvedeno:

- 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 jednotky práce, pokud se použije stejný popisovač fronty. Nemusejí však být uvedeny do stejné jednotky práce. To umožňuje skupině zpráv nebo logické zprávě složené z mnoha fyzických zpráv, které mají být rozděleny mezi dvě nebo více po sobě jdoucích jednotek práce pro daný popisovač fronty.
- Pokud se první fyzická zpráva ve skupině nebo logické zprávě nevloží do pracovní jednotky, žádná z jiných fyzických zpráv ve skupině nebo logické zprávě nemůže být vložena do pracovní jednotky, pokud se použije stejný popisovač fronty.

Nejsou-li tyto podmínky splněny, volání MQPUT se nezdaří s kódem příčiny RC2245 .

Je-li zadáno PMLOGO, deskriptor MQMD dodaný v rámci volání MQPUT nesmí být menší než MDVER2. Není-li tato podmínka splněna, volání selže s kódem příčiny RC2257 .

Není-li PMLOGO uveden, zprávy ve skupinách a segmentech logických zpráv lze vložit do libovolného pořadí a není nutné vkládat úplné skupiny zpráv nebo úplné logické zprávy. Je odpovědností aplikace, aby zajistila, že pole *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* mají odpovídající hodnoty.

Jedná se o techniku, kterou lze použít k restartování skupiny zpráv nebo logické zprávy ve středu, po selhání systému. Když se systém restartuje, může aplikace nastavit pole *MDGID*, *MDSEQ*, *MDOFF*, *MDMFL*a *MDPER* na příslušné hodnoty a pak vydat volání MQPUT s PMSYP nebo PMNSYP nastavit jako *nezbytné*, ale bez uvedení PMLOGO. Je-li toto volání úspěšné, správce front uchová informace o skupině a segmentu a následná volání MQPUT používající tento manipulátor fronty mohou určit PMLOGO jako normální.

Informace o skupinách a segmentech, které správce front uchovává pro volání MQPUT, jsou odděleny od informací o skupině a segmentu, které si zachovává pro volání MQGET.

Pro daný popisovač fronty může aplikace volně míchat volání MQPUT, která uvádí PMLOGO s voláními MQPUT, ale následující body by měly být uvedeny níže:

- Není-li parametr PMLOGO zadán, každá úspěšná volání MQPUT způsobí, že správce front nastaví informace o skupině a segmentu pro manipulátor fronty na hodnoty zadané aplikací. Tato operace nahradí existující informace o skupině a segmentech zachované správcem front pro manipulátor fronty.
- Není-li PMLOGO uveden, volání se nezdaří, pokud existuje aktuální skupina zpráv nebo logická zpráva; volání však může být úspěšné s kódem dokončení CCWARN. Tabulka 718 na stránce 1167 zobrazuje různé případy, které mohou nastat. V těchto případech, pokud kód dokončení není CCOK, kód příčiny je jeden z následujících (podle potřeby):
  - RC2241
  - RC2242
  - RC2185
  - RC2245

  Poznámka: Správce front nekontroluje informace o skupině a segmentu pro volání MQPUT1 .

Tabulka 718. 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	CCFIL	CCFIL
MQPUT bez PMLOGO	CCWARN	KEK
MQCLOSE s neukončené skupinou nebo logickou zprávou	CCWARN	KEK

Aplikace, které pouze chtějí vložit zprávy a segmenty do logického pořadí, se doporučuje uvést PMLOGO, protože se jedná o nejjednodušší volbu, která se má použít. Tato volba zbavuje aplikaci potřeby spravovat informace o skupinách a segmentech, protože tyto informace spravuje správce front. Avšak specializované aplikace mohou vyžadovat více kontroly, než je poskytnuto pomocí volby PMLOGO, a toho lze dosáhnout neuvedením této volby. Po provedení této akce musí aplikace zajistit správné nastavení polí *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v MQMD, a to před každým voláním MQPUT nebo MQPUT1 .

Například aplikace, která chce předávat fyzické zprávy, které přijímá, bez ohledu na to, zda jsou tyto zprávy ve skupinách nebo segmentech logických zpráv, nesmí uvádět PMLOGO. To má dva důvody:

- Pokud jsou zprávy načteny a uvedeny do pořadí, uvedení PMLOGO způsobí přiřazení nového identifikátoru skupiny ke zprávám, a to může ztížit nebo nemožné, aby původce zpráv koreloval jakoukoli zprávu odpovědi nebo zprávy, které jsou výsledkem skupiny zpráv.
- Ve složité síti s více cestami mezi odesílajícím a přijímajícím správcem front může dojít k nedostatku fyzických zpráv v pořadí. Neuvedení PMLOGO a odpovídající GMLOGO na volání MQGET, může předávající aplikace načítat a předávat každou fyzickou zprávu ihned, jakmile dorazí, aniž by bylo nutné čekat na to, až se objeví další, v logickém pořadí.

Aplikace, které generují zprávy hlášení pro zprávy ve skupinách nebo segmentech logických zpráv, nesmí při vložení zprávy sestavy uvádět také PMLOGO.

PMLOGO lze zadat s libovolnou z ostatních voleb PM*.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

PMNOC
K této zprávě nemá být přidružen žádný kontext.

Kontext identity i původ jsou nastaveny tak, aby neoznačovaly žádný kontext. To znamená, že pole kontextu v MQMD jsou nastavena na:

- Mezery pro znaková pole
- Hodnoty null pro bajtová pole
- Nuly pro číselná pole

PMDEFC
Použijte výchozí kontext.

Zpráva má mít k sobě přidružené informace o kontextu, pro identitu i pro původ. Správce front nastaví pole kontextu v deskriptoru zpráv následujícím způsobem:

Tabulka 719. Výchozí hodnoty informací o kontextu pro pole MQMD

Pole v MQMD	Použitá hodnota
MDUID	Určeno z prostředí, je-li to možné; jinak nastavte prázdné znaky.
MDACC	Určeno z prostředí, je-li to možné; jinak nastavte hodnotu ACNONE.
MDAID	Nastavit na mezery.
MDPAT	Určeno z prostředí.
MDPAN	Určeno z prostředí, je-li to možné; jinak nastavte prázdné znaky.
MDPD	Nastavit na datum, kdy je zpráva vložena.
MDPT	Nastavit čas, kdy je zpráva vložena.
MDAOD	Nastavit na mezery.

Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Jedná se o výchozí akci, nejsou-li zadány žádné volby kontextu.

PMPASI
Předat kontext identity z ovladače vstupní fronty.

Zpráva má k sobě přidružené informace o kontextu. Kontext identity je převzat z manipulátoru fronty uvedeného v poli *PMCT* . Informace o kontextu výchozího bodu je vygenerováno správcem front stejným způsobem jako pro PMDEFC (viz předchozí tabulka pro hodnoty). Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Pro volání MQPUT musí být fronta otevřena pomocí volby OOPASI (nebo volba, která jej označuje). Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou OOPASI.

PMPASA
Předejte všechny kontext z ovladače vstupní fronty.

Zpráva má k sobě přidružené informace o kontextu. Obojí identity a původ jsou převzaty z manipulátoru fronty uvedeného v poli *PMCT* . Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOPASA (nebo s volbou, která jej označuje). Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou OOPASA.

PMSETI
Nastavte kontext identity z aplikace.

Zpráva má k sobě přidružené informace o kontextu. Aplikace určuje kontext identity ve struktuře MQMD. Informace o kontextu výchozího bodu je vygenerováno správcem front stejným způsobem jako pro PMDEFC (viz předchozí tabulka pro hodnoty). Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Pro volání MQPUT musí být fronta otevřena pomocí volby OOSETI (nebo volba, která jej označuje). Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou OOSETI.

PMSETA
Nastavte veškerý kontext z aplikace.

Zpráva má k sobě přidružené informace o kontextu. Aplikace určuje kontext identity a původu ve struktuře MQMD. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Pro volání MQPUT musí být fronta otevřena s volbou OOSETA. Pro volání MQPUT1 bude provedena stejná kontrola autorizace jako pro volání MQOPEN s volbou OOSETA.

Zadán může být pouze jedna z voleb kontextu PM*. Není-li zadána žádná z těchto voleb, předpokládá se PMDEFC.

Zadejte typy odezvy. Následující volby řídí odezvu vrácenou na volání MQPUT nebo MQPUT1 . Můžete zadat pouze jednu z těchto voleb. Nejsou-li zadány hodnoty PMARES a PMSRES, předpokládá se PMRASQ nebo PMRAST.

PMŮRY

Volba PMARES vyžaduje dokončení operace MQPUT nebo MQPUT1 , aniž by aplikace čekala na správce front, aby mohl dokončit volání. Použití této volby může zlepšit výkon systému zpráv, zejména u aplikací používajících vazby klienta. Aplikace může periodicky kontrolovat pomocí příkazu MQSTAT, zda k chybě došlo během předchozích asynchronních volání.

Při použití této volby budou v produktu MQMD zaručena pouze následující pole:

- MDAID
- MDPAT
- MDPAN
- MDAODCITY

Kromě toho, pokud je zadán jeden nebo oba PMNMID nebo PMNCID jako volby, vrátí se také MDMID a MDCID. (PMNMID může být implicitně zadáno uvedením prázdného pole MDMID).

Vyplní se pouze dříve uvedená pole. Další informace, které by normálně byly vráceny ve struktuře MQMD nebo MQPMO, nejsou definovány.

Při požadavku na asynchronní odeslání odezvy pro volání MQPUT nebo MQPUT1, hodnota CMPCOD a REASON CCOK a RCNONE nezbytně neznamená, že zpráva byla úspěšně vložena do fronty. Při vývoji aplikace MQI, která používá asynchronní odezvu vložení a je třeba potvrdit, že zprávy byly vloženy do fronty, je třeba z operací vkládání zkontrolovat kódy CMPCOD a REASON a také pomocí příkazu MQSTAT zadávat dotazy na informace o asynchronních chybách.

Ačkoli se nezdařilo okamžitě vrátit úspěch nebo selhání jednotlivých volání MQPUT/MQPUT1 , může být první chyba, která se vyskytla při asynchronním volání, určena později při volání funkce MQSTAT.

Pokud se nepodaří doručit trvalou zprávu pod synchronizačním bodem pomocí asynchronní odezvy vložení a pokusíte transakci potvrdit, operace commit selže a transakce bude vrácena s kódem dokončení CCFAIL a z důvodu RC2003 . Aplikace může provést volání funkce MQSTAT a určit příčinu předchozího selhání operace MQPUT nebo MQPUT1 .

PMSRES.
Zadáním této hodnoty pro volbu vložení do struktury MQPMO zajistíte, aby byla operace MQPUT nebo MQPUT1 vždy vydána synchronně. Je-li operace úspěšná, dokončí se všechna pole v MQMD a MQPMO. Je k dispozici pro zajištění synchronní odezvy bez ohledu na výchozí hodnotu odezvy vložení definovanou na objektu fronty nebo tématu.

PMRAQ
Je-li tato hodnota zadána pro volání MQPUT, použije se použitý typ odezvy vložení z hodnoty DEFPRESP zadané ve frontě při jeho otevření aplikací. Je-li aplikace klienta připojena ke správci front na úrovni dřívější než IBM WebSphere MQ 7.0, chová se, jako by byla zadána položka PMSRES.

Je-li tato volba zadána pro volání MQPUT1 , nebude hodnota DEFPRESP z definice fronty použita. 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 témat.

Další volby: Následující volby kontrolují kontrolu autorizace, a to, co se stane, když správce front přechází do klidového stavu:

PMALTU
Validovat 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 vkládání zpráv do fronty. Volání může být úspěšné pouze v případě, že je tento *ODAU* autorizován k otevření fronty s uvedenými volbami, bez ohledu na to, zda je identifikátor uživatele, pod kterým je aplikace spuštěna, oprávněn tak učinit. (To však neplatí pro zadané volby kontextu, které jsou vždy zkontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.)

Tato volba je platná pouze s voláním MQPUT1 .

PMFIQ.
Selhání, pokud je správce front uváděn do klidového stavu.

Tato volba vynutí selhání volání MQPUT nebo MQPUT1 v případě, že je správce front ve stavu uvedení do klidového stavu.

Volání vrátí kód dokončení CCFAIL s kódem příčiny RC2161 .

Výchozí volba: Pokud žádná z dříve popsaných voleb není povinná, lze použít následující volbu:

PMNONE
Nejsou uvedeny žádné volby.

Tato hodnota může být použita k označení, že nebyly zadány žádné další volby; všechny volby předpokládají jejich výchozí hodnoty. PMNONE je definován pro dokumentaci programu pomoci; není zamýšlen, že tato volba je použita s jinou, ale jako její hodnota je nula, takové použití nelze detekovat.

Toto je vstupní pole. Počáteční hodnota pole *PMOPT* je PMNONE.

PMPRF (10ciferné celé číslo se znaménkem)

Příznaky určující, která pole MQPMR jsou přítomna.

Toto pole obsahuje příznaky, které musí být nastaveny tak, aby určovaly, která pole MQPMR se nacházejí v záznamech vložených zpráv poskytovaných aplikací. *PMPRF* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *PMREC* je nula, nebo obě *PMPRO* a *PMPRP* jsou nula.

Pro pole, která jsou přítomná, používá správce front pro každou cílovou hodnotu hodnoty z polí v odpovídajícím záznamu vložení zprávy. U nepřítomných polí používá správce front hodnoty z struktury MQMD.

Je možné zadat jeden nebo více následujících příznaků, které indikují, která pole se nacházejí v záznamech vložených zpráv:

PFMID
Zobrazí se pole identifikátoru zprávy.

PFARCID
Pole identifikátoru korelace je přítomno.

PFGID
Pole identifikátoru skupiny je přítomno.

PFFB
Je přítomno pole zpětné vazby.

PFAC
Pole Účetní-token je přítomno.

Je-li tento příznak zadán, musí být v poli *PMOPT* uvedena hodnota PMSETI nebo PMSETA; pokud tato podmínka není splněna, volání selže s kódem příčiny RC2158 .

Nejsou-li přítomna žádná pole MQPMR, lze zadat následující:

PFNONE
Nejsou přítomna žádná pole záznamu vložení zprávy.

Je-li tato hodnota uvedena, musí být buď *PMREC* nula, nebo obě *PMPRO* a *PMPRP* musí být nula.

PFNONE je definován v dokumentaci programu podpory. Není určeno, aby byla tato konstanta použita spolu s jinou, ale protože její hodnota je nula, takové použití nelze detekovat.

Pokud příkaz *PMPRF* obsahuje příznaky, které nejsou platné, nebo jsou zadány záznamy zpráv, 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 *PMVER* je menší než PMVER2.

PMPRO (10ciferné celé číslo se znaménkem)

Odstup prvního záznamu vložení zprávy od začátku MQPMO.

Jedná se o posun v bajtech prvního záznamu vložení zprávy MQPMR ze začátku struktury MQPMO. Odsazení může být kladné nebo záporné. *PMPRO* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *PMREC* je nula.

Když je zpráva vložena do distribučního seznamu, může být poskytnuto pole jednoho nebo více záznamů vložení zpráv MQPMR, aby bylo možné určit určité vlastnosti zprávy pro každý cíl jednotlivě; tyto vlastnosti jsou:

- Identifikátor zprávy
- identifikátor korelace
- Identifikátor skupiny
- hodnota zpětné vazby
- Token evidence

Není nutné uvádět všechny tyto vlastnosti, ale jakákoli podmnožina je vybrána, pole musí být uvedena ve správném pořadí. Další podrobnosti naleznete v popisu struktury MQPMR.

Obvykle by mělo být tolik vložených záznamů zpráv, protože při otevření distribučního seznamu jsou záznamy objektů zadány příkazem MQOD; každý záznam vložení zprávy poskytuje vlastnosti zprávy pro frontu označenou odpovídajícím záznamem objektu. Fronty v rozdělovníku, které se nedaří otevřít, musí stále dát záznamy zpráv přidělené pro ně na příslušných pozicích v poli, ačkoli jsou vlastnosti zprávy v tomto případě ignorovány.

Je možné, že se počet záznamů vložení zpráv bude lišit od počtu záznamů objektů. Pokud existuje méně záznamů vložených zpráv než záznamů objektů, vlastnosti zprávy pro místa určení, které nevloží záznamy zpráv, jsou převzaty z odpovídajících polí v deskriptoru zpráv MQMD. Pokud existuje více záznamů vložení zpráv než záznamů objektů, přebytek se nepoužije (ačkoli musí být stále možné k nim přistupovat). Záznamy zpráv o vložení jsou volitelné, ale pokud jsou dodány, musí být *PMREC* z nich.

Záznamy vložení zpráv mohou být poskytnuty podobným způsobem jako záznamy objektů v MQOD, a to buď uvedením offsetu v *PMPRO*, nebo zadáním adresy v *PMPRP* ; Podrobnosti o tom, jak to provést, naleznete v poli *ODORO* , které je popsáno v “MQOD (Object descriptor) na systému IBM i” na stránce 1147.

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 *PMVER* je menší než PMVER2.

PMPRP (ukazatel)

Adresa prvního záznamu zprávy vložení.

Jedná se o adresu prvního záznamu vložení zprávy MQPMR. *PMPRP* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *PMREC* je nula.

Lze použít buď *PMPRP* , nebo *PMPRO* k uvedení záznamů vložení zpráv, ale ne obojí; podrobnosti najdete v popisu pole PMRRO . Není-li parametr *PMPRP* použit, musí být nastaven na nulový ukazatel nebo na null bajtů.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud *PMVER* je menší než PMVER2.

PMREC (10ciferné celé číslo se znaménkem)

Počet vložených záznamů zpráv nebo záznamů odpovědí.

Jedná se o počet vložených záznamů zpráv MQPMR nebo záznamů odpovědí MQRR, které byly poskytnuty aplikací. Toto číslo může být větší než nula pouze v případě, že je zpráva vložena do distribučního seznamu. Záznamy zprávy a záznamy odpovědí jsou volitelné-aplikace nemusí poskytovat žádné záznamy, nebo může poskytnout záznamy pouze jednoho typu. Avšak, pokud aplikace poskytuje záznamy obou typů, musí poskytnout záznamy *PMREC* každého typu.

Hodnota *PMREC* nemusí být stejná jako počet míst určení v rozdělovníku. Je-li zadáno příliš mnoho záznamů, přebytečné nejsou použity; je-li uvedeno příliš málo záznamů, použijí se výchozí hodnoty pro vlastnosti zprávy pro ty cíle, které nevloží záznamy zpráv (viz *PMPRO* dále v tomto tématu).

Je-li *PMREC* menší než nula nebo je větší než nula, ale zpráva se nedistribuuje na distribuční seznam, volání selže s kódem příčiny RC2154 .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *PMVER* je menší než PMVER2.

PMRMN (48 bajtů znakového řetězce)

Vyřešený název správce cílové fronty.

Jedná se o název cílového správce front po provedení rozpoznání názvu pomocí lokálního správce front. Vrácený název je název správce front, který vlastní frontu, kterou identifikuje produkt *PMRQN*, a může to být název lokálního správce front.

Pokud *PMRQN* je sdílená fronta, kterou vlastní skupina sdílení front, do níž patří lokální správce front, *PMRMN* je název skupiny sdílení front. Pokud je fronta vlastníkem některé jiné skupiny sdílení front,

může být produktem *PMRQN* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (charakter vráceného výsledku je určen definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt rozdělovník nebo téma, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána LNQMN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

PMRQN (48 bajtů znakového řetězce)

Vyřešený název cílové fronty.

Jedná se o název cílové fronty po provedení rozpoznání názvu pomocí lokálního správce front. Vrácený název je název fronty, která existuje ve správci front identifikovaném příkazem *PMRMN*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je jedinou frontou; pokud je objekt rozdělovník nebo téma, vrácená hodnota není definována.

Toto je výstupní pole. Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

PMRRO (10ciferné celé číslo se znaménkem)

Posunutí prvního záznamu odpovědi od začátku MQPMO.

Jedná se o posun v bajtech prvního záznamu odezvy MQRR od začátku struktury MQPMO. Odsazení může být kladné nebo záporné. *PMRRO* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *PMREC* je nula.

Při vložení zprávy do distribučního seznamu může být k dispozici pole jednoho nebo více záznamů odpovědí MQRR, aby bylo možné identifikovat fronty, do kterých nebyla zpráva úspěšně odeslána (pole*RRCC* v MQRR), a důvod pro každé selhání (pole*RRREA* v MQRR). Je možné, že zpráva nebyla odeslána, protože došlo k otevření fronty, nebo došlo k selhání 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 se nezdařily, nebo všechny selhaly, ale z různých důvodů); kód příčiny RC2136 z volání označuje tento případ. Pokud se stejný kód příčiny vztahuje na všechny fronty, je tento důvod vrácen v parametru **REASON** volání MQPUT nebo MQPUT1 a záznamy odezvy nejsou nastaveny.

Obvykle by mělo být tolik záznamů odezev, jak jsou záznamy objektů zadány příkazem MQOD při otevření distribučního seznamu; je-li to nutné, každý záznam odpovědi se nastaví na kód dokončení a kód příčiny pro vložení do fronty označené záznamem odpovídajícího objektu. Fronty v rozdělovníku, které se nedaří otevřít, musí mít stále alokovány záznamy odpovědí na odpovídajících pozicích v poli, ačkoli jsou nastaveny na kód dokončení a kód příčiny, který je výsledkem operace otevření, spíše než operace vložení.

Je možné, aby se počet záznamů odpovědí lišil od počtu záznamů objektu. Pokud existuje méně záznamů odezev než záznamů objektů, nemusí být možné, aby aplikace identifikovala všechna místa určení, pro která selhala operace put, nebo důvody pro selhání. Pokud existuje více záznamů odezev než záznamů objektů, přebytek se nepoužije (ačkoli musí být stále možné k nim přistupovat). Záznamy odezvy jsou volitelné, ale pokud jsou dodány, musí být *PMREC* z nich.

Záznamy odezvy mohou být poskytnuty podobným způsobem jako záznamy objektů v MQOD, a to buď uvedením offsetu v *PMRRO*, nebo zadáním adresy v *PMRRP* ; Podrobnosti o tom, jak to provést, naleznete v poli *ODORO* , které je popsáno v “MQOD (Object descriptor) na systému IBM i” na stránce 1147. Avšak, nelze 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 nulové. Důvodem je to, že informace o odezvě (je-li požadována) jsou vraceny v záznamech odpovědí určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *PMVER* je menší než PMVER2.

PMRRP (ukazatel)

Adresa prvního záznamu odezvy.

Jedná se o adresu prvního záznamu odezvy MQRR. *PMRRP* se používá pouze tehdy, když je zpráva vložena do rozdělovníku. Pole je ignorováno, pokud *PMREC* je nula.

Lze použít buď *PMRRP* nebo *PMRR0* k uvedení záznamů odpovědí, ale ne obojí; podrobnosti najdete v popisu pole PMRRO . Není-li parametr *PMRRP* použit, musí být nastaven na nulový ukazatel nebo na null bajtů.

Pro volání MQPUT1 musí být toto pole ukazatelem null nebo null bajtů. Důvodem je to, že informace o odezvě (je-li požadována) jsou vraceny v záznamech odpovědí určených deskriptorem objektu MQOD.

Toto je vstupní pole. Počáteční hodnota tohoto pole je ukazatel Null. Toto pole je ignorováno, pokud *PMVER* je menší než PMVER2.

PMSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

PMSIDV
Identifikátor struktury voleb put-message.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PMSIDV.

PMSL (MQLONG)

Úroveň odběru, na kterou je tato publikace zaměřena.

Pouze ti odběry s nejvyšší *PMSL* menší nebo rovnou této hodnotě obdrží tuto publikaci. Tato hodnota musí být v rozsahu nula až 9; nula je nejnižší úroveň.

Počáteční hodnota tohoto pole je 9.

PMTO (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je -1.

PMUDC (10ciferné celé číslo se znaménkem)

Počet zpráv odeslaných úspěšně do vzdálených front.

Jedná se o počet zpráv, které aktuální volání MQPUT nebo MQPUT1 úspěšně odeslalo do front v rozdělovníku, které se interpretují do vzdálených front. Zprávy, které správce front dočasně uchovává v seznamu položek rozdělovníku jako počet jednotlivých míst určení, 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 *PMVER* je menší než PMVER2.

PMVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být jedna z následujících:

PMVER1
Struktura volby put-message Version-1 .

PMVER2
Struktura voleb vložených zpráv Version-2 .

Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

PMVERC
Aktuální verze struktury voleb put-message.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je PMVER1.

Počáteční hodnoty

Tabulka 720. Počáteční hodnoty polí 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
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	Nulový ukazatel nebo bajty null
PMRRP	Není	Nulový ukazatel nebo bajty 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 (Put-message record) v systému IBM i

Struktura MQPMR se používá k určení různých vlastností zpráv pro jediné místo určení, když je zpráva vložena do distribučního seznamu.

Přehled

Účel: MQPMR je struktura vstupu/výstupu pro volání MQPUT a MQPUT1 .

Znaková sada a kódování: Data v MQPMR musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur na volání MQPUT nebo MQPUT1 je možné zadat různé hodnoty pro každou cílovou frontu v rozdělovníku. Některá z těchto polí jsou vstupem, jiné jsou vstupy/výstupy.

Poznámka: Tato struktura je neobvyklá v tom, že nemá pevné rozvržení. Pole v této struktuře jsou volitelná a přítomnost nebo nepřítomnost každého pole je indikována příznaky v poli *PMPRF* v MQPMO. Pole, která jsou přítomna ***se musí vyskytnout v následujícím pořadí*** :

- *PRMID*
- *PRCID*
- *PRGID*
- *PRFB*
- *PRACC*

Pole, která nejsou přítomna, nezabírají žádný prostor v záznamu.

Vzhledem k tomu, že MQPMR nemá pevné rozvržení, není v souboru COPY poskytnuta žádná definice. Programátor aplikace by měl vytvořit deklaraci obsahující pole, která jsou vyžadována aplikací, a nastavit příznaky v produktu *PMPRF* tak, aby určovaly pole, která jsou k dispozici.

- “Pole” na stránce 1176
- “Počáteční hodnoty” na stránce 1178
- “Deklarace RPG” na stránce 1178

Pole

Struktura MQPMR obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

PRACC (32bitový bitový řetězec)

Token evidence.

Jedná se o účtovací token, který se má použít pro zprávu odeslanou do fronty s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MDACC* v produktu MQMD pro vložení do jedné fronty. Chcete-li získat informace o obsahu tohoto pole, prohlédněte si popis *MDACC* v "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099 .

Není-li toto pole k dispozici, bude použita hodnota v produktu 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 názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MDCID* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta místa určení, která nemají záznam MQPMR obsahující pole *PRCID* .

Je-li zadáno PMNCID, je vygenerován a použit *jediný* nový korelační identifikátor, který se použije pro všechna místa určení v seznamu distribucí bez ohledu na to, zda mají záznamy MQPMR. To se liší od způsobu zpracování PMNMID (viz pole *PRMID*).

Jedná se o vstupní/výstupní pole.

PRFB (10číslicové celé číslo se znaménkem)

Zpětná vazba nebo kód příčiny.

Jedná se o kód zpětné vazby, který má být použit pro zprávu odeslanou do fronty názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MDFB* v produktu MQMD pro vložení do jedné fronty.

Není-li toto pole k dispozici, bude použita hodnota v produktu MQMD.

Toto je vstupní pole.

PRGID (24bajtový bitový řetězec)

Identifikátor skupiny.

Jedná se o identifikátor skupiny, který se má použít pro zprávu odeslanou do fronty s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MDGID* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta místa určení, která nemají záznam MQPMR obsahující pole *PRGID* . Hodnota je zpracována jako dokumentovaná v Tabulka 717 na stránce 1165, ale s následujícími rozdíly:

- V případech, kdy se použije nový identifikátor skupiny, vygeneruje správce front jiný identifikátor skupiny pro každé místo určení (to znamená, že žádná dvě místa určení nemají stejný identifikátor skupiny).
- V těch případech, kdy se použije hodnota v poli, se volání nezdaří s kódem příčiny RC2258.

Jedná se o vstupní/výstupní pole.

PRMID (24bajtový bitový řetězec)

Identifikátor zprávy.

Jedná se o identifikátor zprávy, který má být použit pro zprávu odeslanou do fronty názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 . Zpracovává se stejným způsobem jako pole *MDMID* v produktu MQMD pro vložení do jedné fronty.

Pokud toto pole není přítomno v záznamu MQPMR, nebo existuje méně záznamů MQPMR, než cíle, hodnota ve struktuře MQMD se použije pro ta místa určení, která nemají záznam MQPMR obsahující pole *PRMID* . Je-li tato hodnota MINONE, je vygenerován nový identifikátor zprávy pro *každý* z těchto míst určení (tj. žádné dvě z těchto míst určení nemají stejný identifikátor zprávy).

Je-li zadáno PMNMID, nové identifikátory zpráv jsou generovány pro všechna místa určení v seznamu distribucí bez ohledu na to, zda mají záznamy MQPMR. To se liší od způsobu zpracování procesu PMNCID (viz pole *PRCID*).

Jedná se o vstupní/výstupní pole.

Počáteční hodnoty

Pro tuto strukturu nejsou definovány žádné počáteční hodnoty, protože není poskytnuta žádná deklarace struktury. Následující ukázkové deklarace ukazuje, jak by měl aplikační programátor deklarovat strukturu, pokud jsou všechna pole povinná.

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 záhlaví formátování) v systému 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 produktu *RFNVS*) jsou ve znakové sadě a kódování zadané v polích *MDCSI* a *MDENC* ve struktuře záhlaví, která předchází MQRFH, nebo podle polí ve struktuře MQMD, pokud je MQRFH na začátku dat zprávy aplikace.

Znaková sada musí být taková, která má jednobajtové znaky pro znaky, které jsou platné v názvech front.

- “Pole” na stránce 1178
- “Počáteční hodnoty” na stránce 1181
- “Deklarace RPG” na stránce 1181

Pole

Struktura MQRFH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RFCSI (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady, která následuje za *RFNVS*.

Tato hodnota určuje identifikátor znakové sady dat, která následují za *RFNVS* ; Nevztahuje se na znaková data v samotné struktuře MQRFH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

Číselné kódování dat, která následují za *RFNVS*.

Tato hodnota určuje číselné kódování dat, která jsou následující: *RFNVS* ; Nevztahuje se na číselná data ve struktuře MQRFH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RFFLG (Celé číslo s 10 číslicemi)

Příznaky.

Je možné zadat následující:

RFNONE
Žádné vlajky.

Počáteční hodnota tohoto pole je RFNONE.

RFFMT (8bajtový znakový řetězec)

Název formátu dat, který následuje za *RFNVS*.

Uvádí název formátu dat, která následují za *RFNVS*.

Na základě 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.

Počáteční hodnota tohoto pole je FMNONE.

RFLEN (10ciferné celé číslo se znaménkem)

Celková délka MQRFH včetně *RFNVS*.

Jedná se o délku struktury MQRFH v bajtech, včetně pole *RFNVS* na konci struktury. Tato délka nezahrnuje žádná uživatelská data, která následují za polem *RFNVS* .

Chcete-li se vyvarovat problémů s převodem dat uživatelských dat v některých prostředích, zvažte použití *RFLEN* jako násobného ze čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. o délce kromě pole *RFNVS* :

RFLAV
Délka pevné části struktury MQRFH.

Počáteční hodnota tohoto pole je RFLENV.

RFNVS (n-byte znakový řetězec)

Řetězec obsahující dvojice název-hodnota.

Jedná se o znakový řetězec proměnné délky obsahující dvojice název-hodnota ve formuláři:

```
name1 value1 name2 value2 name3 value3 ...
```

Každý název nebo hodnota musí být oddělena od sousedního názvu nebo hodnoty jedním nebo více prázdnými znaky; tyto mezery nejsou významné. Název nebo hodnota může obsahovat významné mezery tak, že se k názvu nebo hodnotě přidá znak uvozovky; všechny znaky mezi počátečním a odpovídajícím uzavíraným uvozovkem jsou považovány za významné. V následujícím příkladu je název FAMOUS_WORDSa hodnota je `Hello World`:

```
FAMOUS_WORDS "Hello World"
```

Název nebo hodnota může obsahovat jiné znaky než znak null (které se chová jako oddělovač pro *RFNVS*). Aby však mohla aplikace pomoci s interoperabilitou, může aplikace raději omezit názvy na následující znaky:

- První znak: velká nebo malá písmena (A až Z, nebo a až z) nebo podtržítko.
- Následné znaky: velká nebo malá abecední, desetinná číslice (0 až 9), podtržítko, pomlčka nebo tečka.

Pokud název nebo hodnota obsahuje jednu nebo více uvozovek, musí být název nebo hodnota ohraničena uvozovkami a každá uvozovka v řetězci musí být zdvojená:

```
Famous_Words "The program displayed ""Hello World"""
```

Názvy a hodnoty rozlišují velikost písmen, 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 vyvarovat problémů s převodem dat uživatelských dat v některých prostředích, doporučuje se, aby tato délka měla být násobkem čtyř. *RFNVS* musí být vyplněno mezerami do této délky, nebo ukončeno dříve umístěním znaku null za posledním významným znakem v řetězci. Nulový znak a bajty po něm až do zadané délky *RFNVS*jsou ignorovány.

Poznámka: Vzhledem k tomu, že délka tohoto pole není pevná, je pole vynecháno z deklarací struktury, které jsou poskytovány pro podporované programovací jazyky.

RFSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RFSIDV
Identifikátor pro pravidla a formátování struktury záhlaví.

Počáteční hodnota tohoto pole je RFSIDV.

RFVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RFVER1
Pravidla Version-1 a formátovací struktura záhlaví.

Počáteční hodnota tohoto pole je RFVER1.

Počáteční hodnoty

Tabulka 721. Počáteční hodnoty polí v MQRFH		
Název pole	**Název konstanty**	**Hodnota konstanty**
RFSID	RFSIDV	'RFH¬'
RFVER	RFVER1	1
RFLEN	RFLAV	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 formátovací záhlaví 2) v IBM i

Struktura MQRFH2 definuje formát pravidel a záhlaví formátování version-2 .

Přehled

Účel: Toto záhlaví lze použít k odeslání dat, která byla zakódována pomocí syntaxe podobné formátu XML A může obsahovat dvě nebo více struktur MQRFH2 v řadě s uživatelskými daty, které volitelně následují za poslední strukturou MQRFH2 v řadě.

Název formátu: FMRFH2.

Znaková sada a kódování: Speciální pravidla platí pro znakovou sadu a kódování použité pro strukturu MQRFH2 :

- Pole jiná než *RF2NVD* jsou v znakové sadě a kódování, která jsou dána poli *MDCSI* a *MDENC* ve struktuře záhlaví, která předchází MQRFH2, nebo podle polí ve struktuře MQMD, je-li MQRFH2 na začátku dat zprávy aplikace.

  Znaková sada musí být taková, která má jednobajtové znaky pro znaky, které jsou platné v názvech front.

Je-li GMCONV zadán na volání MQGET, převede správce front tato pole na požadovanou znakovou sadu a kódování.

- Parametr *RF2NVD* se nachází ve znakové sadě zadané v poli *RF2NVC* . Pouze určité znakové sady Unicode jsou platné pro *RF2NVC* (podrobnosti viz popis *RF2NVC*).

  Některé znakové sady mají reprezentaci, která je závislá na kódování. Je-li *RF2NVC* jednou z těchto znakových sad, musí být *RF2NVD* ve stejném kódování jako ostatní pole v MQRFH2.

  Je-li GMCONV zadán na volání MQGET, převede správce front *RF2NVD* na požadované kódování, ale nezmění jeho znakovou sadu.

- "Pole" na stránce 1182
- "Počáteční hodnoty" na stránce 1187
- "Deklarace RPG" na stránce 1187

Pole

Struktura MQRFH2 obsahuje následující pole; pole jsou popsána v abecedním pořadí:

RF2CSI (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady dat, který následuje za posledním polem *RF2NVD* .

Tato hodnota určuje identifikátor znakové sady dat, která následuje za posledním polem *RF2NVD* . Nevztahuje se na znaková data ve vlastní struktuře MQRFH2 .

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSINHT.

RF2ENC (10ciferné celé číslo se znaménkem)

Číselné kódování dat za posledním polem *RF2NVD* .

Určuje číselné kódování dat, která následují za posledním polem *RF2NVD* . Nevztahuje se na číselná data ve struktuře MQRFH2 .

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RF2FLG (10ciferné celé číslo se znaménkem)

Příznaky.

Musí být uvedena následující hodnota:

RFNONE
Žádné vlajky.

Počáteční hodnota tohoto pole je RFNONE.

RF2FMT (8bajtový znakový řetězec)

Název formátu dat, který následuje za posledním polem *RF2NVD* .

Uvádí jméno formátu dat, která následuje za posledním polem *RF2NVD* .

Na základě 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.

Počáteční hodnota tohoto pole je FMNONE.

RF2LEN (10ciferné celé číslo se znaménkem)

Celková délka MQRFH2 včetně všech polí *RF2NVL* a *RF2NVD* .

Jedná se o délku v bajtech struktury MQRFH2 , včetně polí *RF2NVL* a *RF2NVD* na konci struktury. Je platný pro více párů polí *RF2NVL* a *RF2NVD* na konci struktury, v posloupnosti:

```
length1, data1, length2, data2, ...
```

RF2LEN nezahrnuje žádná uživatelská data, která mohou následovat za posledním polem *RF2NVD* na konci struktury.

Chcete-li se vyvarovat problémů s převodem dat uživatelských dat v některých prostředích, zvažte použití *RF2LEN* jako násobného ze čtyř.

Následující konstanta udává délku *pevné* části struktury, tj. o délce kromě polí *RF2NVL* a *RF2NVD* :

RFLEN2
Délka pevné části struktury MQRFH2 .

Počáteční hodnota tohoto pole je RFLEN2.

RF2NVC (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady *RF2NVD*.

Tato hodnota určuje identifikátor kódované znakové sady pro data v poli *RF2NVD* . To se liší od znakové sady jiných 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, podmnožina 2.0

17584
UTF-16, verze Unicode 3.0 dílčí sada (obsahuje symbol Euro)

1208
UTF-8, nejnovější podporovaná verze Unicode

Pro znakové sady UTF-16 musí být kódování (pořadí bajtů) produktu *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 na volání MQGET, volání bude dokončeno s kódem příčiny RC2111 a zpráva je vrácena nekonverzovanou.

Počáteční hodnota tohoto pole je 1208.

RF2NVD (n-bajtový znakový řetězec)

Data názvu a hodnoty.

Jedná se o znakový řetězec proměnné délky obsahující data zakódovaná pomocí syntaxe podobné XML. Délka tohoto řetězce je uvedena v poli *RF2NVL* , které předchází poli *RF2NVD* ; tato délka by měla být násobkem čtyř.

Pole *RF2NVL* a *RF2NVD* jsou volitelná, ale pokud jsou přítomna, musí se objevit jako pár a být sousedící. Dvojice polí se mohou opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože tato pole jsou volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

RF2NVD je neobvyklý, protože není převeden na znakovou sadu zadanou na volání MQGET, když je zpráva načtena s volbou GMCONV v platnosti; *RF2NVD* zůstává v původní znakové sadě. Produkt *RF2NVD* se však převede na kódování určené v rámci volání MQGET.

Syntaxe dat název/hodnoty: Řetězec se skládá z jediné "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 značkou konce složky, až do délky definované pomocí *RF2NVL*, musí být prázdné. V rámci složky se každá vlastnost skládá z názvu a hodnoty a volitelně 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; viz následující příklad, kde získáte další informace o jménech.
- `datatype` je volitelný uživatelem určený datový typ vlastnosti; viz následující příklad pro platné datové typy.
- `value` je uživatelem zadaná hodnota vlastnosti; viz následující odstavce, kde získáte další informace o hodnotách.
- Mezery jsou značné mezi znakem > , který předchází hodnotu, a znak < , který následuje za hodnotou, a alespoň jedna mezera musí předcházet `dt=`. Elsewhere blanks can be coded frely between tags, or preceding or following tags (for example, in order to improve čitelnosti); these blanks are not significant.

Pokud se vlastnosti navzájem souvisí, mohou být seskupeny tak, že je uzavřete do počáteční a koncové značky XML se stejným názvem jako skupina:

```
<folder> <group> property1 property2 ... </group> </folder>
```

Skupiny mohou být vnořeny do jiných skupin, a to bez omezení, a skupina se může v rámci složky vyskytnout více než jednou. Je také platný pro složku, která má obsahovat některé vlastnosti ve skupinách a jiné vlastnosti, které nejsou ve skupinách.

Názvy vlastností, skupin a složek: Názvy vlastností, skupin a složek musí být platné názvy značek XML s výjimkou dvojtečkového znaku, který není povolen ve vlastnosti, názvu skupiny nebo složce. Konkrétně se jedná o následující podmínky:

- Názvy musí začínat písmenem nebo podtržítkem. Platná písmena jsou definována ve specifikaci XML W3C a jsou v podstatě kategorií Unicode Ll, Lu, Lo, Lt, Lt a Nl.
- Zbývající znaky v názvu mohou být písmena, desetinná místa, podtržítka, spojovníky nebo tečky. Tyto 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 ve všech směsích malých nebo velkých písmen.

Kromě toho:

- Názvy jsou citlivé na velikost písmen. Například ABC, abca Abc jsou tři různé názvy.
- Každá složka má samostatný obor názvů. Výsledkem je, že skupina nebo vlastnost v jedné složce nekoliduje se skupinou nebo vlastností se stejným názvem v jiné složce.
- Skupiny a vlastnosti zaujímají v rámci složky stejný obor názvů. Výsledkem je, že vlastnost nemůže mít stejný název jako skupina v rámci složky 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 formovány.

Datové typy vlastností: Každá vlastnost může mít volitelný datový typ. Je-li uveden, musí být datový typ jednou z následujících hodnot, v horním, dolním nebo smíšeném případě:

Tabulka 722. Typy dat a jejich použití

Datový typ	**Použití**
`string`	Libovolná posloupnost znaků. Některé znaky musí být určeny pomocí esc sekvencí.
`boolean`	Znak 0 nebo 1 (1 označuje TRUE).
`bin.hex`	Hexadecimální číslice představující okte
`i1`	Celé číslo v rozsahu od 128 do +127, vyjádřené pomocí pouze dekadických číslic a volitelného znaménka.
`i2`	Celé číslo v rozsahu -32 768 až +32 767, vyjádřené pomocí pouze dekadických číslic a volitelného znaménka.
`i4`	Celé číslo v rozsahu -2 147 483 648 až + 2 147 483 647, vyjádřené pomocí pouze dekadických číslic a volitelného znaménka.
`i8`	Celé číslo v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, vyjádřené pouze desítkovými číslicemi a nepovinným znaménkem.
`int`	Celé číslo v rozsahu -9 223 372 036 854 775 808 až + 9 223 372 036 854 775 807, vyjádřené pouze desítkovými číslicemi a nepovinným znaménkem. To lze použít místo `i1`, `i2`, `i4`nebo `i8` , pokud odesilatel nechce implikovat konkrétní přesnost.
`r4`	Číslo s plovoucí řádovou čárkou s velikostí v rozsahu 1.175E-37 až 3.402 823 47E+38, vyjádřené pomocí desetinných číslic, volitelného znaménka, volitelných zlomkových číslic a volitelného exponentu.
`r8`	Plovoucí řádové číslo s rozsahem v rozsahu 2.225E-307 až 1.797 693 134 862 3E+308 vyjádřené pomocí desetinných číslic, volitelného znaku, nepovinných zlomkových číslic a volitelného exponentu.

Hodnoty vlastností: Hodnota vlastnosti se může skládat z libovolných znaků, s výjimkou speciálních znaků, které mají povinnou asociovanou řídicí posloupnost. Každý výskyt v hodnotě znaku, který je označen jako "povinný" v následující tabulce, musí být nahrazen odpovídající řídicí posloupností.

Tabulka také obsahuje znaky, které mají volitelnou řídicí posloupnost znaků změny významu. Každý výskyt v hodnotě znaku, který je označen jako "volitelný", může být nahrazen odpovídající řídicí posloupností, ale toto není povinné.

Tabulka 723. Escaped characters and their usage

Znak	Posloupnost Escape	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 escape sekvence:

```
<Famous_Words>The program displayed "Hello World"</Famous_Words>
```

RF2NVL (10ciferné celé číslo se znaménkem)

Délka *RF2NVD*.

Určuje délku dat v poli *RF2NVD* v bajtech. Chcete-li se vyvarovat problémů s převodem dat z dat (pokud existuje), která následuje za polem *RF2NVD* , *RF2NVL* by mělo být násobkem čtyř.

Poznámka: Pole *RF2NVL* a *RF2NVD* jsou volitelná, ale pokud jsou přítomna, musí se objevit jako pár a být sousedící. Dvojice polí se mohou opakovat tolikrát, kolikrát je třeba, například:

```
length1 data1 length2 data2 length3 data3
```

Protože tato pole jsou volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

RF2SID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RFSIDV
Identifikátor pro pravidla a formátování struktury záhlaví.

Počáteční hodnota tohoto pole je RFSIDV.

RF2VER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RFVER2
Version-2 pravidla a formátování struktury záhlaví.

Počáteční hodnota tohoto pole je RFVER2.

Počáteční hodnoty

Tabulka 724. Počáteční hodnoty polí v 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 (Referenční záhlaví zprávy) v systému IBM i

Struktura MQRMH definuje formát záhlaví referenční zprávy.

Přehled

Účel: Toto záhlaví se používá spolu s uživatelskými ukončovacími programy zpráv napsaných uživatelem k odeslání velkých objemů dat (s názvem "hromadná data"). z jednoho správce front do jiného. Rozdíl v porovnání s normálním systémem zpráv spočívá v tom, že hromadná data nejsou uložena ve frontě; místo toho se do fronty ukládá pouze *odkaz* na data hromadného ukládání. Tím se sníží možnost vyčerpání prostředků produktu IBM MQ několika velkými zprávami.

Jméno formátu: FMRMH.

Znaková sada a kódování: Znaková data v MQRMH a řetězce adresované poli offsetu musí být ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front **CodedCharSetId** . Numerická

data v MQRMH musí být v nativním kódování počítače; to je dáno hodnotou ENNAT pro programovací jazyk C.

Znaková sada a kódování MQRMH musí být nastaveny na pole *MDCSI* a *MDENC* v:

- MQMD (je-li struktura MQRMH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQRMH (všechny ostatní případy).

Použití: Aplikace vloží zprávu sestávající z MQRMH, ale vynechává hromadná data. Je-li zpráva přečtena z přenosové fronty agentem kanálu zpráv (MCA), vyvolá se uživatelská procedura pro zpracování zpráv ke zpracování záhlaví zprávy odkazu. Uživatelská procedura se může připojit k odkazové zprávě o hromadných datech identifikovaných strukturou MQRMH, než agent MCA odešle zprávu prostřednictvím kanálu do dalšího správce front.

Na přijímajícím 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 následuje za MQRMH ve zprávě, a pak předá referenční zprávu bez hromadných dat. Referenční zpráva může být později načtena aplikací, která čte referenční zprávu (bez hromadných dat) z fronty.

Obvykle je struktura MQRMH ve zprávě vše, co je ve zprávě. Je-li však zpráva v přenosové frontě, jedna nebo více dalších záhlaví bude předcházet struktuře MQRMH.

Referenční zpráva může být také odeslána do rozdělovníku. V tomto případě struktura MQDH a její související záznamy předcházejí struktuře MQRMH, když se zpráva nachází v přenosové frontě.

Poznámka: Referenční zpráva by neměla být odeslána jako segmentovaná zpráva, protože uživatelská procedura nemůže správně zpracovat.

- "Převod dat" na stránce 1188
- "Pole" na stránce 1188
- "Počáteční hodnoty" na stránce 1192
- "Deklarace RPG" na stránce 1194

Převod dat

Pro účely konverze dat zahrnuje konverze struktury MQRMH převod dat zdrojového prostředí, název zdrojového objektu, data cílového prostředí a název cílového objektu. Všechny ostatní bajty v rámci *RMLEN* bajtů na 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 se nacházejí ve zprávě, když se provádí konverze dat.
- Pole *RMFMT* v MQRMH má jinou hodnotu než FMNONE.
- Uživatelem zapsaná uživatelská procedura pro převod dat existuje s uvedeným názvem formátu.

Uvědomte si však, že obvykle nejsou hromadná data přítomná ve zprávě, když je zpráva ve frontě, a že hromadné údaje nebudou konvertovány volbou GMCONV.

Pole

Struktura MQRMH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RMCSI (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady hromadných dat.

Tato hodnota určuje identifikátor znakové sady pro hromadný data. Nevztahuje se na znaková data v samotné struktuře MQRMH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

RMDEL (10ciferné celé číslo se znaménkem)

Délka dat cílového prostředí.

Je-li toto pole nula, nejsou k dispozici žádná data cílového prostředí a *RMDEO* je ignorován.

RMDEO (10ciferné celé číslo se znaménkem)

Odsazení dat cílového prostředí.

Toto pole určuje posun dat cílového prostředí ze začátku struktury MQRMH. Data cílového prostředí mohou být uvedena tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Například, data cílového prostředí mohou být cesta k adresáři objektu, kam se mají hromadná data uložit. Pokud však tvůrce neznáme data cílového prostředí, je zodpovědností uživatelského ukončovacího programu pro zprávy, aby určil, že jsou potřebné informace o prostředí.

Délka dat cílového prostředí je dána produktem *RMDEL* ; je-li tato délka nula, nejsou žádná data cílového prostředí a *RMDEO* je ignorován. Je-li tento parametr zadán, musí být data cílového prostředí plně umístěna v rozmezí *RMLEN* bajtů od začátku struktury.

Aplikace by neměly předpokládat, že data cílového prostředí sousedí s libovolní z dat řešených poli *RMSEO*, *RMSNO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMDL (10ciferné celé číslo se znaménkem)

Délka hromadných dat.

Pole *RMDL* určuje délku hromadného dat, na kterou odkazuje struktura MQRMH.

Pokud se hromadná data nacházejí ve zprávě, data začínají na posunutí *RMLEN* bajtů od začátku struktury MQRMH. Délka celé zprávy minus *RMLEN* udává délku hromadného datového souboru.

Pokud jsou data přítomna ve zprávě, *RMDL* uvádí množství dat, která jsou relevantní. Normální případ je určen pro *RMDL* , aby měl stejnou hodnotu jako délka dat přítomných ve zprávě.

Pokud struktura MQRMH představuje zbývající data v objektu (počínaje určeným logickým posunutím), lze pro *RMDL*použít nulovou hodnotu, pokud se hromadná data ve zprávě nezobrazí.

Nejsou-li k dispozici žádná data, je konec MQRMH totožný s koncem zprávy.

Počáteční hodnota tohoto pole je 0.

RMDNL (10číslicové podepsané celé číslo)

Délka názvu cílového objektu.

Je-li toto pole nula, neexistuje žádný název cílového objektu a *RMDNO* je ignorován.

RMDNO (10číslicové podepsané celé číslo)

Offset názvu cílového objektu.

Toto pole určuje posun názvu cílového objektu od začátku struktury MQRMH. Jméno cílového objektu může být zadáno tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Pokud však tvůrce nezná název cílového objektu, zodpovídá za identifikaci objektu, který má být vytvořen nebo upraven, je zodpovědný za uživatelskou proceduru zprávy.

Délka názvu cílového objektu je dána *RMDNL* ; je-li tato délka nula, neexistuje žádný název cílového objektu a *RMDNO* je ignorován. Je-li tento parametr zadán, musí být název cílového objektu zcela umístěn v rozmezí *RMLEN* bajtů od začátku struktury.

Aplikace by neměly předpokládat, že název cílového objektu je souvislý s libovolní z dat adresovaných poli *RMSEO*, *RMSNO*a *RMDEO* .

Počáteční hodnota tohoto pole je 0.

RMDO (10číslicové podepsané celé číslo)

Dolní posun hromadných dat.

Toto pole určuje dolní posun dat hromadného objektu od začátku objektu, jehož součástí jsou hromadné datové formuláře. Posunutí hromadných dat od začátku objektu se nazývá *logický posun*. Toto není fyzický posun hromadných dat od začátku struktury MQRMH-tento posun je dán parametrem *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 offsetu o 1 000 000 000. Je to tedy hodnota v rozsahu od 0 do 999 999 999.
- *RMDO2* představuje výsledek, který se získá, když je logický offset rozdělen do 1 000 000 000. Jedná se tedy o počet úplných násobků 1 000 000 000, které existují v logickém posunu. Počet násobků je v rozsahu od 0 do 999 999 999.

Počáteční hodnota tohoto pole je 0.

RMDO2 (10ciferné celé číslo se znaménkem)

Vysoký posun hromadných dat.

Toto pole určuje horní posun hromadných dat od začátku objektu, jehož součástí jsou hromadné datové formuláře. Je to hodnota v rozsahu od 0 do 999 999 999. Podrobnosti viz *RMDO* .

Počáteční hodnota tohoto pole je 0.

RMENC (10ciferné celé číslo se znaménkem)

Numerické kódování hromadných dat.

Určuje číselné kódování hromadných dat. Nevztahuje se na číselná data v samotné struktuře MQRMH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je ENNAT.

RMFLG (10ciferné celé číslo se znaménkem)

Příznaky referenční zprávy.

Jsou definovány následující příznaky:

RMLAST
Referenční zpráva obsahuje nebo reprezentuje poslední část objektu.

Tento příznak označuje, že referenční zpráva představuje nebo obsahuje poslední část odkazovaného objektu.

RMNLST
Referenční zpráva neobsahuje nebo nereprezentuje poslední část objektu.

Položka RMNLST je definována pro dokumentaci programu podpory. Není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze detekovat.

Počáteční hodnota tohoto pole je RMNLST.

RMFMT (8bajtový znakový řetězec)

Název formátu hromadných dat.

Uvádí název formátu hromadných dat.

Na základě 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.

Počáteční hodnota tohoto pole je FMNONE.

RMLEN (10ciferné celé číslo se znaménkem)

Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadná data.

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:

OIRONON
Není 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 ukončovacím programem zpráv k rozpoznání typů referenční zprávy, které podporuje. Zvažte, zda je název shodný se stejnými pravidly jako pole *RMFMT* .

Počáteční hodnota tohoto pole je 8 mezer.

RMSEL (10ciferné celé číslo se znaménkem)

Délka dat o zdrojovém prostředí.

Je-li toto pole nula, nejsou žádná data o zdrojovém prostředí a *RMSEO* je ignorován.

Počáteční hodnota tohoto pole je 0.

RMSEO (10ciferné celé číslo se znaménkem)

Odsazení dat o zdrojovém prostředí.

Toto pole určuje posun dat o zdrojovém prostředí ze začátku struktury MQRMH. Data o zdrojovém prostředí mohou být určena tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Například, data zdrojového prostředí mohou představovat cestu k adresáři objektu, který obsahuje hromadná data. Pokud však tvůrce neznáme data o zdrojovém prostředí, je zodpovědností uživatelského ukončovacího programu pro zprávy, aby určil potřebné informace o prostředí.

Délka dat zdrojového prostředí je dána *RMSEL* ; Pokud je tato délka nula, nejsou žádná data o zdrojovém prostředí a *RMSEO* je ignorován. Je-li tato možnost přítomna, musí se zdrojová data prostředí zcela nacházet v rozmezí *RMLEN* bajtů od začátku struktury.

Aplikace by neměly předpokládat, že data prostředí začínají bezprostředně po posledním pevném poli ve struktuře nebo že je sousedící s libovolnými daty adresovaným poli *RMSNO*, *RMDEO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

RMSIDV
Identifikátor struktury záhlaví zprávy odkazu.

Počáteční hodnota tohoto pole je RMSIDV.

RMSNL (10číslicové podepsané celé číslo)

Délka názvu zdrojového objektu.

Je-li toto pole nula, neexistuje žádné jméno zdrojového objektu a *RMSNO* se ignoruje.

Počáteční hodnota tohoto pole je 0.

RMSNO (10číslicové podepsané celé číslo)

Offset názvu zdrojového objektu.

Toto pole určuje posun názvu zdrojového objektu od začátku struktury MQRMH. Jméno zdrojového objektu může být zadáno tvůrcem referenční zprávy, pokud je tato data známá tvůrci. Pokud však tvůrce nezná název zdrojového objektu, zodpovídá za identifikaci objektu, ke kterému má být proveden přístup, uživatelem dodaným výstupem zpráv.

Délka názvu zdrojového objektu je dána *RMSNL* ; je-li tato délka nula, neexistuje žádné jméno zdrojového objektu a *RMSNO* je ignorován. Je-li tento název zadán, musí být název zdrojového objektu zcela umístěn v rozmezí *RMLEN* bajtů od začátku struktury.

Aplikace by neměly předpokládat, že název zdrojového objektu je souvislý s libovolní z dat adresovaných poli *RMSEO*, *RMDEO*a *RMDNO* .

Počáteční hodnota tohoto pole je 0.

RMVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

RMVER1
Struktura záhlaví referenční zprávy Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

RMVERC
Aktuální verze struktury záhlaví zprávy odkazu.

Počáteční hodnota tohoto pole je RMVER1.

Počáteční hodnoty

Tabulka 725. Počáteční hodnoty polí 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	OIRONON	Hodnoty null

Tabulka 725. Počáteční hodnoty polí v MQRMH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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                                     00000000000000000000-
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 (Response record) na systému IBM i

Struktura MQRR se používá k přijetí kódu dokončení a kódu příčiny, který je výsledkem operace otevření nebo vložení pro jednu cílovou frontu, je-li 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 poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT. Je-li však aplikace spuštěna jako klient produktu IBM MQ , musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur na voláních MQOPEN a MQPUT nebo na 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 u ostatních selže. Kód příčiny RC2136 z volání označuje, že správce front nastavil záznamy odpovědí (je-li to poskytnuto aplikací).

- "Pole" na stránce 1194
- "Počáteční hodnoty" na stránce 1194
- "Deklarace RPG" na stránce 1195

Pole

Struktura MQRR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

RRCC (10ciferné celé číslo se znaménkem)

Kód dokončení pro frontu.

Jedná se o kód dokončení, který je výsledkem operace otevření nebo vložení pro frontu s názvem, který byl určen příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je CCOK.

RRREA (10ciferné 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 příslušným prvkem v poli struktur MQOR zadaných v rámci volání MQOPEN nebo MQPUT1 .

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je RCNONE.

Počáteční hodnoty

Tabulka 726. Počáteční hodnoty polí v objektu MQRR

Název pole	Název konstanty	Hodnota konstanty
RRCC	KEK	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 (konfigurace TLS) v systému IBM i

Struktura MQSCO (s poli TLS ve struktuře MQCD) umožňuje aplikaci spuštěnou jako IBM MQ MQI client určit volby konfigurace, které řídí použití TLS pro připojení klienta, je-li protokol kanálu TCP/IP.

Přehled

Účel: Struktura je vstupním parametrem volání MQCONNX.

Pokud není protokol kanálu pro kanál klienta TCP/IP, bude struktura MQSCO ignorována.

Znaková sada a kódování: Data v MQSCO musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého ENNAT.

- "Pole" na stránce 1195
- "Počáteční hodnoty" na stránce 1199
- "Deklarace RPG" na stránce 1199

Pole

Struktura MQSCO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SCAIC (10ciferné celé číslo se znaménkem)

Jedná se o počet záznamů ověřovacích informací (MQAIR) adresovaných poli *SCAIP* nebo *SCAIO* . Další informace viz"MQAIR (záznam ověřovacích informací) v systému IBM i" na stránce 1005. Hodnota musí být nula nebo větší. Není-li hodnota platná, volání selže s kódem příčiny RC2383.

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

SCIO (10ciferné celé číslo se znaménkem)

Jedná se o posun v bajtech prvního záznamu ověřovacích informací od začátku struktury MQSCO. Odsazení může být kladné nebo záporné. Pole je ignorováno, pokud *SCAIC* je nula.

K zadání záznamů MQAIR můžete použít buď *SCAIO* nebo *SCAIP* , ale ne obojí; podrobnosti najdete v popisu pole *SCAIP* .

Toto je vstupní pole. Počáteční hodnota tohoto pole je 0.

SCAIP (10číslicové podepsané celé číslo)

Toto je adresa prvního záznamu ověřovacích informací. Pole je ignorováno, pokud *SCAIC* je nula.

Pole záznamů MQAIR můžete zadat jedním ze dvou způsobů:

- Pomocí pole ukazatele *SCAIP*

  V takovém případě může aplikace deklarovat pole záznamů MQAIR, které jsou odděleny od struktury MQSCO, a nastavit proměnnou *SCAIP* na adresu pole.

  Zvažte použití *SCAIP* pro programovací jazyky, které podporují datový typ ukazatele v módě, který je přenosný do různých prostředí (například programovací jazyk C).
- Použití pole offsetu *SCAIO*

V takovém případě musí aplikace deklarovat složenou strukturu obsahující MQSCO, za kterou následuje pole záznamů MQAIR, a nastavit proměnnou *SCAIO* na hodnotu offsetu prvního záznamu v poli od začátku struktury MQSCO. Ujistěte se, že je tato hodnota správná a že má hodnotu, která může být umístěna v rámci MQLONG (nejvíce omezující programovací jazyk je COBOL, pro který je platný rozsah -999 999 999 až +999 999 999).

Zvažte použití *SCAIO* pro programovací jazyky, které nepodporují datový typ ukazatele, nebo které implementují datový typ ukazatele v módě, který není přenosný do různých prostředí (například programovací jazyk COBOL).

Zvolená technika může být použita pouze jedním 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 těchto programovacích jazycích, které podporují ukazatele, a jinak řetězec bajtů se všemi bajty null.

Poznámka: Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

SCCERLBL (10ciferné celé číslo se znaménkem)

Toto pole uvádí podrobnosti o použité návěští 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 je hodnota tohoto pole interpretována za běhu a v případě potřeby je změněna. Určíte-li verzi produktu MQSCO nižší než 5.0nebo přijmete výchozí hodnotu mezer pro pole SCCERLBL, systém použije hodnotu `ibmwebspheremq`*`user_id`*.

Toto je vstupní pole.

SCCERTVPOL (10ciferné celé číslo se znaménkem)

Toto pole uvádí, jaký typ zásady ověření certifikátu se použije. Pole může být nastaveno na jednu z následujících hodnot:

MQ_CERT_VAL_POLICY_ANY
Použít všechny zásady ověření platnosti certifikátů podporované knihovnou SSL (Secure Sockets Layer). Přijměte řetěz certifikátů, pokud některý ze zásad považuje řetězec certifikátů za platný.

MQ_CERT_VAL_POLICY_RFC5280
Použijte pouze zásadu ověření certifikátu vyhovujícího standardu RFC5280 . Toto nastavení poskytuje přísnější validaci než nastavení ANY, ale odmítá některé starší digitální certifikáty.

Počáteční hodnota tohoto pole je MQ_CERT_VAL_POLICY_ANY.

SCCH (10ciferné celé číslo se znaménkem)

Toto pole poskytuje podrobnosti konfigurace kryptografického hardwaru připojeného k systému klienta.

Nastavte pole na řetězec v následujícím formátu, nebo ponechte prázdné 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, uveďte cestu k ovladači PKCS11 , návěští tokenu PKCS11 a řetězce hesel tokenu PKCS11 , přičemž každý z nich bude zakončen 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
```

Návěští tokenu PKCS #11 musí být zcela v malých písmenech. Pokud jste konfigurovali hardware se smíšeným nebo velkými písmeny na štítku s velkými písmeny, překonfigurujte jej pomocí tohoto malého popisku.

Není-li požadována žádná konfigurace kryptografického hardwaru, nastavte pole na prázdné nebo null.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Pokud hodnota není platná, nebo vede k selhání při konfiguraci kryptografického hardwaru, volání selže s kódem příčiny RC2382.

Toto je vstupní pole. Délka tohoto pole je dána LNSSCH. Počáteční hodnota tohoto pole obsahuje prázdné znaky.

SCEPSUITEB (10ciferné celé číslo se znaménkem)
Toto pole Uvádí, zda se použije šifrování vyhovující Suite B a jaká úroveň síly je použita. Hodnota může být jedna nebo více hodnot:

- SCEPSUITEB0

  Šifrování kompatibilní se sadou Suite B se nepoužívá.

- SCEPSUITEB1

  Používá se zabezpečení odolnosti standardu Suite B 128 bitů.

- SCEPSUITEB2

  Je použito 192bitové zabezpečení pevnosti sady Suite B.

Poznámka: Použití SCEPSUITEB0 s jakoukoli jinou hodnotou v tomto poli je neplatné.

SCFR (10číslicové celé číslo se znaménkem)

Produkt IBM MQ lze konfigurovat pomocí kryptografického hardwaru tak, aby použité kryptografické moduly byly ty, které jsou poskytovány hardwarovým produktem; tyto mohou být FIPS certifikovány na určitou úroveň v závislosti na používaném šifrovacím hardwaru produktu.

Prostřednictvím tohoto pole lze určit, že se budou používat pouze algoritmy certifikované podle standardu FIPS, je-li šifrování poskytnuto v softwaru poskytovaného softwaru IBM MQ.

Je-li nainstalován produkt IBM MQ , instaluje se také implementace šifrování TLS, která poskytuje některé moduly certifikované FIPS.

Hodnoty mohou být:

MQSSL_FIPS_NO
Toto je výchozí hodnota. Při nastavení na tuto hodnotu:

- Lze použít jakoukoli CipherSpec podporovanou na konkrétní platformě.
- Pokud se spustí bez použití kryptografického hardwaru, spustí se následující CipherSpecs 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 této hodnoty, pokud nepoužíváte kryptografický hardware k provedení šifrování, si můžete být jisti, že

- Ve specifikaci CipherSpec pro toto připojení klienta lze použít pouze šifrovací algoritmy s certifikací FIPS.

- Příchozí a odchozí připojení kanálu TLS jsou úspěšná pouze v případě, že se použije jedna z následujících specifikací šifer:
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

Notes:

1. CipherSpec TLS_RSA_WITH_3DES_EDE_CBC_SHA byla zamítnuta.
2. Je-li to možné, v případě, že je konfigurován pouze standard FIPS CipherSpecs , pak klient MQI odmítne připojení, která určují neFIPS CipherSpec withRC2393. Produkt IBM MQ nezaručuje odmítnutí všech takových připojení a je vaší odpovědností určit, zda je vaše konfigurace produktu IBM MQ kompatibilní se standardem FIPS-.

SCKR (10ciferné celé číslo se znaménkem)

Toto pole je relevantní pouze pro produkt IBM MQ MQI clients spuštěný na systémech AIX, Linux, and Windows . Určuje umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty. Soubor databáze klíčů musí mít název souboru ve tvaru `zzz.kdb`, kde `zzz` je vybratelný uživatelem. Pole *SCKR* obsahuje cestu k tomuto souboru, spolu s názvem souboru stem (všechny znaky v názvu souboru, ale ne včetně konečné `.kdb`). Přípona souboru `.kdb` se přidá automaticky.

Ke každému souboru databáze klíčů je přidružen *soubor stash hesel*. Zašifruje zašifrovaná hesla, která umožňují programový přístup k databázi klíčů. Soubor pro uložení hesla se musí nacházet ve stejném adresáři a musí mít stejný soubor jako databáze klíčů a musí končit příponou `.sth`.

Pokud má pole *SCKR* například hodnotu `/xxx/yyy/key`, musí být soubor databáze klíčů `/xxx/yyy/key.kdb`a soubor pro uložení hesla musí být `/xxx/yyy/key.sth`, kde `xxx` a `yyy` představují názvy adresářů.

Je-li hodnota kratší než délka pole, ukončete ji znakem null nebo jej odblood mezerami až do délky pole. Hodnota není kontrolována; pokud došlo k chybě při přístupu k úložišti klíčů, volání selže s kódem příčiny RC2381.

Chcete-li spustit TLS připojení 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 (10ciferné celé číslo se znaménkem)

Jedná se o identifikátor struktury; hodnota musí být:

SCSIDV
Identifikátor struktury voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SCSIDV.

SCVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

SCVER1
Struktura konfiguračních voleb TLS Version-1 TLS.

SCVER2
Struktura konfiguračních voleb Version-2 TLS.

Následující konstanta uvádí číslo verze aktuální verze:

SCVERC
Aktuální verze struktury voleb konfigurace TLS.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SCVER2 .

Počáteční hodnoty

Tabulka 727. Počáteční hodnoty polí v MQSCO		
Název pole	**Název konstanty**	**Hodnota konstanty**
SCSID	SCSIDV	'SCO¬'
SCVER	SCVER5	1
SCKR	Není	Nulový řetězec nebo prázdné znaky
SCCH	Není	Nulový řetězec nebo prázdné znaky
SCAIC	Není	0
SCAIO	Není	0
SCAIP	Není	Nulový ukazatel nebo bajty null
SCKRC	Není	Nulový ukazatel nebo bajty null
SCFR	Není	Nulový ukazatel nebo bajty null
SCEPSUITEB	Není	Nulový ukazatel nebo bajty null
SCCERTVPOL	Není	Nulový ukazatel nebo bajty null
SCCERLBL	Není	Nulový ukazatel nebo bajty null

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Volby SCEPSUITEB viz"Deklarace RPG" na stránce 1199.

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 1200
- “Počáteční hodnoty” na stránce 1212
- “Deklarace RPG” na stránce 1213

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 SOSETI , 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 SOSETI , 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 SOSETI , 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 SOALT , můžete změnit hodnotu *MDACC* v budoucích zprávách publikování.

Při návratu z volání MQSUB s použitím volby SORESje 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 SOALTU a pole *SDAU* není zcela prázdné až do prvního znaku null nebo konce pole.

Při návratu z volání MQSUB s použitím volby SORESje toto pole nezměněno.

Další informace viz popis ODASI v datovém typu MQOD.

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 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 S0RES 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:

- S0ALT
- S0RES
- S0CRT

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 728. 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.
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 SOAUID , 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 SOALT.

Tabulka 729. 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
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 730. 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 731. 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 S0RES 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 00SETI, s výjimkou případu, kdy je také použita volba S0MAN . 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 732. 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 S0CRE a S0ALT. Pokud se používá s S0RES, 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 S0ALT , pak *SDSL* nelze změnit.

SDSN (MQCHARV)

SDSN určuje název odběru.

Toto pole je povinné pouze v případě, že parametr *SDOPT* uvádí volbu S0DUR , ale pokud je poskytnuta, je také použita správcem front pro S0NDUR . 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 C0KPSB), nebo jste byli odpojeni od správce front. Identifikace odběru, který má být odebrán po jeho vytvoření, se provádí pomocí volání MQSUB s volbou S0RES . 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 733. Počáteční hodnoty polí 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
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	

Tabulka 733. Počáteční hodnoty polí v MQSD (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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 (Nastavit volby vlastností zprávy) v systému IBM i

Struktura **MQSMPO** umožňuje aplikacím určovat volby, které řídí způsob nastavení vlastností zpráv.

Přehled

Účel: Struktura je vstupním parametrem na volání **MQSETMP** .

Znaková sada a kódování: Data v souboru **MQSMPO** musí být ve znakové sadě aplikace a kódování aplikace (ENNAT).

- "Pole" na stránce 1214
- "Počáteční hodnoty" na stránce 1215
- "Deklarace RPG" na stránce 1215

Pole

Struktura MQSMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SPOPT (10číslicové celé číslo se znaménkem)

Volby umístění: Následující volby se vztahují k relativnímu umístění vlastnosti v porovnání s kurzorem vlastnosti:

SPSETZ
Nastaví hodnotu první vlastnosti, která odpovídá zadanému názvu, nebo pokud neexistuje, přidá novou vlastnost za všechny ostatní vlastnosti s odpovídající hierarchií.

SPSETC
Nastaví hodnotu vlastnosti, na kterou ukazuje kurzor vlastností. Vlastnost, na kterou se odkazuje kurzor vlastností, je ta, která byla naposledy dotazovaná pomocí volby IPINQF nebo IPINQN.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy, nebo když je popisovač zprávy zadán v poli *HMSG* struktury MQGMO na volání MQGET nebo MQPMO struktury MQPUT na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud ustavován nebo pokud byla vlastnost, na kterou ukazuje kurzor vlastností, odstraněna, volání selže s kódem dokončení CCFAIL a kódem příčiny RC2471.

SPSETA
Nastaví novou vlastnost za vlastnost, na kterou ukazuje vlastnost kurzor vlastností. Vlastnost, na kterou se odkazuje kurzor vlastností, je ta, která byla naposledy dotazovaná pomocí volby IPINQF nebo IPINQO.

Kurzor vlastností se resetuje, když se znovu použije popisovač zprávy, nebo když je popisovač zprávy zadán v poli *HMSG* struktury MQGMO na volání MQGET nebo MQPMO struktury MQPUT na volání MQPUT.

Je-li tato volba použita, nebyla-li kurzor vlastnosti dosud ustavován nebo pokud byla vlastnost, na kterou ukazuje kurzor vlastností, odstraněna, volání selže s kódem dokončení CCFAIL a kódem příčiny RC2471.

Pokud nepotřebujete žádné z popsaných voleb, použijte následující volbu:

NESPAČNÝ
Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SPSETF.

SPSID (10ciferné celé číslo se znaménkem)

Jedná se o 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 (10ciferné celé číslo se znaménkem)

Znaková sada hodnoty vlastnosti, která má být nastavena, je-li hodnota znakový řetězec.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **CSAPL**.

SPVALENC (10číslicové celé číslo se znaménkem)

Kódování hodnoty vlastnosti, která má být nastavena, je-li hodnota číselná.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **ENNAT**.

SPVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

SPVER1
Version-1 -nastavení struktury voleb vlastností zprávy.

Následující konstanta uvádí číslo verze aktuální verze:

SPVERC
Aktuální verze struktury voleb vlastností sady zpráv.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **SPVER1**.

Počáteční hodnoty

Tabulka 734. Počáteční hodnoty polí v MQSMPO

Název pole	Název konstanty	Hodnota konstanty
SPSID	SPSIDV	'SMPO'
SPVER	SPVER1	1
SPOPT	NESPAČNÝ	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) v systému IBM i

Struktura MQSRO umožňuje aplikaci určit volby, které řídí způsob provedení požadavku na odběr.

Přehled

Účel: Struktura je vstupní/výstupní parametr v volání MQSUBRQ.

Verze: Aktuální verze MQSRO je SRVER1.

- "Pole" na stránce 1216
- "Počáteční hodnoty" na stránce 1217
- "Deklarace RPG" na stránce 1217

Pole

Struktura MQSRO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

SRNMP (10ciferné celé číslo se znaménkem)

Jedná se o výstupní pole, které se vrátí do aplikace a označuje počet publikování odeslaných do fronty odběru jako výsledek tohoto volání. Přestože byl tento počet publikací odeslán jako výsledek tohoto volání, není zaručeno, že bude pro aplikaci k dispozici mnoho zpráv, zvláště pokud jde o netrvalé zprávy.

Pokud téma přihlášené k odběru obsahovalo zástupný znak, může existovat více než jedno publikování. Pokud nebyly nalezeny žádné zástupné znaky v řetězci tématu, když byl vytvořen odběr představovaný produktem *HSUB* , pak je v důsledku tohoto volání odeslán nanejvýš jedna publikace.

SROPT (10ciferné celé číslo se znaménkem)

Musí být uvedena jedna z následujících voleb. Může být uvedena pouze jedna volba.

Další volby: Následující volba určuje, co se stane, když je správce front uváděn do klidového stavu:

SR.FIQ

Volání MQSUBRQ se nezdaří, je-li správce front ve stavu uvedení do klidového stavu.

Výchozí volba: Pokud dříve popsaná volba není povinná, je třeba použít následující volbu:

SRANONE

Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

SRNONE pomáhá programovou dokumentaci. Ačkoli se nejedná o zamýšlené použití této volby, protože její hodnota je nulová, nelze toto použití detekovat.

SRSID (čtyřbajtový znakový řetězec)

Jedná se o identifikátor struktury; hodnota musí být:

SRSIDV
Identifikátor struktury SROPT požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SRSIDV.

SRVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury; hodnota musí být:

SRVER1
Version-1 Struktura voleb požadavku na odběr.

Následující konstanta uvádí číslo verze aktuální verze:

SRVERC
Aktuální verze struktury Volby požadavku na odběr.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je SRVER1.

Počáteční hodnoty

Tabulka 735. Počáteční hodnoty polí v MQSRO		
Název pole	**Název konstanty**	**Hodnota konstanty**
SRSID	SRSIDV	'SRO¬'
SRVER	SRVER1	1
SROPT	SRANONE	0
SRNMP	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v 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 (Status reporting structure) v systému IBM i

Struktura MQSTS popisuje data ve struktuře stavu vrácené příkazem MQSTAT.

Přehled

Znaková sada a kódování: Znaková data v MQSTS se nacházejí ve znakové sadě lokálního správce front; to je dáno atributem správce front *CodedCharSetId* . Numerická data v MQSTS jsou v nativním kódování počítače; to je dáno *ENNAT*.

Použití: Příkaz MQSTAT se používá k získání informací o stavu. Tyto informace jsou vraceny ve struktuře MQSTS. Informace o příkazu MQSTAT najdete v tématu “MQSTAT (Načtení informací o stavu) v systému IBM i” na stránce 1344.

- “Pole” na stránce 1217
- “Počáteční hodnoty” na stránce 1221
- “Deklarace RPG” na stránce 1221

Pole

Struktura MQSTS obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

STSCC (10číslicové celé číslo se znaménkem)

Jedná se o kód dokončení, který je výsledkem první chyby hlášené ve struktuře MQSTS.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je CCOK.

STSFC (10číslicové podepsané celé číslo)

Jedná se o počet asynchronních volání vložení, která se nezdařila.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0.

STSOBJN (48-bajtový znakový řetězec)

Jedná se o lokální název objektu, který se podílí na prvním selhání.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSOQMGR (48-bajtový znakový řetězec)

Jedná se o název správce front, ve kterém je definován objekt *STSOBJN* . Název, který je zcela prázdný až k prvnímu znaku null nebo konec 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 (10ciferné celé číslo se znaménkem)

Objekt STSOO se používá k otevření objektu, který je hlášen. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Hodnota parametru STSOO závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Nula.

STATREC

Nula.

STASTRER

STSOO použitý, když došlo k selhání. Příčina selhání se vykazuje v polích *STSCC* a *STSRC* ve struktuře MQSTS .

STSOO je výstupní pole. Jeho počáteční hodnota je nula.

STSOS (MQCHARV)

Dlouhý název objektu, u kterého se vykazuje selhávající objekt. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

STSOS je pole MQCHARV s maximální délkou 10240. Popis způsobu použití struktury MQCHARV naleznete v tématu MQCHARV .

Interpretace parametru STSOS 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

STASTRER

Jedná se o dlouhý název objektu objektu, který způsobil selhání opětovného připojení.

STSOS je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

STSOT (10ciferné celé číslo se znaménkem)

Typ objektu, který je pojmenován v produktu *ObjectName*. Možné hodnoty jsou:

OTALSQ
Fronta alias.

OTLOCQ
Lokální fronta.

OTMODQNAME
Modelová fronta.

OTQ
Fronta.

OTŘES
Vzdálená fronta.

OTOPU
.

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je OTQ.

STSRC (10ciferné celé číslo se znaménkem)

Jedná se o kód příčiny, který je výsledkem první chyby hlášené ve struktuře MQSTS

Toto je vždy výstupní pole. Počáteční hodnota tohoto pole je RCNONE.

STSROBJN (48-bajtový znakový řetězec)

Jedná se o název cílové fronty pojmenované v produktu *STSOBJN* poté, co lokální správce front vyřeší daný název. Vrácený název je název fronty, která existuje ve správci front identifikovaném příkazem *STSRQMGR*.

Neprázdná hodnota je vrácena pouze v případě, že objekt je otevřena jediná fronta pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Je-li otevřený objekt jakýkoli z následujících, *STSROBJN* je nastaven na mezery:

- Téma
- Fronta, ale neotevřena pro procházení, vstup nebo výstup

Toto je výstupní pole. Počáteční hodnota tohoto pole je 48 prázdných znaků.

STSRQMGR (48-bajtový znakový řetězec)

Jedná se o název cílového správce front poté, co lokální správce front vyřeší daný název. Vrácený název je název správce front, který vlastní frontu určenou produktem *STSROBJN*. *STSRQMGR* může být název lokálního správce front.

Pokud *STSROBJN* je sdílená fronta, kterou vlastní skupina sdílení front, do níž patří lokální správce front, *STSRQMGR* je název skupiny sdílení front. Pokud je fronta vlastníkem některé jiné skupiny sdílení front, může být produktem *STSROBJN* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (charakter vráceného výsledku je určen definicemi front, které existují v lokálním správci front).

Neprázdná hodnota je vrácena pouze v případě, že objekt je otevřena jediná fronta pro procházení, vstup nebo výstup (nebo libovolnou kombinaci). Je-li otevřený objekt jakýkoli z následujících, *STSRQMGR* je nastaven na mezery:

- Téma
- Fronta, ale neotevřena pro procházení, vstup nebo výstup
- Fronta klastru s uvedeným 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 (10ciferné celé číslo se znaménkem)

Jedná se o počet asynchronních volání vložení, která byla úspěšná.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0.

STSSID (čtyřbajtový znakový řetězec)

Jedná se o identifikátor struktury. Hodnota musí být:

STSSID
Identifikátor struktury vykazování stavu.

Počáteční hodnota tohoto pole je STSSID.

STSSO (10ciferné celé číslo se znaménkem)

STSSO použil k otevření selhávajícího odběru. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

Interpretace parametru STSSO závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Nula.

STATREC

Nula.

STASTRER

STSSO použitý, když došlo k selhání. Příčina selhání se vykazuje v polích *STSCC* a *STSRC* ve struktuře MQSTS . Pokud se 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. Nachází se pouze ve verzi 2 produktu MQSTS nebo vyšší.

STSSUN je pole MQCHARV s délkou maxiumm 10240. Popis způsobu použití struktury MQCHARV naleznete v tématu MQCHARV .

Interpretace parametru STSSUN závisí na hodnotě parametru MQSTAT **STYPE** .

STATAPT

Nulová délka řetězce.

STATREC

Nulová délka řetězce.

STASTRER

Název odběru, který způsobil selhání opětovného připojení. Není-li k dispozici žádný název odběru nebo selhání nesouvisí s odběrem, je to řetězec s nulovou délkou.

STSSUN je výstupní pole. Jeho počáteční hodnota je řetězec s nulovou délkou.

STSVER (10ciferné celé číslo se znaménkem)

Jedná se o číslo verze struktury. Hodnota musí být:

STSVR1
Číslo verze pro strukturu vykazování stavu.

Následující konstanta uvádí číslo verze aktuální verze:

STSVRC
Aktuální verze struktury vykazování stavu.

Počáteční hodnota tohoto pole je STSVR1.

STSWC (10číslicové podepsané celé číslo)

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 736. Počáteční hodnoty polí v MQSTS		
Název pole	**Název konstanty**	**Hodnota konstanty**
STSSID	STSID	
STSVER	STSVRC	STSVR1
STSCC	KEK	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, když se vyskytne událost spouštěče pro frontu.

Přehled

Účel: Tato struktura je součástí produktu IBM MQ Trigger Monitor Interface (TMI), který je jedním z rozhraní rámce IBM MQ .

Název formátu: FMTM.

Znaková sada a kódování: Znaková data ve struktuře MQTM jsou ve znakové sadě správce front, který generuje MQTM. Numerická data v MQTM jsou v kódování počítače správce front, který generuje MQTM.

Znaková sada a kódování MQTM jsou dána poli *MDCSI* a *MDENC* v:

- MQMD (je-li struktura MQTM spuštěna 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 pro monitor spouštěčů. Informace, které mohou být potřebné pro spuštěnou aplikaci, zahrnují *TMQN*, *TMTD*a *TMUD*. Aplikace monitor spouštěčů může předávat strukturu MQTM přímo do spuštěné aplikace, nebo místo toho předat strukturu MQTMC2 , v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci. Informace o příkazu MQTMC2naleznete v tématu “MQTMC2 (Spouštěcí zpráva 2-znakový formát) v systému IBM i” na stránce 1226.

- V systému IBM ipředává aplikace monitoru spouštěčů, která poskytuje IBM MQ , strukturu MQTMC2 do spuštěné aplikace.

Informace o spouštěčích najdete v tématu Nezbytné předpoklady pro spuštění.

MQMD pro zprávu spouštěče

Tabulka 737. Nastavení pro pole v deskriptoru MQMD zprávy spouštěče generované správcem front

Pole v MQMD	Použitá hodnota
MDSID	MDSIDROVSKÁ

Tabulka 737. Nastavení pro pole v deskriptoru MQMD zprávy spouštěče generované správcem front (pokračování)

Pole v MQMD	Použitá hodnota
MDVER	MDVER1
MDREP	RONAN
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	CINNE
MDBOC	0
MDRQ	Mezery
MDRM	Název správce front
MDUID	Mezery
MDACC	ANONE
MDAID	Mezery
MDPAT	ATQM nebo odpovídající pro agenta kanálu zpráv
MDPAN	Prvních 28 bajtů názvu správce front
MDPD	Datum, kdy se odešle zpráva spouštěče
MDPT	Čas odeslání zprávy spouštěče
MDAOD	Mezery

Pro nastavení podobných hodnot se doporučuje použít aplikaci, která vygeneruje zprávu spouštěče, s výjimkou následujících:

- Pole *MDPRI* může být nastaveno na PRQDEF (správce front to změní na výchozí prioritu pro inicializační frontu, když je zpráva vložena).
- Pole *MDRM* může být nastaveno na prázdné místo (správce front to změní na název lokálního správce front, když je zpráva vložena).
- Pole kontextu by měla být nastavena jako odpovídající 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 identifikující aplikaci, která má být spuštěna, a kterou používá aplikace pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole

s hodnotou atributu **ApplId** objektu procesu určeného polem *TMPN* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu v systému IBM i” na stránce 1382 . Obsah těchto dat nemá význam pro správce front.

Význam *TMAI* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný serverem IBM MQ vyžaduje, aby byl *TMAI* název spustitelného programu.

Délka tohoto pole je dána LNPROA. Počáteční hodnota tohoto pole je 256 prázdných znaků.

TMAT (10číslicové podepsané celé číslo)

Typ aplikace.

Identifikuje charakter programu, který má být spuštěn, a je použit aplikací monitor spouštěčů, která přijme zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ApplType** objektu procesu určeného polem *TMPN* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu v systému IBM i” na stránce 1382 . Obsah těchto dat nemá význam pro správce front.

TMAT může mít jednu z následujících standardních hodnot. Mohou být také použity uživatelsky definované typy, ale měly by být omezeny na hodnoty v rozsahu ATUFST přes ATULST:

rovnoCICS
CICS .

ATVSE
CICS/VSE .

AT400
IBM i .

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULSTCITY
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 týkající se aplikace, která má být spuštěna, a kterou používá aplikace pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **EnvData** objektu procesu určeného polem *TMPN* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu v systému IBM i” na stránce 1382 . Obsah těchto dat nemá význam pro správce front.

Délka tohoto pole je dána LNPROE. Počáteční hodnota tohoto pole je 128 prázdných znaků.

TMPN (48-bajtový znakový řetězec)

Název objektu procesu.

Jedná se o název objektu procesu správce front zadaný pro spuštěnou frontu a může být použit aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **ProcessName** fronty identifikované polem *TMQN* ; podrobnosti o tomto atributu viz “Atributy pro fronty” na stránce 1353 .

Názvy, které jsou kratší než definovaná délka pole, jsou vždy doplněny vpravo s mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána LNPRON. Počáteční hodnota tohoto pole je 48 prázdných znaků.

TMQN (48-bajtový znakový řetězec)

Název spuštěné fronty.

Jedná se o název fronty, pro kterou došlo k události spouštěče, a je použita aplikací spuštěnou aplikací pro monitor spouštěčů. Správce front inicializuje toto pole hodnotou atributu **QName** spuštěné fronty; podrobnosti o tomto atributu naleznete v příručce “Atributy pro fronty” na stránce 1353 .

Názvy, které jsou kratší než definovaná délka pole, jsou směrem doprava vyplněny mezerami; nejsou předčasně ukončeny znakem null.

Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

TMSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

TMSIDV
Identifikátor pro strukturu zprávy spouštěče.

Počáteční hodnota tohoto pole je TMSIDV.

TMTD (64bajtový znakový řetězec)

Data spouątěče.

Jedná se o volný formát dat pro použití aplikací monitoru spouštěčů, který přijme zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **TriggerData** fronty identifikované polem *TMQN* ; podrobnosti o tomto atributu viz “Atributy pro fronty” na stránce 1353 . Obsah těchto dat nemá význam pro správce front.

Délka tohoto pole je dána 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 ke spuštění, a používá se aplikací pro monitorování spouštěčů, která přijímá zprávu spouštěče. Správce front inicializuje toto pole s hodnotou atributu **UserData** objektu procesu určeného polem *TMPN* ; podrobnosti o tomto atributu viz “Atributy pro definice procesu v systému IBM i” na stránce 1382 . Obsah těchto dat nemá význam pro správce front.

Délka tohoto pole je dána LNPROU. Počáteční hodnota tohoto pole je 128 prázdných znaků.

TMVER (10číslicové podepsané celé číslo)

Číslo verze struktury.

Hodnota musí být:

TMVER1
Číslo verze pro strukturu zprávy spouštěče.

Následující konstanta uvádí číslo verze aktuální verze:

TMVERC
Aktuální verze struktury zprávy spouštěče.

Počáteční hodnota tohoto pole je TMVER1.

Počáteční hodnoty

Tabulka 738. Počáteční hodnoty polí v MQTM

Název pole	Název konstanty	Hodnota konstanty
TMSID	TMSIDV	'TM¬¬'
TMVER	TMVER1	1
TMQN	Není	Mezery

Tabulka 738. Počáteční hodnoty polí v MQTM (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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 (Spouštěcí zpráva 2-znakový formát) v systému IBM i

Když aplikace pro monitor spouštěčů načte zprávu spouštěče (MQTM) z inicializační fronty, může být nutné, aby monitor spouštěčů předal aplikaci, která je spuštěna monitorem spouštěče, některé nebo všechny informace ve zprávě spouštěče.

Přehled

Účel: Informace, které mohou být potřebné pro spuštěnou aplikaci, zahrnují *TC2QN*, *TC2TD*a *TC2UD*. Aplikace monitoru spouštěčů může přenést strukturu MQTM přímo do spuštěné aplikace, nebo místo toho předat strukturu MQTMC2 , v závislosti na tom, co je povoleno prostředím a vhodné pro spuštěnou aplikaci.

Tato struktura je součástí produktu IBM MQ Trigger Monitor Interface (TMI), který je jedním z rozhraní rámce IBM MQ .

Znaková sada a kódování: Znaková data v souboru MQTMC2 jsou ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front produktu **CodedCharSetId** .

Použití: Struktura MQTMC2 je podobná jako struktura struktury MQTM. Rozdíl spočívá v tom, že neznaková pole v MQTM se změní v MQTMC2 na znaková pole stejné délky a jméno správce front bude přidáno na konec struktury.

- V systému IBM ipředává aplikace monitoru spouštěčů, která byla součástí produktu IBM MQ , strukturu MQTMC2 pro spuštěnou aplikaci.
- “Pole” na stránce 1227
- “Počáteční hodnoty” na stránce 1228
- “Deklarace RPG” na stránce 1228

Pole

Struktura MQTMC2 obsahuje následující pole; tato pole jsou popsána v **abecedním pořadí**:

TC2AI (256bajtový znakový řetězec)

Identifikátor aplikace.

Viz pole *TMAI* ve struktuře MQTM.

TC2AT (čtyřbajtový 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 (48-bajtový znakový řetězec)

Název objektu procesu.

Viz pole *TMPN* ve struktuře MQTM.

TC2QMN (48-bajtový znakový řetězec)

Název správce front.

Jedná se o název správce front, v němž došlo k události spouštěče.

TC2QN (48-bajtový znakový řetězec)

Název spuštěné fronty.

Viz pole *TMQN* ve struktuře MQTM.

TC2SID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

TCSIDV
Identifikátor struktury zprávy spouštěče (znakový formát).

TC2TD (64bajtový znakový řetězec)

Data spouątěče.

Viz pole *TMTD* ve struktuře MQTM.

TC2UD (128bajtový znakový řetězec)

Uživatelská data.

Viz pole *TMUD* ve struktuře MQTM.

TC2VER (čtyřbajtový znakový řetězec)

Číslo verze struktury.

Hodnota musí být:

TCVER2
Struktura zpráv spouštěcího impulsu verze 2 (znaková formát).

Následující konstanta uvádí číslo verze aktuální verze:

TCVERC
Aktuální verze struktury zprávy spouštěče (ve znakovém formátu).

Počáteční hodnoty

Tabulka 739. Počáteční hodnoty polí v 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) v systému IBM i

Struktura MQWIH popisuje informace, které se musí nacházet na začátku zprávy, kterou má zpracovat správce zátěže produktu z/OS .

Přehled

Název formátu: FMWIH.

Znaková sada a kódování: Pole ve struktuře MQWIH jsou ve znakové sadě a kódování dána 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, které jsou platné v názvech front.

Použití: Je-li zpráva zpracovávána správcem zátěže produktu z/OS , musí zpráva začínat strukturou MQWIH.

- "Pole" na stránce 1229
- "Počáteční hodnoty" na stránce 1231
- "Deklarace RPG" na stránce 1231

Pole

Struktura MQWIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

WICSI (10ciferné celé číslo se znaménkem)

Identifikátor znakové sady dat, který následuje za MQWIH.

Určuje identifikátor znakové sady pro data, která následují strukturu MQWIH. Nevztahuje se na znaková data v samotné struktuře MQWIH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům. Je možné použít následující speciální hodnotu:

CSINHT
Zdědit identifikátor znakové sady této struktury.

Znaková data v datech *následující* tato struktura se nachází ve stejné znakové sadě jako tato struktura.

Správce front změní tuto hodnotu ve struktuře odeslané ve zprávě na skutečný identifikátor znakové sady struktury. Není-li zjištěna žádná chyba, hodnota CSINHT se nevrací pomocí volání MQGET.

CSINHT nelze použít, je-li hodnota pole *MDPAT* v MQMD je ATBRKR.

Počáteční hodnota tohoto pole je CSUNDF.

WIENC (10číslicové podepsané celé číslo)

Číselné kódování dat za MQWIH.

Určuje číselné kódování dat, která se řídí strukturou MQWIH. Nevztahuje se na číselná data v samotné struktuře MQWIH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je 0.

WIFLG (Celé číslo s desetimístným znaménkem)

Příznaky

Hodnota musí být:

WINON
Žádné vlajky.

Počáteční hodnota tohoto pole je WINONE.

WIFMT (8bajtový znakový řetězec)

Název formátu dat, který následuje za MQWIH.

Určuje název formátu dat, která následují za strukturou MQWIH.

Na základě 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 LNFMT. Počáteční hodnota tohoto pole je FMNONE.

WILEN (10ciferné celé číslo se znaménkem)

Délka struktury MQWIH.

Hodnota musí být:

WILEN1
Délka struktury záhlaví pracovních informací version-1 .

Následující konstanta uvádí délku aktuální verze:

WILENC
Délka aktuální verze struktury záhlaví pracovních informací.

Počáteční hodnota tohoto pole je WILEN1.

WIRSV (32bajtový znakový řetězec)

Vyhrazeno.

Jedná se o vyhrazené pole; musí být prázdné.

WISID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

WISIDSTAR
Identifikátor pro strukturu 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 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.

V případě volání MQPUT a MQPUT1 je toto pole ignorováno. Délka tohoto pole je dána LNMTOK. Počáteční hodnota tohoto pole je MTKNON.

WIVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

WIVER1
Struktura záhlaví pracovních informací Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

WIVERC
Aktuální verze struktury záhlaví pracovních informací.

Počáteční hodnota tohoto pole je WIVER1.

Počáteční hodnoty

Tabulka 740. Počáteční hodnoty polí v MQWIH

Název pole	Název konstanty	Hodnota konstanty
WISID	WISIDSTAR	'WIH¬'
WIVER	WIVER1	1
WILEN	WILEN1	120
WIENC	Není	0
WICSI	CSUNDF	0
WIFMT	FMNONE	Mezery
WIFLG	WINON	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                                      000000000000000000')
     D* Reserved
     D  WIRSV                  89    120    INZ
```

IBM i MQXQH (záhlaví přenosové fronty) v systému IBM i

Struktura MQXQH popisuje informace, které jsou uvedeny předponou zprávy zpráv aplikace při jejich přenosu do přenosových front.

Přehled

Účel: Přenosová fronta je speciální typ lokální fronty, která dočasně uchovává zprávy určené pro vzdálené fronty (tedy určené pro fronty, které nepatří do lokálního správce 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 poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front daných ENNAT pro programovací jazyk C.

Znaková sada a kódování MQXQH musí být nastaveny na pole *MDCSI* a *MDENC* v:

- Samostatný MQMD (je-li struktura MQXQH na začátku dat zprávy), nebo
- Struktura záhlaví, která předchází struktuře MQXQH (všechny ostatní případy).

Použití: Zpráva, která se nachází v přenosové frontě, má *dva* deskriptory zpráv:

- Jeden deskriptor zprávy je uložen odděleně od dat zprávy; toto se nazývá *samostatný popisovač zprávy*a je generován správcem front, když je zpráva vložena do přenosové fronty. Některá z polí v odděleném deskriptoru zpráv se kopírují z deskriptoru zpráv poskytovaného aplikací v rámci volání MQPUT nebo MQPUT1 .

  Samostatný deskriptor zpráv je ten, který je vrácen aplikaci v parametru **MSGDSC** v rámci volání MQGET, když je zpráva odebrána z přenosové fronty.
- Druhý deskriptor zprávy je uložen ve struktuře MQXQH jako součást dat zprávy; nazývá se *vložený popisovač zpráv*a je kopií deskriptoru zpráv, který byl poskytnut aplikací v rámci volání MQPUT nebo MQPUT1 (s menšími variantami).

  Vložený deskriptor zprávy je vždy version-1 MQMD. Pokud má zpráva uvedená v aplikaci nevýchozí hodnoty pro jedno nebo více polí version-2 v MQMD, struktura MQMDE následuje za MQXQH a je dále následována daty zprávy aplikace (pokud existují). MQMDE je buď:

  – Generováno správcem front (pokud aplikace používá MQMD version-2 k vložení zprávy), nebo
  – Již existuje na začátku dat zprávy aplikace (pokud aplikace používá MQMD version-1 k vložení zprávy).

  Vložený deskriptor zpráv je ten, který je vrácen aplikaci v parametru **MSGDSC** v rámci volání MQGET při odebrání zprávy z fronty konečného cíle.

- “Pole v odděleném deskriptoru zpráv” na stránce 1233
- “Pole v deskriptoru vložených zpráv” na stránce 1234
- “Vložení zpráv do vzdálených front” na stránce 1234
- “Vložení zpráv přímo do přenosových front” na stránce 1234
- “Získávání zpráv z přenosových front” na stránce 1235
- “Pole” na stránce 1235
- “Počáteční hodnoty” na stránce 1236
- “Deklarace RPG” na stránce 1236

Pole v odděleném deskriptoru zpráv

Pole v samostatném deskriptoru zpráv jsou nastavena správcem front, jak je zobrazeno v následujícím seznamu. Pokud správce front nepodporuje MQMD version-2 , použije se MQMD version-1 bez ztráty funkce.

Tabulka 741. Pole v odděleném deskriptoru zpráv a použitých hodnotách

Pole v samostatném deskriptoru MQMD	Použitá hodnota
MDSID	MDSIDROVSKÁ
MDVER	MDVER2
MDREP	Zkopírováno z vloženého deskriptoru zpráv, ale s bity identifikovanými pomocí ROAUXM nastaveným na nulu. (To zabraňuje generování zprávy COA nebo CHSK, když je zpráva vložena nebo odebrána z přenosové fronty.)
MDMT	Zkopírováno z vloženého deskriptoru zpráv.
MDEXP	Zkopírováno z vloženého deskriptoru zpráv.
MDFB	Zkopírováno z vloženého deskriptoru zpráv.
MDENC	ENNAT
MDCSI	Atribut **CodedCharSetId** správce front.
MDFMT	FMXQH
MDPRI	Zkopírováno z vloženého deskriptoru zpráv.
MDPER	Zkopírováno z vloženého deskriptoru zpráv.
MDMID	Nová hodnota je generována správcem front. Tento identifikátor zprávy se liší od identifikátoru *MDMID* , který správce front mohl vygenerovat pro deskriptor vložené zprávy (viz předchozí popis).
MDCID	*MDMID* z deskriptoru vložených zpráv.
MDBOC	0
MDRQ	Zkopírováno z vloženého deskriptoru zpráv.
MDRM	Zkopírováno z vloženého deskriptoru zpráv.
MDUID	Zkopírováno z vloženého deskriptoru zpráv.
MDACC	Zkopírováno z vloženého deskriptoru zpráv.
MDAID	Zkopírováno z vloženého deskriptoru zpráv.
MDPAT	ATQM.
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.
MDAOD	Mezery
MDGID	GINON
MDSEQ	1
MDOFF	0
MDMFL	MFNONE
MDOLN	OLUNDFE.

Pole v deskriptoru vložených zpráv

Pole v deskriptoru vloženého zprávy mají stejné hodnoty jako pole v parametru **MSGDSC** volání MQPUT nebo MQPUT1 , s výjimkou následujících:

- Pole *MDVER* má vždy hodnotu MDVER1.
- Má-li pole *MDPRI* hodnotu PRQDEF, je nahrazena hodnotou atributu **DefPriority** fronty.
- Má-li pole *MDPER* hodnotu PEQDEF, je nahrazena hodnotou atributu **DefPersistence** fronty.
- Pokud má pole *MDMID* hodnotu MINONE, nebo byla zadána volba PMNMID, nebo je zpráva zprávou rozdělovníku, *MDMID* je nahrazen novým identifikátorem zprávy generovaným správcem front.

  Je-li zpráva distribučního seznamu rozdělena do menších zpráv v seznamu přenosových front umístěných v různých přenosových frontách, je pole *MDMID* v každém z nových deskriptorů vložených zpráv stejné jako v původní zprávě distribučního seznamu.
- Pokud byla zadána volba PMNCID, produkt *MDCID* se nahradí novým identifikátorem korelace generovaným správcem front.
- Pole kontextu jsou nastavena tak, jak jsou označena volbami PM* uvedenými v parametru **PMO** ; jsou to pole kontextu:
  - *MDACC*
  - *MDAID*
  - *MDAOD*
  - *MDPAN*
  - *MDPAT*
  - *MDPD*
  - *MDPT*
  - *MDUID*
- Pole version-2 (pokud byla přítomná) budou odebrána z MQMD a přesunuta do struktury MQMDE, pokud jedno nebo více polí version-2 má nevýchozí hodnotu.

Vložení zpráv do vzdálených front

: Když aplikace vloží zprávu do vzdálené fronty (buď uvedením názvu vzdálené fronty přímo, nebo pomocí lokální definice vzdálené fronty), lokálního správce front:

- Vytvoří strukturu MQXQH obsahující deskriptor vnořené zprávy
- Připojí prostředí MQMDE, je-li potřebný, a ještě není přítomen
- Připojí data zprávy aplikace
- Umístí zprávu do příslušné přenosové fronty

Vložení zpráv přímo do přenosových front

Je také možné, aby aplikace vložila zprávu přímo do přenosové fronty. V takovém případě musí aplikace před daty zprávy aplikace připojit strukturu MQXQH a inicializovat pole s příslušnými hodnotami. Kromě toho musí mít pole *MDFMT* v parametru **MSGDSC** volání MQPUT nebo MQPUT1 hodnotu FMXQH.

Znaková data ve struktuře MQXQH vytvořená aplikací musí být ve znakové sadě lokálního správce front (definovaného atributem správce front produktu **CodedCharSetId**) a celočíselné data musí být v kódování nativního počítače. Kromě toho musí být znaková data ve struktuře MQXQH vyplněna mezerami na definovanou délku pole; data nesmí být ukončena předčasně pomocí znaku hex 00, protože správce front nekonvertuje null a následné znaky na mezery ve struktuře MQXQH.

Povšimněte si však, že správce front nekontroluje přítomnost struktury MQXQH nebo že pro tato pole byly zadány platné hodnoty.

Získávání zpráv z přenosových front

Aplikace, které získaní zprávy z přenosové fronty, musí zpracovat informace ve struktuře MQXQH vhodným způsobem. Přítomnost struktury MQXQH na začátku dat zprávy aplikace je označena hodnotou FMXQH, která je vrácena v poli *MDFMT* v parametru **MSGDSC** volání MQGET. Hodnoty vrácené v polích *MDCSI* a *MDENC* v argumentu **MSGDSC** udávají znakovou sadu a kódování znakových a celočíselných dat ve struktuře MQXQH. Znaková sada a kódování dat zprávy aplikace jsou definovány v polích *MDCSI* a *MDENC* v deskriptoru vložených zpráv.

Pole

Struktura MQXQH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

XQMD (MQMD1)

Deskriptor původní zprávy.

Jedná se o vložený deskriptor zpráv a je to kopie deskriptoru MQMD deskriptoru zpráv, která byla zadána jako parametr **MSGDSC** v rámci volání MQPUT nebo MQPUT1 , když byla zpráva původně vložena do vzdálené fronty.

Poznámka: Jedná se o version-1 MQMD.

Počáteční hodnoty polí v této struktuře jsou stejné jako počáteční hodnoty v rámci struktury MQMD.

XQRQ (48-bajtový znakový řetězec)

Název cílové fronty.

Jedná se o název fronty zpráv, která je zdánlivým konečným místem určení zprávy (může se ukázat, že se nejedná o skutečné místo určení, pokud je například tato fronta definována v *XQRQM* jako lokální definice jiné vzdálené fronty).

Pokud se jedná o zprávu distribučního seznamu (to znamená, že pole *MDFMT* v deskriptoru vloženého zprávy je FMDH), *XQRQ* je prázdné.

Délka tohoto pole je dána LNQN. Počáteční hodnota tohoto pole je 48 prázdných znaků.

XQRQM (48-bajtový znakový řetězec)

Název správce cílové fronty.

Jedná se o název správce front nebo skupiny sdílení front, která vlastní frontu, která je zdánlivě konečným cílem zprávy.

Je-li zpráva zprávou rozdělovníku, *XQRQM* je prázdné.

Délka tohoto pole je dána 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 (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

XQVER1
Číslo verze pro strukturu záhlaví přenosové fronty.

Následující konstanta uvádí číslo verze aktuální verze:

XQVERC
Aktuální verze struktury záhlaví přenosové fronty.

Počáteční hodnota tohoto pole je XQVER1.

Počáteční hodnoty

Tabulka 742. Počáteční hodnoty polí 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 710 na stránce 1139	-

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 v systému IBM i

Tyto informace použijte k seznámení se s voláními funkce dostupnými v programování IBM i .

Konvence použité v popisech volání v systému IBM i

U každého volání tato kolekce témat uvádí popis parametrů a použití volání. Následuje typická vyvolání volání a typická deklarace jejích parametrů v programovacím jazyce RPG.

Důležité: Při kódování volání rozhraní API produktu IBM MQ je třeba zajistit, aby byly poskytnuty všechny relevantní parametry (jak je popsáno v následujících sekcích). Pokud tak neučiníte, může dojít k nepředvídatelným výsledkům.

Popis každého volání obsahuje následující sekce:

Název volání
Název volání, za nímž následuje stručný popis účelu volání.

Parametry
Pro každý parametr je za názvem následován jeho datový typ v závorkách (). a jeho směrem; například:

CMPCOD (9-ciferné dekadické celé číslo)-výstup

Další informace o datových typech struktury v produktu “Elementární datové typy” na stránce 988jsou k dispozici.

Směr parametru může být:

Vstup
Tento parametr musíte zadat vy (programátor).

Výstup
Volání vrátí tento parametr.

Vstup a výstup
Tento parametr musíte zadat, ale tento parametr je upraven voláním.

Je zde také stručný popis účelu parametru spolu se seznamem všech hodnot, které může parametr provést.

Poslední dva parametry v každém volání jsou kód dokončení a kód příčiny. Kód dokončení označuje, zda bylo volání dokončeno úspěšně, částečně nebo vůbec. Další informace o částečném úspěchu nebo selhání volání jsou uvedeny v kódu příčiny.

Poznámky k použití
Další informace o volání popisují, jak ji použít a jaká omezení jejího použití používají.

Vyvolání RPG
Typické vyvolání volání a deklarace jeho parametrů v jazyce RPG.

Ostatní notační konvence jsou:

Konstanty
Názvy konstant se zobrazují velkými písmeny, např. OOOUT.

Pole
U některých volání jsou parametry pole znakových řetězců s velikostí, která není pevná. V popisech těchto parametrů představuje malá písmena *n* číselnou konstantu. Když kódíte deklaraci pro tento parametr, nahraďte hodnotu *n* číselnou hodnotou, kterou požadujete.

IBM i MQBACK (změny změn) v systému IBM i

Volání MQBACK označuje správci front, že všechny zprávy typu get a put se vyskytly od posledního synchronizačního bodu, které mají být vráceny. Zprávy, které byly vloženy jako součást pracovní jednotky, se odstraní; zprávy načtené jako součást pracovní jednotky jsou obnoveny ve frontě.

- Toto volání je podporováno v následujících prostředích:
  -  AIX
  -  IBM i
  -  Windows
- “Syntaxe” na stránce 1238
- “Poznámky k použití” na stránce 1238
- “Parametry” na stránce 1239
- “Deklarace RPG” na stránce 1240

Syntaxe

MQBACK *(Hconn, CompCode, Reason)*

Poznámky k použití

Při použití funkce MQBACK zvažte použití těchto poznámek k použití.

1. Toto volání lze použít pouze v případě, že správce front koordinuje jednotku práce. Jedná se o lokální jednotku práce, kde změny ovlivní pouze IBM MQ prostředků.
2. V prostředích, kde správce front nekoordinuje transakci, musí být místo MQBACK použito příslušné zpětné volání. Prostředí může také podporovat implicitní vrácení způsobené aplikací nestandardně ukončeným způsobem.
   - V systému IBM ilze toto volání použít pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** nesmí být vydán pro úlohu.
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce “MQDISC (Odpojení správce front) v systému IBM i” na stránce 1276 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
   - Hodnoty polí *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

   Správce front uchovává *tři* sady informací o skupinách a segmentech, jednu sadu pro každou z následujících možností:

   - Poslední úspěšné volání MQPUT (může být součástí jednotky práce).
   - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí jednotky práce).
   - Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (to nemůže být součástí pracovní jednotky).

   Pokud aplikace vkládá nebo získává zprávy jako součást pracovní jednotky, a pak se aplikace rozhodne zálohovat jednotku práce, informace o skupině a segmentu se obnoví na hodnotu, kterou předtím měla:

   - Informace přidružené k volání MQPUT se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQPUT pro tento popisovač fronty v aktuální transakci.

- Informace přidružené k volání MQGET se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQGET pro daný popisovač fronty v aktuální pracovní jednotce.

Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnovenou skupinovou a segmentovou informaci, pokud je jednotka práce zálohována.

Obnova informace o skupině a segmentu na její předchozí hodnotu, když je zálohována jednotka práce, umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů přes několik jednotek práce a restartovat ve správném bodu ve skupině zpráv nebo v logické zprávě, pokud se jedna z jednotek práce nezdaří. Použití několika jednotek práce může být výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv ve správném okamžiku, pokud dojde k selhání systému. Podrobnosti o restartování ve správném bodu po selhání systému naleznete v části PMLOGO popsané v části “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161a v části GMLOGO popsané v části “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066.

Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce:

1. Jednotka práce má stejný rozsah jako manipulátor připojení. To znamená, že všechna volání IBM MQ , která mají vliv na konkrétní pracovní jednotku, musí být prováděna pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz parametr **HCONN** popsaný v tématu “MQCONN (Připojit správce front) v systému IBM i” na stránce 1263 .
2. Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
3. Dlouhá-spuštěná aplikace, která vydává příkazy MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydá potvrzení nebo odvolání, může způsobit, že fronty budou zaplňovat zprávy, které nejsou k dispozici pro jiné aplikace. Pro ochranu proti této možnosti by měl správce nastavit atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily správně pracovat s očekávanými aplikacemi systému zpráv.

Parametry

Volání MQBACK má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *COMCOD*.

Pokud má parametr *COMCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *COMCOD* 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 je poškozen.

RC2123
(2123, X'84B') Výsledek operace commit nebo back-out je smíšený.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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 (Begin unit of work) na IBM i

Volání MQBEGIN zahajuje transakci, která je koordinována správcem front, a která může zahrnovat externí správce prostředků.

- Toto volání je podporováno v následujících prostředích:
  - AIX AIX
  - IBM i IBM i
  - Windows Windows
- “Syntaxe” na stránce 1241
- “Poznámky k použití” na stránce 1241
- “Parametry” na stránce 1242
- “Deklarace RPG” na stránce 1243

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 jednotek práce:

   Správce front-koordinovaná lokální jednotka práce
   Jedná se o pracovní jednotku, v níž je správce front jediným účastníkem správce prostředků, a správce front tak vystupuje jako koordinátor jednotky práce.

   - Chcete-li spustit tento typ jednotky práce, měla by být volba PMSYP nebo GMSYP zadána na první volání MQPUT, MQPUT1nebo MQGET v pracovní jednotce.

     Není nutné, aby aplikace vydala volání MQBEGIN pro spuštění pracovní jednotky, ale pokud se použije MQBEGIN, volání bude dokončeno s CCWARN a kódem příčiny RC2121.
   - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, musí být použit volání MQCMIT nebo MQBACK.

   Správce front-koordinovaná globální transakce práce
   This is a unit of work in which the queue manager acts as the unit-of-work coordinator, both for IBM MQ resources *a* for resources belonging to other resource managers. Tito správci prostředků spolupracují se správcem front, aby zajistili, že všechny změny prostředků v pracovní jednotce budou potvrzeny nebo vráceny společně.

   - Chcete-li spustit tento typ jednotky práce, musí být použito volání MQBEGIN.
   - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, musí být použita volání MQCMIT a MQBACK.

   Externě koordinovaná globální jednotka práce
   Jedná se o jednotku práce, v níž je správce front účastníkem, ale správce front nepracuje jako koordinátor jednotky práce. Místo toho je zde externí koordinátor jednotek práce, se kterým správce front spolupracuje.

   - Chcete-li spustit tento typ jednotky práce, musí být použit odpovídající volání poskytnutý externím koordinátorem jednotky práce.

     Pokud se volání MQBEGIN použije k pokusu o spuštění pracovní jednotky, volání selže s kódem příčiny RC2012.
   - Chcete-li potvrdit nebo vrátit tento typ pracovní jednotky, je třeba použít potvrzení o provedení a zpětné volání poskytované koordinátorem externí jednotky práce.

     Je-li volání MQCMIT nebo MQBACK použito k pokusu o potvrzení nebo vrácení pracovní jednotky, volání selže s kódem příčiny RC2012.
2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce “MQDISC (Odpojení správce front) v systému IBM i” na stránce 1276 .
3. Aplikace se může účastnit pouze jedné transakce v daném okamžiku. Volání MQBEGIN selže s kódem příčiny RC2128 , pokud již existuje jednotka práce pro aplikaci, bez ohledu na typ jednotky práce, kterou tento objekt má.
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 jednotek práce pro globální jednotky práce, jsou správci prostředků, kteří se mohou podílet na pracovní jednotce, definovány v konfiguračním souboru správce front.
6. V systému IBM ijsou podporovány tyto tři typy pracovní jednotky:

- **Koordinované lokální jednotky práce správce front** lze použít pouze tehdy, když definice vázaného zpracování neexistuje na úrovni úlohy, to znamená, že příkaz STRCMTCTL s argumentem **CMTSCOPE(*JOB)** nesmí být pro danou úlohu vydán.
- **Koordinované globální jednotky práce správce front** nejsou podporovány.
- **Externě koordinované globální pracovní jednotky** lze použít pouze tehdy, když definice vázaného zpracování existuje na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** musí být vydán pro úlohu. Pokud byla tato akce provedena, operace IBM i COMMIT a ROLLBACK se vztahují na prostředky IBM MQ i na prostředky patřící do jiných zúčastněných správců prostředků.

Parametry

Volání MQBEGIN má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

BEGOP (MQBO)-vstupní/výstupní

Volby, které řídí akci MQBEGIN.

Podrobnosti viz “MQBO (Začátek voleb) na IBM i” na stránce 1009.

Nejsou-li vyžadovány žádné volby, programy napsané v assembleru C nebo S/390 mohou uvádět adresu parametru null místo určení adresy struktury MQBO.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2121
(2121, X'849 ') Nejsou registrovány žádné zúčastněné správce prostředků.

RC2122
(2122, X'84A') Zúčastněné správce prostředků není k dispozici.

Je-li *CMPCOD* CCFAIL:

RC2134
(2134, X'856 ') Struktura začátku-volby 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 platné v prostředí.

RC2018
(2018, X'7E2') Popisovač připojení není platný.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2128
(2128, X'850 ') Jednotka práce již byla spuštěna.

Deklarace 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) v systému IBM i

Volání funkce MQBUFMH převede vyrovnávací paměť na popisovač zprávy a je inverzní k volání MQMHBUF.

Toto volání přebírá deskriptor zprávy a vlastnosti MQRFH2 ve vyrovnávací paměti a zpřístupňuje je prostřednictvím popisovače zprávy. Vlastnosti MQRFH2 v datech zprávy jsou volitelně odebrány. Pole *Encoding*, *CodedCharSetId*a *Format* deskriptoru zpráv se aktualizují, je-li to nutné, aby správně popisovaly obsah vyrovnávací paměti po odebrání vlastností.

- “Syntaxe” na stránce 1244
- “Poznámky k použití” na stránce 1244
- “Parametry” na stránce 1244
- “Deklarace RPG” na stránce 1246

Syntaxe

MQBUFMH *(Hconn, Hmsg, BufMsgHOpts, MsgDesc, Buffer, BufferLength, DataLength, CompCode, Reason)*

Poznámky k použití

Volání MQBUFMH nelze zachytit pomocí uživatelských procedur rozhraní API-vyrovnávací paměť je převedena na popisovač zprávy v prostoru aplikace; volání není k dispozici pro správce front.

Parametry

Volání MQBUFMH má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *HCONN* se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu, který převádí vyrovnávací paměť na popisovač zprávy. Není-li ustanoveno platné připojení, volání selže s chybou RC2009.

HMSG (20-digit signed integer)-vstup

Tento úchyt je popisovač zprávy, 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 zpracovávány manipulátory zpráv z vyrovnávacích pamětí.

Podrobnosti viz “MQBMHO (Vyrovnávací paměť pro volby zpracování vyrovnávací paměti) v systému IBM i” na stránce 1008.

MSGDSC (MQMD)-vstupní/výstupní

Struktura *MSGDSC* obsahuje vlastnosti deskriptoru zpráv a popisuje obsah oblasti vyrovnávací paměti.

Ve 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 v kódování aplikace.

BUFLEN (10ciferné číslicové celé číslo)-vstup

BUFLEN je délka oblasti vyrovnávací paměti, v bajtech.

BUFLEN z nulového počtu bajtů je platný a indikuje, že oblast vyrovnávací paměti neobsahuje žádná data.

BUFFER (1-bytový bitový řetězec x BUFLEN)-vstupní/výstupní

BUFFER definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat musíte zarovnat vyrovnávací paměť na 4bajtové hranici.

Pokud *BUFFER* obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MSGDSC** na hodnoty odpovídající datům; to umožní převod dat, je-li to nutné.

Jsou-li vlastnosti nalezeny ve vyrovnávací paměti zpráv, mohou být odebrány později. Později budou k dispozici od obslužné rutiny zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel-to-void, což znamená, že adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BUFLEN** nastaven na nulu, *BUFFER* se na něj neodkazuje. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

DATLEN (10ciferné celé číslo se znaménkem)-výstup

DATLEN je délka vyrovnávací paměti, která může mít odebrané vlastnosti, v bajtech.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *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 domovské ASID se liší.

RC2489
(2489, X'09B9') Struktura obslužného programu vyrovnávací paměti pro zpracování zprávy není platná.

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 bylo zadáno 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 zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

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 nebylo možné analyzovat.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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) v systému IBM i

Volání MQCB znovu zaregistruje zpětné volání pro zadaný popisovač objektu a řídí aktivaci a změny pro zpětné volání.

Zpětné volání je část kódu (zadaná buď jako název funkce, kterou lze dynamicky propojit, nebo jako ukazatel funkce) volanou produktem IBM MQ , když dojde k určitým událostem.

Chcete-li použít MQCB a MQCTL na klientovi V7 , musíte být připojeni k serveru V7 a parametr **SHARECNV** kanálu musí mít nenulová hodnota.

Informace o globálních pracovních jednotkách najdete v tématu Globální jednotky práce.

Typy zpětného volání, které lze definovat, jsou:

Spotřebitel zpráv.
Funkce zpětného volání spotřebitele zpráv se volá tehdy, je-li na manipulátoru objektu dostupná zpráva splňující zadaná kritéria výběru.

Na každém popisovači objektu může být registrována pouze jedna funkce zpětného volání. Má-li být jedna fronta čtena s více kritérii výběru, musí být fronta otevřena vícekrát a musí být registrována funkce spotřebitele na každém popisovači.

obslužná rutina událostí
Obslužná rutina událostí je volána pro podmínky, které ovlivňují celé prostředí zpětného volání.

Funkce je volána, když se vyskytne podmínka události, například správce front nebo zastavení připojení nebo uvedení do klidového stavu.

Funkce není volána pro podmínky, které jsou specifické pro jednotlivého spotřebitele zpráv, například RC2016; , je však volán, pokud funkce zpětného volání neskončí normálně.

- “Syntaxe” na stránce 1247
- “Poznámky k použití pro MQCB” na stránce 1247
- “Parametry pro MQCB” na stránce 1248
- “Deklarace RPG” na stránce 1254

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 použije k získání zprávy.
2. MQCB lze použít k definování rutin zpětného volání před spuštěním spotřeby s rozhraním MQCTL nebo je lze použít v rámci rutiny zpětného volání.
3. Chcete-li použít funkci MQCB mimo rutinu zpětného volání, je třeba nejprve pozastavit spotřebu zpráv pomocí funkce MQCTL a pokračovat ve spotřebě po jejím použití.

Posloupnost zpětného volání odběratele zpráv

V průběhu životního cyklu spotřebitele můžete nakonfigurovat odběratele k vyvolání zpětného volání v klíčových bodech. Příklad:

- když je spotřebitel poprvé registrován,
- při spuštění připojení,
- když je připojení zastaveno a
- je-li odběratel deregistrován, ať už explicitně, nebo implicitně MQCLOSE.

Tabulka 743. Definice příkazu MQCTL

Sloveso	**Význam**
MQCTL (START)	Volání MQCTL s použitím operace CTRLR
MQCTL (ZASTAVIT)	Volání MQCTL pomocí operace CTLSP
MQCTL (ČEKÁNÍ)	Volání MQCTL pomocí operace CTRLW

Umožňuje spotřebiteli udržovat stav přidružený k odběrateli. Je-li aplikace požádána o zpětné volání, jsou pravidla pro vyvolání spotřebitele následující:

Registrovat
Jedná se vždy o první typ vyvolání zpětného volání.

Je vždy volán na stejném podprocesu jako volání MQCB (CBREG).

SPUSTIT
Je vždy volán synchronně s použitím příkazu MQCTL (START).

- Všechna zpětná volání START jsou dokončeny před návratem příkazu MQCTL (START).

Je na stejném vláknu jako doručení zprávy, pokud se požaduje CTLTHR.

Volání se spuštěním není garantováno, pokud například předchozí zpětné volání vyvolá MQCTL (STOP) během MQCTL (START).

ZASTAVIT
Po tomto volání nebudou po tomto volání doručeny žádné další zprávy nebo události, dokud není připojení znovu spuštěno.

Hodnota STOP je garantována, pokud byla aplikace dříve volána pro START, nebo zprávu nebo událost.

ZRUŠIT REGISTRACI
Je vždy posledním typem vyvolání zpětného volání.

Ujistěte se, že aplikace provádí inicializaci a vyčištění na základě podprocesů ve zpětných voláních START a STOP. Inicializaci a vyčištění založené na podprocesu můžete provést pomocí zpětných volání REGISTER a DEREGISTER.

Neuvádějte žádné hypotézy o životnosti a dostupnosti jiného podprocesu než toho, co je uvedeno. Nespoléhejte se například na podproces, který zůstává naživu nad posledním voláním funkce DECREGISTER. Podobně, když jste se rozhodli nepoužívat CTLTHR, nepředpokládejte, že vlákno existuje, kdykoli je připojení spuštěno.

Pokud má vaše aplikace určité požadavky na charakteristiky vlákna, může to vždy vytvořit odpovídajícím způsobem podproces, pak použít MQCTL (WAIT). Tento krok *daruje* podproces na IBM MQ pro asynchronní doručování zpráv.

Použití připojení spotřebitele zpráv

Za normálních okolností, když aplikace vydá jiné volání MQI, zatímco jeden je nevyřízený, volání selže s kódem příčiny RC2219.

Existují však speciální případy, kdy musí aplikace vydat další volání MQI před dokončením předchozího volání. Odběratel může být například vyvolán během volání MQCB s CBRE.

V takovém případě, kdy v důsledku aplikace, která vydala příkaz MQCB nebo MQCTL, je aplikace volána zpět, je aplikace povolena pro další volání MQI. Tato instance znamená, že můžete zadat například volání MQOPEN ve funkci odběratele, když se zavolá s typem CBCCALLT CBCTRC. Je povoleno jakékoli volání MQI s výjimkou MQDISC.

Parametry pro MQCB

Volání MQCB má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Spravovat funkci zpětného volání-parametr HCONN.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OPERATN (10ciferné celé číslo se znaménkem)-vstup

Spravovat funkci zpětného volání-parametr OPERATN.

Operace se zpracovává na zpětné volání definované pro zadaný popisovač objektu. Je třeba určit jednu z následujících voleb. Je-li vyžadována více než jedna volba, lze hodnoty přidat (nepřidávat stejnou konstantu více než jednou) nebo kombinovat s použitím bitové operace OR (pokud programovací jazyk podporuje bitové operace).

Kombinace, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné.

CBREG
Definujte funkci zpětného volání pro zadaný popisovač objektu. Tato operace definuje funkci, která má být volána, a kritéria výběru, která se mají použít.

Je-li již definována funkce zpětného volání pro popisovač objektu, definice je nahrazena. Je-li při nahrazování zpětného volání zjištěna chyba, bude zrušena registrace funkce.

Je-li zpětné volání zaregistrováno v rámci stejné funkce zpětného volání, ve které byla zrušena registrace, je toto volání považováno za operaci nahrazení; počáteční nebo poslední volání se nevyvolá.

CBREG můžete použít s CTLSBU nebo CTLRE.

CBUNKR
Zastavte spotřebovávání zpráv pro popisovač objektu a odeberte popisovač z těch vhodných pro zpětné volání.

Zpětné volání se automaticky zruší, je-li přidružený popisovač uzavřen.

Je-li CBUNR volána ze zákaznického serveru a zpětné volání má definované zastavení volání, je vyvoláno po návratu ze strany spotřebitele.

Je-li tato operace vydána proti *Hobj* bez registrovaného odběratele, volání se vrátí s hodnotou RC2448.

CTLSCITY
Pozastaví příjem zpráv pro popisovač objektu.

Je-li tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí při pozastavení události nepřijímá události a všechny události, které jste minuli v pozastaveném stavu, nejsou při pokračování operace poskytnuty.

Během pozastavení funkce odběratele pokračuje v získávání zpětných volání typu ovládacího prvku.

CTLLO
Obnovte příjem zpráv pro popisovač objektu.

Je-li tato operace použita na obslužnou rutinu událostí, obslužná rutina událostí při pozastavení události nepřijímá události a všechny události, které jste minuli v pozastaveném stavu, nejsou při pokračování 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 naleznete v příručce "MQCBD-Deskriptor zpětného volání" na stránce 287 .

Deskriptor zpětného volání je požadován pouze pro volbu CBREG; pokud není deskriptor povinný, předaný adresa parametru může mít hodnotu null.

HOTBJ (10ciferné celé číslo se znaménkem)-vstup

Správa funkce zpětného volání-parametr HOBJ.

Tento manipulátor představuje přístup, který byl vytvořen objektu, ze kterého má být zpráva spotřebována. Jedná se o popisovač, který byl vrácen z předchozího volání MQOPEN nebo MQSUB (v parametru **HOBJ**).

HOBJ není vyžadována při definování rutiny obslužné rutiny událostí (CBTEH) a musí být zadána 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

Správa funkce zpětného volání-parametr MSGDSC.

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy.

Parametr **MsgDesc** definuje atributy zpráv požadovaných odběratelem a verze MQMD, která má být předána spotřebiteli zpráv.

Parametry *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*a *Offset* v deskriptoru MQMD se používají pro výběr zpráv v závislosti na tom, které volby jsou určeny parametrem **GetMsgOpts** .

Volby *Encoding* a *CodedCharSetId* se používají ke konverzi zpráv, pokud uvedete volbu GMCONV.

Podrobnosti viz MQMD .

MsgDesc se používá pouze pro CBREG a, pokud požadujete hodnoty jiné než výchozí hodnoty pro jakákoli pole. *MsgDesc* se nepoužívá pro obslužnou rutinu událostí.

Pokud deskriptor není požadován, poslaná adresa parametru může mít hodnotu null.

Všimněte si, že pokud je více spotřebitelů registrováno ve stejné frontě s překrývajícími se selektory, zvolený spotřebitel pro každou zprávu není definován.

GMO (MQGMO)-vstup

Spravovat funkci zpětného volání-parametr GMO.

Volby, které řídí, jak bude spotřebitel zpráv přijímat zprávy.

Všechny volby mají význam, jak je popsáno v části "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066, je-li použito na volání MQGET, s výjimkou:

GMSSIG
Tato volba není povolena.

GMBRWF, GMBRWN, GMMBH, GMMBC
Pořadí zpráv doručených uživateli prohlížení je určeno kombinací těchto voleb. Mezi důležité kombinace patří:

GMBRWFCH.
První zpráva ve frontě se doručí opakovaně spotřebiteli. To je užitečné, když spotřebitel destruktivně spotřebovává zprávu ve zpětném volání. Použijte tuto volbu s opatrností.

GMBRWN
Spotřebiteli je dána každá zpráva ve frontě, od aktuální pozice kurzoru, dokud není dosaženo konce fronty.

GMBRWF + GMBRWN
Kurzor se resetuje na začátek fronty. Spotřebitel pak dostane každou zprávu, dokud se kurzor nedostane na konec fronty.

GMBRWF + GMMBH nebo GMMBC
Od začátku fronty je spotřebiteli dána první neoznačená zpráva ve frontě, která je poté označena pro tohoto spotřebitele. Tato kombinace zajistí, aby spotřebitel mohl přijímat nové zprávy za aktuální bod kurzoru za aktuální.

GMBRWN + GMMBH nebo GMMBC
Počínaje pozicí kurzoru je spotřebitel přidělen další neoznačenou zprávu ve frontě, která je poté označena pro tohoto spotřebitele. Tuto kombinaci používejte s pečlivostí, protože zprávy lze přidávat do fronty za aktuální pozicí kurzoru.

GMBRWF + GMBRWN + GMBMBH nebo GMMBC
Tato kombinace není povolena, pokud se používá, volání vrátí hodnotu RC2046.

GMNWT, GMWT a GMWI
Tyto volby řídí způsob vyvolání odběratele.

GMNWT.
Spotřebitel není nikdy volán s RC2033. Spotřebitel je vyvolán pouze pro zprávy a události.

GMWT s nulovou GMWI
Kód RC2033 je předán zákazníkovi pouze tehdy, když nejsou žádné zprávy a

- spotřebitel byl spuštěn
- Spotřebitel byl doručen alespoň jedna zpráva od posledního kódu příčiny zprávy.

Tím zabráníte tomu, aby spotřebitel byl ve smyčce v zaneprázdněném cyklu, je-li zadán nulový interval čekání.

GWT a pozitivní GMWI
Uživatel je vyvolán po uvedeném intervalu čekání s kódem příčiny RC2033. Toto volání se provádí bez ohledu na to, zda byly odběrateli doručovány nějaké zprávy. To umožní uživateli provést zpracování prezenčního signálu nebo zpracování dávkového zpracování.

GMWT a GMWI WIULIM
Tento parametr určuje nekonečné čekání před vrácením hodnoty RC2033. Spotřebitel není nikdy volán s RC2033.

GMO se používá pouze pro CBREG a, pokud požadujete hodnoty jiné než výchozí hodnoty pro jakákoli pole. *GMO* se nepoužívá pro obslužnou rutinu událostí.

Pokud volby nejsou povinné, předaná adresa parametru může mít hodnotu null.

Je-li v rámci struktury MQGMO zadán popisovač vlastností zprávy, je v rámci struktury MQGMO, která je předána do zpětného volání spotřebitele, předána kopie. Při návratu z volání MQCB může aplikace odstranit popisovač vlastností zprávy.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Správa funkce zpětného volání-parametr CMPCOD.

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Spravovat funkci zpětného volání-parametr REASON.

Následující kódy příčiny jsou kódy, které správce front může vrátit pro parametr **REASON** .

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* CCFAIL:

RC2204
(2204, X'89C') Adaptér není k dispozici.

RC2133
(2133, X'855 ') Nelze načíst moduly služeb pro převod dat.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2374
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2157
(2157, X'86D') Primární a domovské ASID se liší.

RC2005
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

RC2219
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2487
(2487, X'9B7') Nesprávné pole typu zpětného volání.

RC2448
(2448, X' 990 ') Nelze zrušit registraci, pozastavit nebo obnovit činnost, protože neexistuje žádné registrované zpětné volání.

RC2486
(2486, X'9B6') Musí být zadán buď *CallbackFunction* , nebo *CallbackName* , ale ne obojí.

RC2483
(2483, X'9B3') Nesprávné pole typu zpětného volání.

RC2484
(2484, X'9B4') Nesprávné pole voleb MQCBD.

RC2140
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2217
(2217, X'8A9') Chybí autorizace pro připojení.

RC2202
(2202, X'89A') Připojení je uváděno do klidového stavu.

RC2203
(2203, X'89B') Spojení se vypíná.

RC2207
(2207, X'89F') Chyba identifikátoru korelace.

RC2010
(2010, X'7DA') Parametr délky dat není platný.

RC2016
(2016, X'7E0') Získá informace o zablokování fronty.

RC2351
(2351, X'92F') Globální jednotky konfliktu práce.

RC2186
(2186, X'88A') Struktura voleb získání zprávy není platná.

RC2353
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

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 místní jednotkou práce.

RC2247
(2247, X'8C7') Volby shody nejsou platné.

RC2485
(2485, X'9B4') Nesprávné pole *MaxMsgLength* .

RC2026
(2026, X'7EA') Deskriptor zprávy není platný.

RC2497
(2497, X'9C1') Uvedený vstupní bod funkce nebyl nalezen v modulu.

RC2496
(2496, X'9C0') Modul byl nalezen, avšak je nesprávného typu; není 32bitový, 64bitový, nebo platnou dynamickou knihovnou odkazů.

RC2495
(2495, X'9BF') Modul nebyl nalezen v cestě pro vyhledávání, nebo neměl oprávnění k načtení.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2331
(2331, X'91B') Použití tokenu zprávy není platné.

RC2033
(2033, X'7F1') Nejsou k dispozici žádné zprávy.

RC2034
(2034, X'7F2') Procházení kurzoru není umístěno na zprávě.

RC2036
(2036, X'7F4') Fronta není otevřená pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2206
(2206, X'89E') Nesprávný kód operace na 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 špatný typ indexu.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2069
(2069, X'815 ') Signál nevyřízený pro tento popisovač.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2109
(2109, X'83D') Volání potlačeno ukončovacím programem.

RC2024
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

RC2072
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2354
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

RC2355
(2355, X' 933 ') Směs volání jednotek práce není podporována.

RC2255
(2255, X'8CF') Unit of work not available for the queue manager to use.

RC2090
(2090, X'82A') Čekací interval v MQGMO není platný.

RC2256
(2256, X'8D0') Chybná verze dodávaného MQGMO.

RC2257
(2257, X'8D1') Chybná verze dodaných MQMD.

RC2298
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

Deklarace 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 (Zavření objektu) na IBM i

Volání MQCLOSE uvolní přístup k objektu a je inverzní k volání MQOPEN.

Syntaxe

MQCLOSE *(HCONN, HOBJ, OPTS, CMPCOD, REASON)*

Poznámky k použití

1. Když aplikace vydá volání MQDISC nebo skončí buď normálně, nebo nestandardně, všechny objekty, které byly otevřeny aplikací a jsou stále otevřené, jsou automaticky uzavřeny s volbou CONONE.
2. Následující body se používají, je-li zavřen objekt *queue*:

- Pokud jsou operace ve frontě prováděny jako součást pracovní jednotky, lze frontu zavřít před nebo po výskytu synchronizačního bodu, aniž by to mělo vliv na výsledek synchronizačního bodu.
- Pokud byla fronta otevřena pomocí volby OOBRW, je kurzor procházení zničen. Je-li fronta později znovu otevřena pomocí volby OOBRW, vytvoří se nový kurzor procházení (viz volba OOBRW popsaná v MQOPEN).
- Pokud je zpráva momentálně uzamčena pro tento popisovač v době volání MQCLOSE, zámek se uvolní (viz volba GMLK popsaná v části "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066).

3. Následující body se použijí, pokud objekt, který se uzavírá, je *dynamická fronta* (buď trvalá, nebo dočasná):
   - U dynamické fronty lze volby CODEL nebo COPURG zadat bez ohledu na volby uvedené v odpovídajícím volání MQOPEN.
   - Když je odstraněna dynamická fronta, všechna volání MQGET s volbou GMWT, která jsou neprovedená proti frontě, jsou zrušena a vrátí se kód příčiny RC2052 . Prohlédněte si volbu GMWT popsanou v "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066.

     Po odstranění dynamické fronty se jakékoli volání (jiné než MQCLOSE), které se pokouší odkazovat na frontu pomocí dříve získaného manipulátoru *HOBJ* , nezdaří s kódem příčiny RC2052.

     Buďte si vědomi toho, že ačkoli k odstraněné frontě není přístup aplikací, fronta se neodebere ze systému a přidružené prostředky se neuvolní, dokud všechny manipulátory, které odkazují na frontu, nebyly zavřeny, a všechny jednotky práce, které ovlivňují frontu, byly buď potvrzeny, nebo vráceny.
   - Je-li odstraněna trvalá dynamická fronta, je-li popisovač *HOBJ* uvedený v příkazu MQCLOSE volaný při volání MQOPEN, které vytvořilo frontu, je provedena kontrola, že identifikátor uživatele, který byl použit k ověření volání MQOPEN, je autorizován k odstranění fronty. Pokud byla v rámci volání MQOPEN zadána volba OOALTU, identifikátor uživatele je zaškrtnut *ODAU*.

     Tato kontrola se neprovede, pokud:
     - Uvedený popisovač je ten, který byl vrácen voláním MQOPEN, který vytvořil frontu.
     - Odstraněná fronta je dočasná dynamická fronta.
   - Je-li ukončena dočasná dynamická fronta, je-li popisovač *HOBJ* uvedený v rámci volání MQCLOSE ten, který byl vrácen voláním MQOPEN, který vytvořil frontu, je tato fronta odstraněna. Tato situace nastane bez ohledu na volby zavření určené v rámci volání MQCLOSE. Pokud ve frontě existují zprávy, jsou zahozeny; nejsou generovány žádné zprávy sestav.

     Pokud existují nepotvrzené jednotky práce, které mají vliv na frontu, fronta a její zprávy jsou stále odstraněny, ale to nezpůsobí selhání jednotek práce. Avšak, jak již bylo popsáno dříve, prostředky přidružené k pracovním jednotkám se neuvolní, dokud nebude každá z jednotek práce potvrzena nebo vrácena.

4. Následující body se použijí, je-li objekt, který se zavírá, *distribuční seznam*:
   - Jediná platná volba zavření pro rozdělovník je CONONE; volání selže s kódem příčiny RC2046 nebo RC2045 , pokud jsou zadány jakékoli jiné volby.
   - Když se zavře distribuční seznam, jednotlivé kódy dokončení a kódy příčiny se nevrátí pro fronty v seznamu-pouze parametry **CMPCOD** a **REASON** volání jsou k dispozici pro diagnostické účely.

     Pokud dojde k selhání při zavírání jedné z front, bude správce front pokračovat ve zpracování a pokusí se zavřít zbývající fronty v seznamu distribucí. Parametry **CMPCOD** a **REASON** volání jsou potom nastaveny na vrácení informací popisujících selhání. Proto je možné, aby byl kód dokončení CCFAIL, i když většina front byla úspěšně uzavřena. Fronta, ve které došlo k chybě, není identifikována.

     Dojde-li k selhání ve více než jedné frontě, není definováno, které selhání se vykazuje v parametrech **CMPCOD** a **REASON** .

Parametry

Volání MQCLOSE má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOTOBJ (10-číslicové celé číslo)-vstupní/výstupní

Popisovač objektu.

Tento manipulátor 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 prostředí. Tato hodnota je:

HOUUH
Nepoužitelná obsluha objektu.

OPTS (10ciferné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQCLOSE.

Parametr **OPTS** řídí, jak je objekt uzavřen. Pouze trvalé dynamické fronty a odběry mohou být zavřeny více než jedním způsobem. 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 1353). Volby zavření jsou shrnuty v tabulce později v tomto tématu.

Trvalé odběry lze buď zachovat, nebo odebrat; tyto odběry jsou vytvářeny pomocí volání MQSUB s volbou SODUR.

Při zavírání popisovače do spravovaného cíle (tj. parametru **Hobj** vráceného při volání MQSUB, který používal volbu SOMAN) správce front při odebrání přidruženého odběru vyčistí veškeré nenačtené publikace. To se provádí pomocí volby CORMSB u parametru **Hsub** vráceného na volání MQSUB. Všimněte si, že CORMSB je výchozí chování MQCLOSE pro netrvalý odběr.

Při zavírání popisovače do nespravovaného místa určení jste zodpovědni za vyčištění fronty, kde jsou publikování odesílána. Doporučuje se uzavřít odběr pomocí souboru CORMSB nejprve a poté zprávy zpracovat mimo frontu, dokud nezbývají žádné zbývající zprávy.

Musí být zadán jeden (a pouze jeden) z následujících:

Volby uzavření dynamické fronty

Tyto volby řídí, jak jsou zavírány trvalé dynamické fronty:

KODELA
Odstraňte frontu.

Fronta je odstraněna, pokud platí některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a neexistují žádné zprávy ve frontě a neexistují žádné nepotvrzené příkazy pro získání nebo vložení nevyřízených požadavků do fronty (buď pro aktuální úlohu, nebo pro libovolnou jinou úlohu).
- Jedná se o dočasnou dynamickou frontu, která byla vytvořena voláním MQOPEN, které vrátilo hodnotu *HOBJ*. V tomto případě budou vymazány všechny zprávy ve frontě.

Ve všech ostatních případech, včetně případu, kdy byl příkaz *Hobj* vrácen při volání MQSUB, volání selže s kódem příčiny RC2045a objekt nebude odstraněn.

KOPURGU
Odstraňte frontu a odstraňte na ní zprávy.

Fronta je odstraněna, pokud platí některá z následujících podmínek:

- Jedná se o trvalou dynamickou frontu vytvořenou předchozím voláním MQOPEN a neexistují žádné nepotvrzené příkazy get nebo put pro danou frontu (buď pro aktuální úlohu, nebo pro kteroukoli jinou úlohu).
- Jedná se o dočasnou dynamickou frontu, která byla vytvořena 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 RC2045a objekt nebude odstraněn.

Další tabulka uvádí, které volby zavření jsou platné, a zda je objekt zachován nebo odstraněn.

Tabulka 744. Platné volby zavření pro použití s zadržené nebo odstraněné objekty

Typ objektu nebo fronty	**CONONE**	**KODELA**	**KOPURGU**
Objekt jiný než fronta	Zachováno	Neplatný	Neplatný
Předdefinovaná fronta	Zachováno	Neplatný	Neplatný
permanentní dynamická fronta	Zachováno	Odstraněno, pokud jsou prázdné a žádné nevyřízené aktualizace	Odstraněné zprávy; fronta odstraněna, pokud nejsou žádné nevyřízené aktualizace
Dočasná dynamická fronta (volání vydané tvůrcem fronty)	Odstraněno	Odstraněno	Odstraněno
Dočasná dynamická fronta (volání není vydáno tvůrcem fronty)	Zachováno	Neplatný	Neplatný
Distribuční seznam	Zachováno	Neplatný	Neplatný
Místo určení spravovaného odběru	Zachováno	Neplatný	Neplatný
Distribuční seznam (odběr byl odebrán)	Zprávy odstraněny; fronta odstraněna	Neplatný	Neplatný

Volby uzavření odběru

Tyto volby řídí, zda jsou trvalé odběry odebrány při zavření popisovače a zda jsou znovu vyčištěny publikace, které stále čekají na čtení aplikací. Tyto volby jsou platné pouze pro použití s manipulátorem na objekt vrácený v parametru **HSUB** volání MQSUB.

COKPSSCOMMENT
Manipulátor s odběrem je uzavřen, ale odběr je zachován. Publikace budou nadále odesílána do místa určení uvedeného v odběru. Tato volba je platná pouze v případě, že byl odběr proveden s volbou SODUR. COKPSB je výchozí nastavení, je-li odběr trvalý

CORMSB
Odběr je odebrán a popisovač pro odběr je uzavřen.

Parametr **Hobj** volání MQSUB není zneplatněn uzavřením parametru **Hsub** a může být i nadále používán pro příkazy MQGET nebo MQCB k přijetí zbývajících publikování. Je-li také uzavřen parametr **Hobj** volání MQSUB, pokud se jednalo o spravované místo určení, budou odebrány všechny nenačtené publikace.

CORMSB je výchozí hodnota, pokud je odběr netrvalý.

Tyto volby uzavření odběru jsou shrnuty v následujících tabulkách:

Chcete-li zavřít trvalý popisovač odběru, ale ponechat jej v odběru, použijte následující volby uzavření odběru:

Tabulka 745. Volby úlohy pro zavření manipulátoru trvalého odběru a opuštění odběru

Úloha	**Volba uzavření odběru**
Ponechat publikace na obslužné rutiny MQOPENed	COKPSSCOMMENT
Odebrat publikace na obslužné rutiny MQOPENed	Akce není povolena
Ponechat publikace na popisovači se společností SOMAN	COKPSSCOMMENT
Odebrat publikování na popisovači se společností SOMAN	Akce není povolena

Chcete-li zrušit odběr, buď uzavřením manipulátoru trvalého odběru a zrušením jeho odběru nebo uzavřením popisovače netrvalého odběru, použijte následující volby uzavření odběru:

Tabulka 746. Volby úloh pro zrušení odběru

Úloha	**Volba uzavření odběru**
Ponechat publikace na obslužné rutiny MQOPENed	CORMSB
Odebrat publikace na obslužné rutiny MQOPENed	Akce není povolena
Ponechat publikace na popisovači se společností SOMAN	CORMSB
Odebrat publikování na popisovači se společností SOMAN	KOPGSB

Volby dopředného čtení

Následující volby řídí, co se stane s netrvalými zprávami, které byly odeslány klientovi dříve, než je aplikace požadovala a ještě nebyla aplikací spotřebována. Tyto zprávy jsou uloženy ve vyrovnávací paměti pro čtení napřed klienta čekající na žádost aplikací a mohou být zahozeny nebo spotřebovávány z fronty před dokončením operace MQCLOSE.

COIMM
Objekt se zavře okamžitě a všechny zprávy, které byly odeslány na klienta před tím, než je aplikace požadovala, jsou vyřazeny a nejsou k dispozici pro použití žádnou aplikací. Toto je výchozí hodnota.

KOQSC
Je vytvořen požadavek na uzavření objektu, ale pokud se všechny zprávy, které byly odeslány klientovi před požadovanou aplikací, stále nacházejí v vyrovnávací paměti čtení napřed klienta, volání MQCLOSE se vrátí s kódem varování RC2458a popisovač objektu zůstane platný.

Aplikace pak může pokračovat v používání ovladače objektu k načítání zpráv, dokud není k dispozici více informací, a poté objekt zavřít znovu. Žádné další zprávy nebudou odeslány klientovi před aplikací, která požaduje aplikaci, je nyní vypnuto čtení napřed.

Aplikace se doporučuje používat COQSC raději než se pokoušet o dosažení bodu, kdy ve vyrovnávací paměti pro čtení klienta již nejsou žádné další zprávy, protože zpráva může přijít mezi posledním voláním MQGET a následujícím příkazem MQCLOSE, které by bylo vyřazeno, pokud byl použit modul COIMM.

Je-li příkaz MQCLOSE s rozhraním COQSC zadán v rámci asynchronní funkce zpětného volání, platí stejné chování při čtení zpráv s dopředným čtením. Je-li vrácen kód varování RC2458 , bude funkce zpětného volání zavolána alespoň jedna. Pokud byla do funkce zpětného volání předána poslední zbývající zpráva, která byla dopředným čtením, pole CBCFLG je nastaveno na CBCFBE.

Výchozí volba

Pokud již nepotřebujete žádné z výše popsaných voleb, můžete použít následující volbu:

CONONE
Není vyžadováno žádné volitelné ukončení zpracování.

Musí být zadán pro:

- Objekty jiné než fronty
- Předdefinované fronty
- Dočasné dynamické fronty (ale pouze v těch případech, kdy *HOBJ* není popisovač vrácený voláním MQOPEN, který vytvořil frontu).
- Distribuční seznamy

Ve všech předchozích případech je objekt zachován a není odstraněn.

Je-li tato volba zadána pro dočasnou dynamickou frontu:

- Fronta se odstraní, pokud byla vytvořena voláním MQOPEN, které vrátilo *HOBJ* ; všechny zprávy, které jsou ve frontě, jsou vyprázdněny.
- Ve všech ostatních případech jsou fronta (a všechny její zprávy v něm) uchována.

Je-li tato volba zadána pro trvalou dynamickou frontu, je fronta zachována a není odstraněna.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2241
(2241, X'8C1') Skupina zpráv není úplná.

RC2242
(2242, X'8C2') Logická zpráva není úplná.

Je-li *CMPCOD* 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') Chybí autorizace pro přístup.

RC2101
(2101, X'835 ') Objekt je 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 není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2055
(2055, X'807 ') Fronta obsahuje jednu nebo více zpráv nebo nepotvrzené vložení nebo získání požadavků.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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) v systému IBM i

Volání MQCMIT signalizuje správci front, že aplikace dosáhla synchronizačního bodu, a že všechny operace get a put, které se vyskytly od posledního synchronizačního bodu, jsou trvalé. Zprávy, které jsou vloženy jako součást pracovní jednotky, jsou zpřístupněny ostatním aplikacím; zprávy načtené jako součást pracovní jednotky jsou odstraněny.

- "Syntaxe" na stránce 1261
- "Poznámky k použití" na stránce 1261
- "Parametry" na stránce 1262

- “Deklarace RPG” na stránce 1263

Syntaxe

MQCMIT *(HCONN, COMCOD, REASON)*

Poznámky k použití

Při použití MQCMIT zvažte použití těchto poznámek k použití.

1. Toto volání lze použít pouze v případě, že správce front koordinuje jednotku práce. Jedná se o lokální jednotku práce, kde změny ovlivní pouze IBM MQ prostředků.
2. V prostředích, ve kterých správce front nekoordinuje pracovní jednotku, je třeba namísto funkce MQCMIT použít příslušné volání potvrzení. Prostředí může také podporovat implicitní potvrzení způsobené normálně ukončeným aplikačním programem.
   - V systému IBM ilze toto volání použít pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **CMTSCOPE(*JOB)** nesmí být vydán pro úlohu.
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce “MQDISC (Odpojení správce front) v systému IBM i” na stránce 1276 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
   - Hodnoty polí *MDGID*, *MDSEQ*, *MDOFF*a *MDMFL* v MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

   Když je jednotka práce potvrzena, správce front zachová informace o skupině a segmentu a aplikace může pokračovat ve vkládání nebo získávání zpráv do aktuální skupiny zpráv nebo logické zprávy.

   Zachování informací o skupině a segmentech při potvrzení transakce umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů v rámci několika pracovních jednotek. Použití několika jednotek práce může být výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv ve správném okamžiku, pokud dojde k selhání systému. Podrobnosti o restartování ve správném bodu po selhání systému naleznete v části PMLOGO popsané v části “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161a v části GMLOGO popsané v části “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066.

Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce:

1. Jednotka práce má stejný rozsah jako manipulátor připojení. To znamená, že všechna volání IBM MQ , která mají vliv na konkrétní pracovní jednotku, musí být prováděna pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu popisovačů připojení naleznete v popisu parametru **HCONN** popsaného v MQCONN.
2. Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
3. Dlouhá-spuštěná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydá potvrzení nebo zpětné volání, může způsobit, že fronty budou zaplňovat zprávy, které nejsou k dispozici pro jiné aplikace. Pro ochranu proti této možnosti by měl správce nastavit atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily správně pracovat s očekávanými aplikacemi systému zpráv.

Parametry

Volání MQCMIT má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

COMCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *COMCOD*.

Pokud má parametr *COMCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *COMCOD* CCWARN:

RC2003
(2003, X'7D3') Unit of work backed out.

RC2124
(2124, X'84C') Výsledek operace vázaného zpracování je nevyřízený.

Je-li *COMCOD* 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 je poškozen.

RC2123
(2123, X'84B') Výsledek operace commit nebo back-out je smíšený.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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) v systému IBM i

Volání MQCONN připojí aplikační program ke správci front. Poskytuje manipulátor připojení ke správci front, který je používán aplikací při následných voláních front zpráv.

- Aplikace musí používat volání MQCONN nebo MQCONNX pro připojení ke správci front a volání MQDISC k odpojení od správce front.

V systému IBM MQ for Multiplatformsse každý podproces v aplikaci může připojit k různým správcům front. V jiných systémech se musí všechna souběžná připojení v rámci procesu nacházet ve stejném správci front.

- "Syntaxe" na stránce 1263
- "Poznámky k použití" na stránce 1263
- "Parametry" na stránce 1264
- "Deklarace RPG" na stránce 1266

Syntaxe

MQCONN *(QMNAME, HCONN, CMPCOD, REASON)*

Poznámky k použití

1. Správce front, k němuž je vytvořeno připojení pomocí volání MQCONN, se nazývá *lokální správce front*.
2. Fronty vlastněné lokálním správcem front se v aplikaci zobrazují jako lokální fronty. Je možné vkládat zprávy do těchto front a získávat zprávy z těchto front.

   Sdílené fronty, které vlastní skupina sdílení front, do níž patří lokální správce front, se do aplikace zobrazují jako lokální fronty. Je možné vkládat zprávy do těchto front a získávat zprávy z těchto front.

   Fronty vlastněné vzdálenými správci front se zobrazují jako vzdálené fronty. Je možné vkládat zprávy do těchto front, ale z těchto front není možné zprávy načíst.
3. Pokud správce front selže při spuštění aplikace, musí aplikace znovu vydat volání MQCONN za účelem získání nového manipulátoru připojení, který má být použit při následných voláních produktu IBM MQ . Aplikace může volání MQCONN periodicky volat, dokud nebude volání úspěšné.

   Pokud aplikace není jisté, zda je připojena ke správci front, může aplikace bezpečně vydat volání MQCONN za účelem získání manipulátoru připojení. Je-li aplikace již připojena, vrácený popisovač bude shodný s tím, který vrátil předchozí volání MQCONN, ale s kódem dokončení CCWARN a kódem příčiny RC2002.
4. Po dokončení použití volání produktu IBM MQ by aplikace měla použít volání MQDISC k odpojení od správce front.

5. V systému IBM inejsou programy, které končí abnormálně, automaticky odpojeny od správce front. Aplikace by proto měly být napsány tak, aby umožňovaly možnost volání MQCONN nebo volání MQCONNX při vracení kódu dokončení CCWARN a kódu příčiny RC2002. Manipulátor připojení vrácený v této situaci může být použit jako normální.

Parametry

Volání MQCONN má následující parametry:

QMNAME (48-bajtový znakový řetězec)-vstup

Název správce front.

Jedná se o název správce front, k němuž se aplikace chce připojit. Název může obsahovat následující znaky:

- Velká abecední znaky (A až Z)
- Malá abecední znaky (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 významných dat v názvu; hodnoty null a libovolné znaky následující za ním jsou považovány za prázdné znaky. V označeném prostředí platí následující omezení:

- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procento uzavřeny do uvozovek, jsou-li zadány v příkazech. Tyto uvozovky nesmí být zadané v argumentu **QMNAME** .

Je-li název tvořen zcela mezerami, použije se název *výchozího* správce front.

Název zadaný pro *QMNAME* musí být název správce front *connectable* .

Skupiny sdílení front:V systémech, ve kterých existuje několik správců front a kteří jsou konfigurováni pro vytvoření skupiny sdílení front, lze název skupiny sdílení front zadat pro produkt *QMNAME* místo názvu správce front. To umožňuje aplikaci připojit se k *libovolnému* správci front, který je k dispozici ve skupině sdílení front. Systém může být také nakonfigurován tak, že mezera *QMNAME* způsobí připojení ke skupině sdílení front, nikoli k výchozímu správci front.

Pokud parametr *QMNAME* uvádí název skupiny sdílení front, ale v systému je také správce front s tímto názvem, bude připojení k prvnímu správci front připojované k dřívější verzi. Pouze v případě, že připojení selže, je pokus o připojení k jednomu ze správců front v dané skupině sdílení front.

Je-li připojení úspěšné, může být manipulátor vrácený voláním MQCONN nebo MQCONNX použit pro přístup ke *všem* prostředkům (sdíleným i nesdíleným), které patří ke konkrétnímu správci front, k němuž došlo k připojení. Přístup k těmto prostředkům je předmětem typického řízení autorizace.

Pokud aplikace vydá dvě volání MQCONN nebo MQCONNX za účelem navázání souběžných připojení a jedno nebo obě volání určuje název skupiny sdílení front, může druhé volání vrátit kód dokončení CCWARN a kód příčiny RC2002. To nastane, 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 rámci dávky, dávky RRS a prostředí TSO.

Aplikace klienta IBM MQ: Pro aplikace produktu IBM MQ MQI client je pokus o připojení pro každou definici kanálu připojení klienta s určeným názvem správce front až do úspěšného provedení připojení. Správce front však musí mít stejný název jako určený název. Je-li zadán název all-blank, je každý kanál připojení klienta se všemi prázdnými názvy správce front úspěšný, dokud nebude jeden úspěšný; v tomto případě není žádná kontrola proti skutečnému názvu správce front.

Skupiny správců front klienta IBM MQ: Pokud zadaný název začíná hvězdičkou (*), může skutečný správce front, k němuž je připojení vytvořeno, mít jiný název než ten, který je určen aplikací. Určený název (bez hvězdičky) definuje *skupinu* správců front, kteří jsou způsobilí pro připojení. Implementace vybere jednu ze skupin tím, že se každé z nich pokusí postupně, v abecedním pořadí, dokud

nebude nalezeno připojení, na které může být vytvořeno připojení. Není-li pro připojení k dispozici žádný správce front ve skupině, volání se nezdaří. Každý správce front je zkoušeni pouze jednou. Pokud je pro název zadána pouze hvězdička, bude použita výchozí skupina správce front definovaná implementací.

Skupiny správců front jsou podporovány pouze pro aplikace spuštěné v prostředí klienta MQ; volání se nezdaří, pokud aplikace, která není typu klient, určuje název správce front začínající hvězdičkou. Skupina je definována poskytnutím několika definic kanálů připojení klienta se stejným názvem správce front (zadané jméno bez hvězdičky) ke komunikaci s každým z správců front ve skupině. Výchozí skupina je definována poskytnutím jedné nebo více definic kanálů připojení klienta, každý s prázdným názvem správce front (zadání celého prázdného názvu má proto stejný účinek jako uvedení jedné hvězdičky pro název aplikace klienta).

Po připojení k jednomu správci front skupiny může aplikace v polích názvu správce front v deskriptorech zpráv a v deskriptorech objektů určovat mezery tak, aby určoval název správce front, ke kterému byla aplikace skutečně připojena (*lokální správce front*). Pokud aplikace potřebuje znát tento název, lze pomocí volání MQINQ zadat dotaz na atribut správce front produktu **QMgrName** .

Při určení předpony názvu připojení je nutné, aby aplikace nebyla závislá na připojení ke konkrétnímu správci front ve skupině. Vhodná aplikace by byla:

- Aplikace, které vložila zprávy, ale nedostali zprávy.
- Aplikace, které vložila zprávy požadavků a poté získaly zprávy odpovědí z *dočasné dynamické* fronty.

Nevhodné aplikace by byly ty, které potřebují získat zprávy z určité fronty v konkrétním správci front; takové aplikace by neměly mít předponu názvu s hvězdičkou.

Všimněte si, že je-li uvedena hvězdička, maximální délka zbytku názvu je 47 znaků.

Délka tohoto parametru je dána LNQMN.

HCONN (10ciferné celé číslo se znaménkem)-výstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Musí být zadán ve všech následných voláních do front zpráv, které byly vydány aplikací. Po zadání volání MQDISC přestane být platná, nebo když se ukončí jednotka zpracování, která definuje rozsah manipulátorů.

Rozsah manipulátoru je omezen na nejmenší jednotku. paralelní zpracování podporované platformou, na které je aplikace spuštěna; popisovač není platný mimo jednotku paralelního zpracování, ze které bylo vydáno volání MQCONN.

- V systému IBM ije rozsah manipulátoru úloha, která volá volání.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2002
(2002, X'7D2') Aplikace je již připojena.

Je-li *CMPCOD* 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') Chybí autorizace pro přístup.

RC2137
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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 (Připojit správce front (rozšířený)) v systému IBM i

Volání MQCONNX připojuje aplikační program ke správci front. Poskytuje manipulátor připojení ke správci front, který je používán aplikací při následných voláních IBM MQ .

Volání MQCONNX se podobá volání MQCONN, až na to, že MQCONNX umožňuje určit 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 se musí všechna souběžná připojení v rámci procesu nacházet ve stejném správci front.

Syntaxe

MQCONNX *(QMNAME, CNOPT, HCONN, CMPCOD, REASON)*

Parametry

Volání MQCONNX má následující parametry:

QMNAME (48-bajtový znakový řetězec)-vstup

Název správce front.

Podrobné informace naleznete v popisu parametru **QMNAME** popsaného v příručce "MQCONN (Připojit správce front) v systému IBM i" na stránce 1263 .

CNOPT (MQCNO)-vstupní/výstupní

Volby, které řídí akci MQCONNX.

Podrobnosti viz "MQCNO (Volby připojení) na systému IBM i" na stránce 1038.

HCONN (10ciferné celé číslo se znaménkem)-výstup

Manipulátor připojení.

Podrobné informace naleznete v popisu parametru **HCONN** popsaného v příručce "MQCONN (Připojit správce front) v systému IBM i" na stránce 1263 .

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Podrobné informace naleznete v popisu parametru **CMPCOD** popsaného v příručce "MQCONN (Připojit správce front) v systému IBM i" na stránce 1263 .

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Podrobnosti o možných kódech příčiny naleznete v popisu parametru **REASON** popsaného v příručce "MQCONN (Připojit správce front) v systému IBM i" na stránce 1263 .

Volání MQCONNX může vrátit následující další kódy příčiny:

Je-li *CMPCOD* CCFAIL:

RC2278
(2278, X'8E6') Pole připojení klienta nejsou platná.

RC2139
(2139, X'85B') Struktura volby Connect-options není platná.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

Deklarace 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 manipulátoru zprávy) v systému IBM i

Volání MQCRTMH vrací popisovač zprávy.

Aplikace ji může použít při následných voláních front zpráv:

- Pomocí volání MQSETMP můžete nastavit vlastnost pro popisovač zprávy.
- Pomocí volání MQINQMP můžete zjišťovat hodnotu vlastnosti obslužné rutiny zprávy.
- Pomocí volání MQDLTMP můžete odstranit vlastnost popisovače zprávy.

Manipulátor zpráv lze použít v rámci volání MQPUT a MQPUT1 k přidružení vlastností popisovače zpráv k vlastnostem vkládané zprávy. Podobně zadáním manipulátoru zprávy na volání MQGET lze k vlastnostem načítané zprávy přistupovat prostřednictvím obsluhy zprávy po dokončení volání MQGET.

K odstranění manipulátoru zprávy použijte příkaz MQDLTMH .

- “Syntaxe” na stránce 1268
- “Parametry” na stránce 1268
- “Deklarace RPG” na stránce 1270

Syntaxe

MQCRTMH *(Hconn, CrtMsgHOpts, Hmsg, CompCode, Reason)*

Parametry

Volání MQCRTMH má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX. Pokud připojení ke správci front přestane být platné a žádné volání příkazu IBM MQ na obslužné rutiny zprávy není k dispozici, MQDLTMH je implicitně voláno pro odstranění zprávy.

Případně můžete zadat následující hodnotu:

HCUNAS
Manipulátor připojení nepředstavuje připojení k žádnému konkrétnímu správci front.

Je-li použita tato hodnota, musí být popisovač zprávy odstraněn s explicitním voláním funkce MQDLTMH , aby bylo možné uvolnit úložiště, které mu bylo přiděleno; IBM MQ nikdy implicitně odstraní popisovač zprávy.

Musí existovat alespoň jedno platné připojení ke správci front zavedenému na podprocesu, který vytváří obslužnou rutinu zpráv, jinak se volání nezdaří s chybou RC2018.

CRTOPT (MQCMHO)-vstup

Volby, které řídí akci MQCRTMH. Podrobnosti viz MQCMHO .

HMSG (20místný podepsaný integer)-výstup

Na výstupu je vrácen popisovač zprávy, který lze použít k nastavení, zjišťování a odstranění vlastností popisovače zpráv. Na počátku popisovač zprávy neobsahuje žádné vlastnosti.

Popisovač zprávy má také přidružený deskriptor zprávy. Na počátku tento deskriptor zprávy obsahuje výchozí hodnoty. Hodnoty asociovaných polí deskriptoru zpráv lze nastavit a provádět dotazy pomocí volání MQSETMP a MQINQMP. Volání MQDLTMP resetuje pole deskriptoru zprávy zpět na výchozí hodnotu.

Je-li parametr *HCONN* zadán jako hodnota HCUNAS, vrácený manipulátor zprávy lze použít pro volání MQGET, MQPUT nebo MQPUT1 k jakémukoli připojení v rámci jednotky zpracování, ale může být v daném okamžiku používáno pouze jedním voláním IBM MQ . Pokud je manipulátor používán, když se druhý volání IBM MQ pokusí použít stejný popisovač zprávy, dojde k selhání druhého volání IBM MQ s kódem příčiny RC2499.

Pokud parametr *HCONN* nemá hodnotu HCUNAS, lze vrácený popisovač zprávy použít pouze pro zadané připojení.

Stejná hodnota parametru *HCONN* musí být použita při následných voláních MQI, kde je použit tento manipulátor zprávy:

- MQDLTMH
- MQSETMP
- MQINQMP
- MQDLTMP
- MQMBUF
- MQBUFMH5

Vrácený popisovač zprávy přestane být platný, když je pro popisovač zprávy vydán volání MQDLTMH, nebo když je ukončena jednotka zpracování, která definuje rozsah manipulátoru. Příkaz MQDLTMH je volán implicitně, pokud je při vytvoření popisovače zprávy zadáno specifické připojení a připojení ke správci front již není platné, například pokud je volána funkce MQDBC.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *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 domovské ASID se liší.

RC2219
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2461
(2461, X'099D') Není platná struktura voleb popisovače zprávy vytvoření zprávy.

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 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část “Návratové kódy pro IBM i (ILE RPG)” na stránce 1411.

Deklarace 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 (Řízení zpětného volání) v systému IBM i

Volání MQCTL provádí řízení akcí na manipulátorech objektů otevřených pro připojení.

- “Syntaxe” na stránce 1271
- “Poznámky k použití” na stránce 1271
- “Parametry” na stránce 1271
- “Deklarace RPG” na stránce 1275

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 vydat příkaz MQCB (CBREG), aby zabránil opakovanému volání rutiny zpětného volání.

Parametry

Volání MQCTL má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OPERATN (10ciferné celé číslo se znaménkem)-vstup

Operace se zpracovává na zpětné volání definované pro zadaný popisovač objektu. Musíte uvést jednu a jednu jedinou z následujících možností:

CLSR
Spustit přijímání zpráv pro všechny definované funkce odběratele zpráv pro uvedený popisovač připojení.

Zpětná volání se spouští na podprocesu spuštěnému systémem, který se liší od všech podprocesů aplikace.

Tato operace poskytuje řízení poskytovaného manipulátoru připojení k systému. Jediné volání MQI, které může být vydáno jiným vláknem, než je odběratelský podproces, je:

- MQCTL s operací CTLSP
- MQCTL s operací CTRLU
- MQDISC-Provádí operaci MQCTL s operací CTLSP před odpojením připojení HConn.

Hodnota RC2500 je vrácena v případě, že je při spuštění manipulátoru připojení zadáno volání rozhraní API IBM MQ a volání nepochází z funkce spotřebitele zpráv.

Dojde-li k selhání připojení, bude konverzace ukončena co nejdříve. Je tedy možné, aby bylo při volání rozhraní API produktu IBM MQ na hlavním podprocesu za chvíli obdržen návratový kód RC2500 , za nímž následuje návratový kód RC2009 , když se připojení vrátí do zastaveného stavu.

To může být vydáno ve funkci odběratele. Pro stejné připojení jako rutina zpětného volání je jeho jediným účelem zrušení dříve vydané operace CTLSP.

Tato volba není podporována, je-li aplikace svázána s knihovnou IBM MQ bez podprocesů.

CTLSW
Spustit přijímání zpráv pro všechny definované funkce odběratele zpráv pro uvedený popisovač připojení.

Spotřebitelé zpráv se spouštějí na stejném podprocesu a řízení se nevrací volajícímu objektu MQCTL, dokud:

- Uvolněno v použití operací MQCTL CTLSP nebo CTRLU, nebo
- Všechny rutiny odběratele byly deregistrovány nebo pozastaveny.

Jsou-li všichni spotřebitelé odregistrováni nebo pozastaveni, je vydána implicitní operace CTLSP.

Tuto volbu nelze použít v rámci rutiny zpětného volání, a to ani pro aktuální popisovač připojení, ani pro žádný jiný manipulátor připojení. Pokud je volání vyzkoušeno, vrátí se s hodnotou RC2012.

Pokud v průběhu operace CTRLW neexistují žádné registrované, nepozastavené spotřebitele, volání selže s kódem příčiny RC2446.

Je-li během operace CTRLW připojení pozastaveno, volání MQCTL vrátí kód příčiny varování RC2521; , připojení zůstane 'spuštěno'.

Aplikace se může rozhodnout pro vydání CTLSP nebo CTLRE. V této instanci jsou bloky operací CTLRE.

Tato volba není podporována v jednom vláknovém klientovi.

CLSP
Zastavte příjem zpráv a počkejte, až všichni spotřebitelé dokončí své operace před dokončením této volby. Tato operace uvolní manipulátor připojení.

Je-li tato volba vydána v rámci rutiny zpětného volání, nebude tato volba účinná, dokud rutina nebude ukončena. Žádné další rutiny pro spotřebitele zpráv se nezavolají po dokončení zpracování rutin pro zprávy, které již byly přečteny, a po zastavení volání (je-li požadována) pro rutiny zpětného volání.

Je-li vydáno mimo rutinu zpětného volání, řízení se nevrátí k volajícímu, dokud nebudou dokončeny rutiny odběratele pro zprávy, které již byly načteny, a po ukončení volání (je-li požadována) na zpětné volání. Samotné zpětné volání však zůstává registrováno.

Tato funkce nemá žádný vliv na zprávy dopředného čtení. Musíte zajistit, aby spotřebitelé spouštěli MQCLOSE (COQSC), ze své funkce zpětného volání, abyste určili, zda jsou k dispozici nějaké další zprávy, které je možné doručit.

CTLSCITY
Pozastavit příjem zpráv. Tato operace uvolní manipulátor připojení.

To nemá vliv na dopředné čtení zpráv pro aplikaci. Hodláte-li přestat spotřebovávat zprávy po dlouhou dobu, zvažte uzavření fronty a opětovné otevření, až bude spotřeba pokračovat.

Je-li vydáno v rámci rutiny zpětného volání, neprojeví se, dokud rutina nebude ukončena. Po ukončení aktuální rutiny nebudou volány žádné další rutiny pro spotřebitele zpráv.

Je-li vydáno mimo zpětné volání, řízení se nevrátí k volajícímu, dokud nebude dokončena aktuální zákaznický rutina a nebudou zavolány žádné další.

CTLLO
Pokračujte ve spotřebování zpráv.

Tato volba je obvykle vydána z hlavního podprocesu aplikace, ale lze ji také použít v rámci rutiny zpětného volání ke zrušení dřívější žádosti o pozastavení vydané ve stejné rutině.

Pokud se CTLRE používá k obnovení CTLSW, pak bloky operace.

PCTLOP (MQCTLO)-vstup

Volby, které řídí akci MQCTL

Podrobnosti o struktuře viz MQCTLO .

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Následující kódy příčiny jsou ty, které může správce front vrátit pro parametr **Reason** .

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCFAIL:

RC2133
(2133, X'855 ') Nelze načíst moduly služeb pro převod dat.

RC2204
(2204, X'89C') Adaptér není k dispozici.

RC2130
(2130, X'852 ') Nelze načíst modul služby adaptéru.

RC2374
(2374, X' 946 ') API uživatelské procedury se nezdařilo.

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2157
(2157, X'86D') Primární a domovské 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, pozastavení nebo obnovení, protože neexistuje žádné registrované zpětné volání

RC2486
(2486, X'9B6') Buď byla zadána hodnota CallbackFunction i CallbackName v rámci volání CBREG, nebo byla zadána jedna z voleb CallbackFunction nebo CallbackName , ale neodpovídá aktuálně registrované funkci zpětného volání.

RC2483
(2483, X'9B3') Nesprávné pole typu CallBackType.

RC2219
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2444
(2444, X'98C') Blok volby je chybný.

RC2484
(2484, X'9B4') Nesprávné pole voleb MQCBD.

RC2140
(2140, X'85C') Požadavek na čekání byl odmítnut CICS.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2217
(2217, X'8A9') Chybí autorizace pro připojení.

RC2202
(2202, X'89A') Připojení je uváděno do klidového stavu.

RC2203
(2203, X'89B') Spojení se vypíná.

RC2207
(2207, X'89F') Chyba identifikátoru korelace.

RC2016
(2016, X'7E0') Získá informace o zablokování fronty.

RC2351
(2351, X'92F') Globální jednotky konfliktu práce.

RC2186
(2186, X'88A') Struktura voleb získání zprávy není platná.

RC2353
(2353, X' 931 ') Manipulátor v použití pro globální pracovní jednotku.

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 místní jednotkou práce.

RC2247
(2247, X'8C7') Volby shody nejsou platné.

RC2485
(2485, X'9B5') Nesprávná hodnota pole MaxMsgLength

RC2026
(2026, X'7EA') Deskriptor zprávy není platný.

RC2497
(2497, X'9C1') Uvedený vstupní bod funkce nebyl nalezen v modulu.

RC2496
(2496, X'9C0') Modul je nalezen, ale je nesprávného typu (32 bitů nebo 64 bitů) nebo není platnou knihovnou DLL.

RC2495
(2495, X'9BF') Modul nebyl nalezen v cestě pro vyhledávání, nebo neměl oprávnění 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 otevřená pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2488
(2488, X'9B8') Nesprávný kód operace na 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 špatný typ indexu.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2069
(2069, X'815 ') Signál nevyřízený pro tento popisovač.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2109
(2109, X'83D') Volání potlačeno ukončovacím programem.

RC2072
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2354
(2354, X' 932 ') Zařazení do globální jednotky práce se nezdařilo.

RC2355
(2355, X' 933 ') Směs volání jednotek práce není podporována.

RC2255
(2255, X'8CF') Unit of work not available for the queue manager to use.

RC2090
(2090, X'82A') Čekací interval v MQGMO není platný.

RC2256
(2256, X'8D0') Chybná verze dodávaného MQGMO.

RC2257
(2257, X'8D1') Chybná verze dodaných MQMD.

RC2298
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

Deklarace 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 (Odpojení správce front) v systému IBM i

Volání MQDISC přeruší spojení mezi správcem front a aplikačním programem a je inverzní k volání MQCONN nebo MQCONNX.

Syntaxe

MQDISC *(HCONN, CMPCOD, REASON)*

Poznámky k použití

1. Je-li volání MQDISC vydáno, je-li aplikace stále má otevřené objekty, jsou tyto objekty zavřeny správcem front a volby zavření nastaveny na CONONE.
2. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, jak aplikace končí:
   a. Pokud aplikace vydá volání MQDISC před ukončením:
      - Pro koordinovanou pracovní jednotku správce front vydá správce front volání MQCMIT pro danou aplikaci. Jednotka práce je potvrzena, pokud je to možné, a vrácena, pokud ne.
      - Pro externě koordinovanou jednotku práce není žádná změna stavu pracovní jednotky; správce front však bude informovat o tom, že pracovní jednotka by měla být potvrzena koordinátorem jednotky práce, který má být potvrzený.
   b. Pokud aplikace skončí normálně, ale bez zadání volání MQDISC, je jednotka práce vrácena zpět.
   c. Pokud aplikace skončí *nestandardně* bez volání volání MQDISC, bude jednotka práce vrácena zpět.

Parametry

Volání MQDISC má následující parametry:

HCONN (10ciferné celé číslo se znaménkem)-vstupní/výstupní

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

Při úspěšném dokončení volání nastaví správce front *HCONN* na hodnotu, která není platným popisovačem pro dané prostředí. Tato hodnota je:

HCUNUH
Nepoužitelná obsluha připojení.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* 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 není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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 (Výmaz manipulátoru zprávy) v systému IBM i

Volání MQDLTMH odstraní popisovač zprávy a je inverzní k volání MQCRTMH.

Syntaxe

MQDLTMH *((Hconn, Hmsg, DltMsgHOpts, CompCode, Reason)*

Poznámky k použití

1. Toto volání můžete použít pouze v případě, že správce front sám koordinuje pracovní jednotku. To může být:
   - Lokální jednotka práce, kde se změny týkají pouze IBM MQ prostředků.
   - Globální jednotka práce, kde mohou změny ovlivnit prostředky patřící jiným správcům prostředků a které ovlivňují prostředky produktu IBM MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz “MQBEGIN (Begin unit of work) na IBM i” na stránce 1240.
2. V prostředích, kde správce front nekoordinuje jednotku práce, použijte místo MQBACK odpovídající zpětné volání. Prostředí může také podporovat implicitní vrácení zpět v důsledku abnormálního ukončení aplikace.
   - V systému z/OSpoužijte následující volání:
     - Dávkové programy (včetně dávkových DL/I programů produktu IMS) mohou použít volání MQBACK, pokud má jednotka práce vliv pouze na prostředky produktu IBM MQ . However, if the unit of work affects both IBM MQ resources and resources belonging to other resource managers (for example, Db2), use the SRRBACK call provided by the z/OS Recoverable Resource Service (RRS). Volání SRRBACK vrací změny prostředků náležejících ke správcům prostředků, kteří byli povoleni pro koordinaci RRS.
     - Aplikace produktu CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k zálohování jednotky práce. Nepoužívejte volání MQBACK pro aplikace produktu CICS .
     - Aplikace produktu IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. produkt `ROLB` , aby odvrátila jednotku práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové DL/I programy).
   - V systému IBM ipoužijte toto volání pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz `STRCMTCTL` s parametrem **`CMTSCOPE(*JOB)`** nesmí být vydán pro úlohu.
3. Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce “MQDISC (Odpojení správce front) v systému IBM i” na stránce 1276 .
4. Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
   - Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v MQMD.
   - Zda je zpráva součástí jednotky práce.
   - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

   Správce front uchovává tři skupiny informací o skupinách a segmentech, jednu sadu pro každou z následujících možností:
   - Poslední úspěšné volání MQPUT (může být součástí jednotky práce).
   - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí jednotky práce).

- Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (to nemůže být součástí pracovní jednotky).

Pokud aplikace vkládá nebo získává zprávy jako součást pracovní jednotky a aplikace pak zálohuje pracovní jednotku, informace o skupině a segmentu se obnoví na hodnotu, kterou předtím měla:

- Informace přidružené k volání MQPUT se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQPUT pro tento popisovač fronty v aktuální transakci.
- Informace přidružené k volání MQGET se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQGET pro daný popisovač fronty v aktuální pracovní jednotce.

Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnovenou skupinovou a segmentovou informaci, pokud je jednotka práce zálohována.

Obnova informace o skupině a segmentu na její předchozí hodnotu, když je zálohována jednotka práce, umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů přes několik jednotek práce a restartovat ve správném bodu ve skupině zpráv nebo v logické zprávě, pokud se jedna z jednotek práce nezdaří. Použití několika jednotek práce může být výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv na správném místě, pokud dojde k selhání systému.

Podrobnosti o restartování ve správném bodu po selhání systému najdete v tématu PMLOGO popsané v části PMOPT (10 číslic se znaménkem celého čísla)a s volbou GMLOGO popsané v souboru GMOPT (celé číslo se znaménkem 10).

Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce:

5. Jednotka práce má stejný rozsah jako manipulátor připojení. Všechna volání IBM MQ , která ovlivňují konkrétní jednotku práce, musí být prováděna pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz HCONN (10 číslic signed integer)-výstup pro informace o rozsahu manipulátorů připojení.
6. Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
7. Dlouhá-spuštěná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydává výzvu k potvrzení nebo vrácení, může vyplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Pro ochranu proti této možnosti musí administrátor nastavit atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily správné fungování očekávaných aplikací systému zpráv.

Parametry

Volání MQDLTMH má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu, který odstraňuje popisovač zprávy, jinak se volání nezdaří s RC2009 .

HMSG (20ciferné celé číslo se znaménkem)-vstupní/výstupní

Jedná se o popisovač zprávy, který má být odstraněn. Hodnota byla vrácena předchozím voláním MQCRTMH.

Při úspěšném dokončení volání je manipulátor nastaven na neplatnou hodnotu pro dané prostředí. Tato hodnota je:

HMUNHAF
Nepoužitelná obsluha zprávy.

Popisovač zprávy nelze odstranit, pokud probíhá jiný volání IBM MQ , kterému byl předán stejný popisovač zprávy.

DLTOPT (MQDMHO)-vstup

Podrobnosti viz MQDMHO .

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

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 domovské ASID se liší.

RC2219
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2462
(2462, X'099E') Struktura obslužného programu odstranění zprávy není platná.

RC2460
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část “Návratové kódy pro IBM i (ILE RPG)” na stránce 1411.

Deklarace 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-Odstranění vlastnosti zprávy

Volání MQDLTMP odstraní vlastnost z manipulátoru zprávy a je inverzní k volání MQSETMP.

- “Syntaxe” na stránce 1281
- “Parametry” na stránce 1281
- “Deklarace RPG” na stránce 1282

Syntaxe

MQDLTMP *(Hconn, Hmsg, DltPropOpts, Name, CompCode, Reason)*

Parametry

Volání MQDLTMP má následující parametry:

HCONN (10ciferné celé číslo se znaménkem)-Vstup

Tento manipulátor představuje připojení ke správci front. Hodnota se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu, který odstraňuje manipulační prostředek zprávy, jinak se volání nezdaří s chybou RC2009.

HMSG (20-digit signed integer)-vstup

Jedná se o popisovač zprávy obsahující vlastnost, která má být odstraněna. Hodnota byla vrácena předchozím voláním MQCRTMH.

DLTOPT (MQDMPO)-Vstup

Podrobnosti naleznete v datovém typu MQDMPO .

PRNAME (MQCHARV)-vstup

Název vlastnosti, která má být odstraněna. Viz Názvy vlastností , kde jsou další informace o názvech vlastností.

Zástupné znaky nejsou v názvu vlastnosti povoleny.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2471
(2471, X'09A7') Vlastnost není k dispozici.

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nebylo možné analyzovat.

Je-li *CMPCOD* CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'086D') Primární a domovské ASID se liší.

RC2219
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2009
(2009, X'07D9') Připojení ke správci front bylo ztraceno.

RC2481
(2481, X'09B1') Odstranění struktury voleb vlastnosti zprávy není platné.

RC2460
(2460, X'099C') Popisovač zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

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 ') Vyskytla se neočekávaná chyba.

Další informace o těchto kódech najdete v tématu Kódy dokončení a příčin rozhraní API.

Deklarace 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 (Načtení zprávy) v systému 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 vymazána z fronty. Toto odstranění se může vyskytnout jako součást samotného volání MQGET, nebo jako součást synchronizačního bodu. Odstranění zprávy se neprovede, pokud je v parametru **GMO** uvedena volba GMBRWF nebo GMBRWN (viz pole *GMOPT* popsané v "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066).
2. Je-li uvedena volba GMLK s jednou z voleb procházení, je procházená zpráva uzamčena tak, aby byla viditelná pouze pro tento popisovač.

   Je-li zadána volba GMUNLK, je odemknuta dříve zamčená zpráva. V tomto případě není načtena žádná zpráva a parametry **MSGDSC**, **BUFLEN**, **BUFFER** a **DATLEN** se nekontrolují ani nemění.
3. Je-li aplikace, která vydala volání MQGET, spuštěna jako IBM MQ MQI client, je možné, aby zpráva byla ztracena při zpracování volání MQGET při nestandardním ukončení IBM MQ MQI client nebo je přerušeno připojení klienta. K tomu dochází proto, že náhradní osoba, která je spuštěna na platformě správce front a která vydává volání MQGET v zastoupení klienta, nemůže zjistit ztrátu klienta, dokud náhradní identifikátor nevrátí zprávu klientovi; to je poté, co byla zpráva odebrána z fronty. K tomu může dojít u trvalých zpráv i přechodných zpráv.

   Riziko ztráty zpráv tímto způsobem lze eliminovat vždy načtením zpráv v rámci pracovních jednotek (tj. určením volby GMSYP na volání MQGET a použitím volání MQCMIT nebo MQBACK k potvrzení nebo vrácení jednotky práce při zpracování zprávy). Je-li uveden GMSYP a klient se ukončí abnormálně nebo je spojení přerušeno, náhradní jednotka provede odvolání transakce na správci front a zpráva je ve frontě znovu zavedena.

   V zásadě může stejná situace nastat s aplikacemi, které jsou spuštěny na platformě správce front, ale v tomto případě je okno, během kterého může být zpráva ztracená, malá. Avšak stejně jako u IBM MQ MQI clients lze riziko eliminovat načtením zprávy v rámci pracovní jednotky.
4. Pokud aplikace vkládá posloupnost zpráv do konkrétníhofronta v rámci jedné jednotky práce a poté potvrdí, že jednotka práce byla úspěšně dokončena, zprávy jsou k dispozici pro načtení následujícím způsobem:
   - Je-li fronta *nesdílená fronta* (tedy lokální fronta), budou všechny zprávy v rámci jednotky práce k dispozici ve stejnou dobu.
   - Je-li fronta *sdílená fronta*, budou zprávy v rámci jednotky práce dostupné v pořadí, ve kterém byly vloženy, ale ne všechny najednou. Je-li systém silně zatížen, je možné, aby první zpráva v jednotce práce byla úspěšně načtena, ale pro volání MQGET pro druhou nebo následující zprávu v jednotce

práce selžou s chybou RC2033. Pokud k tomu dojde, aplikace musí čekat krátkou dobu a poté se pokusit o provedení operace znovu.

5. Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny určité podmínky. Podrobnosti naleznete v poznámkách k použití v popisu volání MQPUT. Jsou-li podmínky splněny, jsou zprávy předloženy přijímající aplikaci v pořadí, v jakém byly odeslány, pokud:

   - Z fronty získává zprávy pouze jeden příjemce.

     Pokud existují dvě nebo více aplikací, které dostávají zprávy z fronty, musí souhlasit s odesilatelem mechanizmu, který má být použit k identifikaci zpráv, které patří do posloupnosti. Odesilatel může například nastavit všechna pole MDCID ve zprávách v posloupnosti na hodnotu, která byla jedinečná pro danou posloupnost zpráv.
   - Příjemce neprovede záměrné změny pořadí načítání, například zadáním konkrétního MDMID nebo MDCID.

   Pokud odesílající aplikace zadala zprávy jako skupinu zpráv, jsou zprávy předkládány přijímající aplikaci ve správném pořadí, pokud přijímající aplikace uvádí volbu GMLOGO na volání MQGET. Další informace o skupinách zpráv viz:

   - Pole MDMFL v deskriptoru MQMD
   - Volba PMLOGO v MQPMO
   - Volba GMLOGO v produktu MQGMO
6. Test aplikací pro kód zpětné vazby FBQUIT v poli MDFB parametru **MSGDSC** . Je-li tato hodnota nalezena, aplikace se ukončí. Další informace naleznete v poli MDFB uvedeném v příručce “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099 .
7. Pokud byla fronta označená HOBJ otevřena pomocí volby OOSAVA, a kód dokončení z volání MQGET je CCOK nebo CCWARN, kontext přidružený k manipulátoru fronty HOBJ je 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 na následné volání MQPUT nebo MQPUT1 uvedením voleb PMPASI nebo PMPASA. To umožňuje přenést kontext přijaté zprávy jako celek nebo jeho část do jiné zprávy (například, když je zpráva předána do jiné fronty). Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.
8. Je-li parametr GMCONV zahrnut do parametru **GMO** , data zprávy aplikace jsou převedena na reprezentaci požadovanou přijímající aplikací před tím, než jsou data vložena do parametru **BUFFER** :

   - Pole MDFMT v informacích o ovládacím prvku ve zprávě identifikuje strukturu dat aplikace a pole MDCSI a MDENC v řídicích informacích ve zprávě uvádí identifikátor a kódování znakové sady.
   - Aplikace, která volá volání MQGET, uvádí v polích MDCSI a MDENC v parametru **MSGDSC** identifikátor a kódování znakové sady, do kterého musí být data zprávy aplikace převedena.

   Je-li konverze dat zprávy nezbytná, provede převod buď samotným správcem front, nebo uživatelem zapsaným výstupem, v závislosti na hodnotě pole MDFMT v informacích o ovládacím prvku ve zprávě:

   - Následující formáty jsou automaticky převedeny správcem front; tyto formáty se nazývají "vestavěné" formáty:

FMADMN	FMMUDE
FMCICS	FMPCF
FMCMD1	FMRMHCACH
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 při načítání zprávy z fronty nepokusí o převod.

  Poznámka: Je-li parametr GMCONV zadán v rámci volání MQGET pro zprávu s názvem formátu FMNONE a znaková sada nebo kódování zprávy se liší od hodnoty zadané argumentem **MSGDSC** , zpráva je stále vrácena v parametru **BUFFER** (nepředpokládá se žádné další chyby), ale volání je dokončeno s kódem dokončení CCWARN a kódem příčiny RC2110.

  FMNONE lze použít buď tehdy, když povaha dat zprávy znamená, že nevyžaduje převod, nebo když se odesílající a přijímající aplikace dohodly mezi sebou formulářem, ve kterém by měla být data zprávy odeslána.
- Všechny ostatní názvy formátu způsobí, že zpráva bude předána uživatelské proceduře pro převod. Ukončení má stejný název jako formát, kromě dodatků specifických pro prostředí. Názvy formátů zadaných uživatelem nesmí začínat písmeny "MQ", protože tyto názvy mohou být v konfliktu s názvy formátů, které jsou podporovány v budoucnosti.

Uživatelská data ve zprávě lze převést mezi libovolnými podporovanými znakovými sadami a kódováními. Uvědomte si však, že pokud zpráva obsahuje jednu nebo více struktur záhlaví IBM MQ , zprávu nelze převést ze znakové sady, která má znaky s dvoubajtovou nebo vícebajtovou nebo vícebajtovou znakovou sadou pro některý ze znaků platných v názvech front. Kód příčiny RC2111 nebo RC2115 má za následek pokus o provedení tohoto pokusu a zpráva je vrácena nekonverzovanou. Příkladem takové znakové sady je znaková sada Unicode UTF-16 .

Při návratu z MQGET označuje následující kód příčiny, že zpráva byla úspěšně převedena:

- RCNONE

Následující kód příčiny informuje o tom, že zpráva mohla být úspěšně převedena; aplikace musí zkontrolovat pole MDCSI a MDENC v parametru **MSGDSC** , aby zjistila:

- RC2079

Všechny ostatní kódy příčiny indikují, že zpráva nebyla převedena.

Poznámka: Interpretace kódu příčiny popsaného v tomto příkladě platí pro převody prováděné uživatelem napsanými výstupy pouze v případě, že uživatelská procedura odpovídá pokynům pro zpracování.

9. U vestavěných formátů uvedených výše může správce front provést výchozí převod znakových řetězců ve zprávě, je-li zadána volba GMCONV. Výchozí převod umožňuje správci front použít výchozí znakovou sadu určenou pro instalaci, která se blíží ke skutečné znakové sadě při převodu řetězcových dat. Výsledkem je, že volání MQGET může být úspěšné s kódem dokončení CCOK místo dokončení s CCWARN a kódem příčiny RC2111 nebo RC2115.

   Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězcových dat je to, že některé znaky mohou být nesprávně převedeny. Tomu lze zabránit tak, že použijete v řetězci pouze znaky, které jsou společné jak pro skutečnou znakovou sadu, tak pro výchozí znakovou sadu.

   Výchozí převod platí jak pro data zprávy aplikace, tak pro znaková pole v strukturách MQMD a MQMDE:

   - Výchozí konverze dat zprávy aplikace se vyskytne pouze tehdy, když jsou všechny následující příkazy pravdivé:
     - Aplikace specifikuje GMCONV.
     - Zpráva obsahuje data, která musí být převedena buď ze znakové sady nebo do znakové sady, která není podporována.
     - Výchozí převod byl povolen, když byl správce front nainstalován nebo restartován.
   - 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 volba GMCONV není uvedena aplikací na volání MQGET.

10. Parametr **BUFFER** uvedený v příkladu programování v RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li požadována větší vyrovnávací paměť, musí být deklarovaný parametr deklarovaný jako struktura nebo jako pole ve fyzickém souboru.

    Deklarování parametru jako struktury zvyšuje maximální povolenou délku na 9999 bajtů při deklarování parametru jako pole ve fyzickém souboru zvyšuje maximální možnou délku na přibližně 32 KB.

Parametry

Volání MQGET má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota HCONN byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOTBJ (10ciferné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento popisovač představuje frontu, ze které se má načíst zpráva. Hodnota HOBJ byla vrácena předchozím voláním MQOPEN. Fronta musí být otevřena s jednou nebo více z následujících voleb (podrobnosti viz “MQOPEN (Otevřít objekt) v systému IBM i” na stránce 1307):

- OOINPS
- OOINPX
- OOINPQ
- OOBRW

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zpráv.

Tato struktura popisuje atributy požadované zprávy a atributy načtené zprávy. Podrobnosti viz “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099.

Je-li BUFLEN menší než délka zprávy, MSGDSC je stále zadán správcem front, zda je GMATM zadán v parametru **GMO** (viz pole GMOPT popsané v “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066).

Pokud aplikace poskytuje version-1 MQMD, vrácená zpráva má před daty zprávy aplikace předponu MQMDE, ale pouze v případě, že jedno nebo více polí v prostředí MQMDE má nevýchozí hodnotu. Pokud mají všechna pole v MQMDE výchozí hodnoty, je MQMDE vynechán. Název formátu FMMDE v poli MDFMT v MQMD označuje, že je přítomen MQMDE.

GMO (MQGMO)-vstup/výstup

Volby, které řídí akci MQGET.

Podrobnosti viz “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066.

BUFLEN (10ciferné číslicové celé číslo)-vstup

Délka (v bajtech) oblasti BUFFER .

Nula lze zadat pro zprávy, které nemají žádná data, nebo pokud má být zpráva odebrána z fronty a vyřazena data (GMATM musí být v tomto případě uvedeno).

Poznámka: Délka nejdelší zprávy, kterou je možné číst z fronty, je dána atributem fronty **MaxMsgLength** ; viz “Atributy pro fronty” na stránce 1353.

BUFFER (1-bytový 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 záhlaví záhlaví IBM MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Je-li BUFLEN menší než délka zprávy, přesune se do BUFFER co nejvíce zpráv; k tomu dochází, je-li v parametru **GMO** zadán GMATM (více informací viz pole GMOPT popsané v "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066).

Znaková sada a kódování dat v **BUFFER** jsou dána poli MDCSI a MDENC vrácenými v argumentu **MSGDSC** . Jsou-li tyto hodnoty odlišné od hodnot požadovaných příjemcem, příjemce musí data zprávy aplikace převést na znakovou sadu a požadované kódování. Volbu GMCONV lze použít s uživatelem napsaným výstupem pro provedení převodu dat zprávy (podrobnosti o této volbě naleznete v části "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066).

Poznámka: Všechny ostatní parametry volání MQGET se nacházejí ve znakové sadě a kódování lokálního správce front (přidělený atributem správce front **CodedCharSetId** a ENNAT).

Pokud se volání nezdaří, mohl by se obsah vyrovnávací paměti stále měnit.

DATLEN (10ciferné celé číslo se znaménkem)-výstup

Délka zprávy.

Jedná se o délku dat aplikace ve zprávě v bajtech. Je-li tato délka zprávy větší než BUFLEN, vrátí se v parametru **BUFFER** pouze BUFLEN bytů (to znamená, že zpráva je zkrácena). Je-li hodnota nula, znamená to, že zpráva neobsahuje žádná data aplikace.

Je-li BUFLEN menší než délka zprávy, DATLEN je stále zadán správcem front, zda je GMATM zadán v parametru **GMO** (viz pole GMOPT popsané v "MQGMO (volby získání zpráv) v systému IBM i" na stránce 1066). To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k umístění dat zprávy a pak znovu vydat volání s vyrovnávací pamětí odpovídající velikosti.

Je-li však uvedena volba GMCONV a převedená data zprávy jsou příliš dlouhá na to, aby se vešly do BUFFER, hodnota vrácená pro DATLEN je:

- Délka nepřevedených dat, pro formáty definované správcem front.

  V tomto případě, pokud by charakter dat způsobil rozšíření během konverze, musí aplikace alokovat vyrovnávací paměť větší než hodnotu vrácenou správcem front pro DATLEN.
- Hodnota vrácená uživatelskou procedurou pro převod dat pro formáty definované aplikací.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující CMPCOD.

Následující kódy příčiny jsou ty, které může správce front vrátit pro parametr **REASON** . Pokud aplikace uvádí volbu GMCONV a uživatelská procedura je vyvolána pro převod některých nebo všech dat zprávy, je to uživatelské procedury, která rozhodne, jaká hodnota je vrácena pro parametr **REASON** . V důsledku toho jsou možné hodnoty jiné než hodnoty zdokumentované později v této sekci.

Pokud má parametr CMPCOD hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li CMPCOD CCWARN:

RC2120
(2120, X'848 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

RC2190
(2190, X'88E') Konvertovaný řetězec je příliš velký pro pole.

RC2150
(2150, X'866 ') DBCS řetězec není platný.

RC2110
(2110, X'83E') Formát zprávy není platný.

RC2243
(2243, X'8C3') Segmenty zprávy mají odlišné CCSID.

RC2244
(2244, X'8C4') Segmenty zprávy mají odlišné kódování.

RC2209
(2209, X'8A1') Žádná zpráva nebyla zamknuta.

RC2119
(2119, X'847 ') Data zprávy nejsou převedena.

RC2272
(2272, X'8E0') Data zprávy jsou 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 čísel s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

RC2112
(2112, X'840 ') Kódování celého čísla zdroje nebylo rozpoznáno.

RC2143
(2143, X'85F') Parametr délky zdroje není platný.

RC2146
(2146, X'862 ') Cílový parametr vyrovnávací paměti není platný.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2117
(2117, X'845 ') Packed-decimal encoding specified by receiver not recognized.

RC2118
(2118, X'846 ') Kódování čísel s pohyblivou řádovou čárkou určené příjemcem není rozpoznáno.

RC2116
(2116, X'844 ') Cílové celé číslo kódování nebylo rozpoznáno.

RC2079
(2079, X'81F') Byla vrácena oříznutá zpráva (zpracování dokončeno).

RC2080
(2080, X'820 ') Byla vrácena zkrácená zpráva (zpracování není dokončeno).

Je-li CMPCOD 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 informace o zablokování fronty.

RC2186
(2186, X'88A') Struktura voleb získání zprávy 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 úplná.

RC2242
(2242, X'8C2') Logická zpráva není úplná.

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 zprávy není platný.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2033
(2033, X'7F1') Nejsou k dispozici žádné zprávy.

RC2034
(2034, X'7F2') Procházení kurzoru není umístěno na zprávě.

RC2036
(2036, X'7F4') Fronta není otevřená pro procházení.

RC2037
(2037, X'7F5') Fronta není otevřena pro vstup.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

RC2101
(2101, X'835 ') Objekt je 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 není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2024
(2024, X'7E8') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

RC2072
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2255
(2255, X'8CF') Unit of work not available for the queue manager to use.

RC2090
(2090, X'82A') Čekací interval v MQGMO není platný.

RC2256
(2256, X'8D0') Chybná verze dodávaného MQGMO.

RC2257
(2257, X'8D1') Chybná verze dodaných MQMD.

Deklarace 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 (Dotaz na atributy objektů) v systému IBM i

Volání MQINQ vrací pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

Platné jsou tyto typy objektů:

- Fronta
- Seznam názvů
- Definice procesu
- Správce front

Syntaxe

MQINQ *(HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON)*

Poznámky k použití

1. Vrácené hodnoty jsou snímky vybraných atributů. Neexistuje žádná záruka, že atributy nebudou změněny dříve, než bude aplikace moci reagovat na vrácené hodnoty.
2. Otevřete-li modelovou frontu, vytvoří se dynamická lokální fronta. To platí i v případě, že otevřete modelovou frontu s dotazem na její atributy.

   Atributy dynamické fronty (s určitými výjimkami) jsou stejné jako atributy modelové fronty v době, kdy je vytvořena dynamická fronta. Pokud poté použijete volání MQINQ v této frontě, správce front vrátí atributy dynamické fronty, nikoli atributy modelové fronty. Podrobnosti o tom, které atributy modelové fronty jsou zděděny dynamickou frontou, viz Tabulka 1 .
3. Je-li dotazovaný objekt alias fronta, jsou hodnoty atributů vrácené voláním MQINQ těmi z alias fronty, a nikoli z fronty základní fronty, na kterou je alias interpretován.
4. Pokud je dotazovaný objekt fronta klastru, atributy, které mohou být dotazovány, závisí na tom, jak je fronta otevřena:
   - Je-li fronta klastru otevřena pro dotaz s jedním nebo více vstupními, procházením nebo sadou, musí existovat lokální instance fronty klastru, aby byla otevřená úspěšná. V tomto případě jsou atributy, které mohou být dotazovány, platné pro lokální fronty.
   - Je-li fronta klastru otevřena pouze pro dotaz nebo dotaz a výstup, je možno se dotazovat pouze na následující atributy; atribut **QType** má v tomto případě hodnotu QTCLUS:
     - CAQD
     - CAQN
     - IADBND
     - IADPER
     - IADPRI
     - IAIPUT
     - IAQTYP

     Je-li fronta klastru otevřená bez pevné vazby (tj. OOBNDN zadané v rámci volání MQOPEN nebo OOBNDQ zadané, když má atribut **DefBind** hodnotu BNDNOT), mohou po sobě jdoucí volání MQINQ pro frontu zjišťovat různé instance fronty klastru, ačkoli všechny instance mají stejné hodnoty atributu.

   Další informace o frontách klastru najdete v tématu Konfigurace klastru správců front.

5. Pokud má být zjištěn určitý počet atributů a některé z nich mají být nastaveny pomocí volání MQSET, může být vhodné umístit na začátku polí selektoru atributy, které mají být nastaveny, aby bylo možné použít stejná pole (se sníženými počty) pro aplikaci MQSET.
6. Pokud se objeví více než jedna z varovných situací (viz parametr **CMPCOD**), vrácený kód příčiny je *první* v následujícím seznamu, který se používá:
   a. RC2068
   b. RC2022
   c. RC2008
7. Další informace o atributech objektů najdete v tématech:
   - “Atributy pro fronty” na stránce 1353
   - “Atributy pro seznamy názvů” na stránce 1381
   - “Atributy pro definice procesu v systému IBM i” na stránce 1382
   - “Atributy pro správce front v systému IBM i” na stránce 1384
8. Nová lokální fronta SYSTEM.ADMIN.COMMAND.EVENT se používá pro zprávy fronty generované při každém vydání příkazu. Zprávy se umístí do této fronty pro většinu příkazů v závislosti na tom, jak je nastaven atribut správce front CMDEV:
   - ENABLED-zprávy událostí příkazů jsou generovány a vloženy do fronty pro všechny úspěšné příkazy.
   - NODISPLAY-zprávy událostí příkazů jsou generovány a vloženy do fronty pro všechny úspěšné příkazy jiné než příkaz DISPLAY (MQSC) a příkaz Inquire (PCF).
   - DISABLED-Zprávy událostí příkazů nejsou generovány (jedná se o počáteční výchozí hodnotu správce front).

Parametry

Volání MQINQ má následující parametry:

HCONN (kladné celé číslo se znaménkem 10)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOTOBJ (10 číslic se znaménkem celého čísla)-vstup

Popisovač objektu.

Tento manipulátor představuje objekt (typu libovolného typu) s požadovanými atributy. Popisovač musí být vrácen předchozím voláním MQOPEN, které bylo určeno volbou OOINQ.

SELCNT (celé číslo se znaménkem 10 číslic)-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.

EL (10 číslic se znaménkem x SELCNT)-vstup

Pole selektorů atributů.

Jedná se o pole selektorů atributů produktu **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 se volání nezdaří s kódem dokončení CCFAIL a kódem příčiny RC2067.

Ve zvláštním případě front:

- Není-li selektor platný pro fronty typu *any* , volání selže s kódem dokončení CCFAIL a kódem příčiny RC2067.
- Je-li selektor použitelný pouze pro fronty typu nebo typů jiných typů, než je typ objektu, volání uspěje s kódem dokončení CCWARN a kódem příčiny RC2068.
- Je-li dotazovaná fronta fronta klastru, selektory, které jsou platné, závisí na tom, jak byla fronta vyřešena; viz poznámka 4 pro další podrobnosti.

Selektory mohou být zadány v libovolném pořadí. Hodnoty atributu odpovídající celočíselným selektorům atributů (selektory IA*) se vrací v produktu *INTATR* ve stejném pořadí, ve kterém se tyto selektory vyskytují v produktu *SELS*. Hodnoty atributu, které odpovídají selektorům znakových atributů (selektory CA*), se vrací v produktu *CHRATR* ve stejném pořadí, v jakém se tyto selektory vyskytují. Selektory IA* mohou být prokládané selektory CA*; důležitá je pouze relativní pořadí v rámci každého typu.

Poznámka:

1. Selektory atributů celého čísla a znaku jsou přiděleny ve dvou různých rozsazích; selektory IA* jsou umístěny v rozsahu IAFRST až IALAST a selektory CA* v rozsahu CAFRST přes CALAST.

   Pro každý rozsah definují konstanty IALSTU a CALSTU nejvyšší hodnotu, kterou správce front přijme.
2. Pokud se všechny selektory IA* vyskytnou jako první, mohou být použita stejná čísla prvků pro adresování příslušných prvků v polích *SELS* a *INTATR* .

Atributy, které mohou být dotazovány, jsou vypsány v následujících tabulkách. Pro selektory funkce CA* je konstanta, která definuje délku výsledného řetězce v řetězci *CHRATR* v závorkách, uvedena v bajtech.

Tabulka 747. Selektory atributů MQINQ pro fronty

Selektor	**Popis**	**Poznámka**
CAALTD	Datum poslední změny (LNDATE).	1
CAALTTŮV	Čas poslední změny (LNTIME).	1
KABRQN	Nadměrný název back-requue (LNQN).	5
KABAŠINA	Název fronty, jejíž alias se interpretuje jako (LNQN).	
CACFSN	Název struktury prostředku Coupling Facility (LNCFSN).	3
CACLN	Název klastru (LNCLUN).	1
KACLNCOMMENT	Seznam názvů klastru (LNNLN).	1
CAKRTD	Datum vytvoření fronty (LNCRTD).	
CAKRTTOVÁ	Čas vytvoření fronty (LNCRTT).	
KAINIQ	Název inicializační fronty (LNQN).	
CAPRON	Název definice procesu (LNPRON).	
CAQD	Popis fronty (LNQD).	
CAQN	Název fronty (LNQN).	
KARQMN	Název vzdáleného správce front (LNQMN).	
KARQN	Název vzdálené fronty, jak je známo ve vzdáleném správci front (LNQN).	
KATRGD	Data spouštěče (LNTRGD).	5
KAXQN	Název přenosové fronty (LNQN).	

Tabulka 747. Selektory atributů MQINQ pro fronty (pokračování)		
Selektor	**Popis**	**Poznámka**
IABTHR	Prahová hodnota vyřazených zpráv.	5
IACDEP	Počet zpráv ve frontě.	
IADBND	Výchozí vazba.	1
IADINP	Výchozí volba open-for-input.	5
IADPER	Výchozí trvalost zpráv.	
IADPRI	Výchozí priorita zprávy.	5
IADEFT	Typ definice fronty.	
IADRIST	Podpora distribučního seznamu.	2
IHGB	Zda se má ukrýt počet odvolání.	5
IAIGET	Zda jsou povoleny operace get.	
IAIPUT	Zda jsou povoleny operace vložení.	
IAMLEN	Maximální délka zprávy.	
IAMDEP	Maximální počet zpráv povolených ve frontě.	
IAMDY	Určuje, zda je priorita zprávy relevantní.	5
IAOÁT	Počet volání MQOPEN, které mají otevřenou frontu pro vstup.	
IAOOU	Počet volání MQOPEN, které mají otevřenou frontu pro výstup.	
IAQDHE	Řídicí atribut pro vysoké události hloubky fronty.	4, 5
IAQDHL	Horní mez hloubky fronty.	4, 5
IAQDLE	Řídicí atribut pro události nízké hloubky fronty.	4, 5
IAQDLL	Dolní mez hloubky fronty.	4, 5
IAQDME	Řídicí atribut pro maximální události hloubky 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	Obor definice fronty.	4, 5
IASHCITY	Zda lze frontu sdílet pro vstup.	
IATRAGC	Řízení spouštěče.	
IATRAGD	Hloubka spouątěče.	5
IATRGP	Priorita zprávy prahové hodnoty pro spouątěče.	5
SPRÁVA ITRGT	Typ spouątěče.	
IAUSA	Využití.	
CLWLUSEQ	Použít vzdálené fronty.	

Poznámka:

1. Podporováno na následujících platformách:
   - AIX
   - IBM i
   - Windows
   - z/OS

   a pro IBM MQ MQI clients připojené 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 z/OS.
4. z/OS Nepodporováno na z/OS.
5. Nepodporováno na VSE/ESA.

Tabulka 748. Selektory atributů MQINQ pro seznamy názvů

Selektor	Popis	Poznámka
CAALTD	Datum poslední změny (LNDATE)	1
CAALTTŮV	Čas poslední změny (LNTIME)	1
CALSTD	Popis seznamu názvů (LNNLD)	1
CALSTN	Název objektu seznamu názvů (LNNLN)	1
BANITY	Názvy v seznamu názvů (LNQN x *Number of names in the list)*	1
IANAMCZAMA	Počet názvů v seznamu názvů	1
IAQSGD	Dispozice skupiny sdílení front	3

Tabulka 749. Selektory atributů MQINQ pro definice procesu

Selektor	Popis	Poznámka
CAALTD	Datum poslední změny (LNDATE)	1
CAALTTŮV	Čas poslední změny (LNTIME)	1
CAAPPI	Identifikátor aplikace (LNPROA)	5
CAENDCOMMENT	Data prostředí (LNPROE)	5
KAPROD	Popis definice procesu (LNPROD)	5
CAPRON	Název definice procesu (LNPRON)	5
CAUSRD	Uživatelská data (LNPROU)	5
IAAPPT	Typ aplikace	5

Tabulka 749. Selektory atributů MQINQ pro definice procesu (pokračování)		
Selektor	**Popis**	**Poznámka**
IAQSGD	Dispozice skupiny sdílení front	3

Tabulka 750. Selektory atributů MQINQ pro správce front		
Selektor	**Popis**	**Poznámka**
CAALTD	Datum poslední změny (LNDATE)	1
CAALTTŮV	Čas poslední změny (LNTIME)	1
CACADX	Název uživatelské procedury automatické definice kanálu (LNEXN)	1
CACLWD	Data předávaná uživatelské proceduře pracovní zátěže klastru (LNEXDA)	1
CACLWX	Název uživatelské procedury pracovní zátěže klastru (LNEXN)	1
CACMDQ	Název vstupní fronty systémových příkazů (LNQN)	5
KADLQ	Název fronty nedoručených zpráv (LNQN)	5
KADXQN	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
FUNKCE CAQSGN	Název skupiny sdílení front (LNQSGN)	3
KARPN	Název klastru, pro který správce front poskytuje služby úložiště (LNQMN)	1
KARPNL	Název objektu seznamu názvů obsahujícího názvy klastrů, pro které správce front poskytuje služby úložiště (LNNLN)	1
CMDEV	Řídicí atribut, který určuje, zda mají být zprávy generované při vydání příkazů vloženy do fronty	8
IAAUNIT.	Řídicí atribut pro události oprávnění	4, 5
IAKAD	Řídicí atribut pro automatickou definici kanálu	2
IACADE	Řídicí atribut pro události automatické definice kanálu	2
IACLXQ.	Výchozí typ přenosové fronty klastru	4
IACL	Délka pracovní zátěže klastru	1
IACCSI	Identifikátor znakové sady	5
IAKMDL	Úroveň příkazů podporovaná správcem front	5
IACFGE	Řídicí atribut pro události konfigurace	3
IADRIST	Podpora seznamu distribuce	2
IAINHE	Řídicí atribut pro blokování událostí	4, 5
IALCLE	Řídicí atribut pro lokální události	4, 5
IAMHND.	Maximální počet popisovačů	5
IAMLEN	Maximální délka zprávy	5
IAMPRI	Maximální priorita	5

Tabulka 750. Selektory atributů MQINQ pro správce front (pokračování)		
Selektor	**Popis**	**Poznámka**
IAMUNC	Maximální počet nepotvrzených zpráv v rámci jednotky práce	5
IAPFME	Řídicí atribut pro události výkonu	4, 5
IAPLAT	Platforma, na které je správce front umístěn	5
IARTE	Řídicí atribut pro vzdálené události	4, 5
IASSE	Řídicí atribut pro události zahájení zastavení	4, 5
IASYNC	Dostupnost bodu synchronizace	5
IATRLFT	Životnost nepoužitých neadministrativních témat	
IATRIGI	Interval spouštěče	5

IACNT (celé číslo se znaménkem 10 číslic)-vstup

Počet celočíselných atributů.

Toto je počet prvků v poli INTATR . Nula je platná hodnota.

Pokud se jedná o alespoň počet selektorů IA* v parametru **SELS** , jsou vráceny všechny požadované celočíselné atributy.

INTATR (10místný podepsaný integer x IACNT)-výstup

Pole celočíselných atributů.

Toto je pole celočíselných hodnot atributů *IACNT* .

Hodnoty celočíselných atributů se vrací ve stejném pořadí jako selektory IA* v parametru **SELS** . Pokud pole obsahuje více prvků než počet selektorů IIA*, přebytečné prvky se nezmění.

Pokud HOBJ představuje frontu, ale selektor atributu se nevztahuje na tento typ fronty, specifická hodnota IAVNA se vrátí pro odpovídající prvek v poli INTATR .

CALEN (10ciferné celé číslo se znaménkem)-vstup

Délka vyrovnávací paměti atributů znaků.

Toto je délka v bajtech parametru **CHRATR** .

Musí to být alespoň součet délek požadovaných znakových atributů (viz SELS). Nula je platná hodnota.

CHRATR (1bajtový znakový řetězec x CALEN)-výstup

Atributy znaků.

Jedná se o vyrovnávací paměť, ve které jsou zřetězeny znakové atributy, zřetězené. Délka vyrovnávací paměti je dána parametrem **CALEN** .

Atributy znaků se vrací 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 vyplněna doprava s mezerami, je-li to nutné. Je-li vyrovnávací paměť větší, než je potřeba, aby obsahovala všechny požadované atributy znaků (včetně doplnění), bajty za poslední vrácenou hodnotou atributu se nezměnily.

Pokud HOBJ představuje frontu, ale selektor atributu se nevztahuje na tento typ fronty, řetězec znaků sestávající pouze z hvězdiček (*) je vrácen jako hodnota tohoto atributu v CHRATR.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (celé číslo se znaménkem 10 číslic)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud má parametr CMPCOD hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2008
(2008, X'7D8') Nedostatek prostoru povolený pro znakové atributy.

RC2022
(2022, X'7E6') Nedostatek prostoru povolený pro celočíselné atributy.

RC2068
(2068, X'814 ') Selektor není použitelný pro typ fronty.

Je-li *CMPCOD* 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 atributů znaků 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 celočíselné atributy není platné.

RC2038
(2038, X'7F6') Fronta není otevřena pro zjištění.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2065
(2065, X'811 ') Počet selektorů není platný.

RC2067
(2067, X'813 ') Selektor atributu není platný.

RC2066
(2066, X'812 ') Počet selektorů je příliš velký.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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 (Dotaz na vlastnost zprávy) v systému IBM i

Volání MQINQMP vrací hodnotu vlastnosti zprávy.

- “Syntaxe” na stránce 1299
- “Parametry” na stránce 1300
- “Deklarace RPG” na stránce 1303

Syntaxe

MQINQMP *(Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type, ValueLength, Value, DataLength, CompCode, Reason)*.

Parametry

Volání MQINQMP má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front. Hodnota *Hconn* se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **Hmsg** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu dotazu na vlastnost popisovače zprávy, jinak se volání nezdaří s chybou RC2009.

HMSG (20-digit signed integer)-vstup

Toto je popisovač zprávy, který má být dotazován. Hodnota byla vrácena předchozím voláním příkazu **MQCRTMH** .

INQOPT (MQIMPO)-vstup

Podrobnosti naleznete v datovém typu MQIMPO .

PRNAME (MQCHARV)-vstup

Tento text popisuje název vlastnosti, která se má dotázat.

Pokud nelze nalézt žádnou vlastnost s tímto názvem, volání selže s kódem příčiny RC2471.

Na konci názvu vlastnosti můžete použít znak procenta (%). Zástupný znak odpovídá žádnému znaku nebo více znakům, včetně znaku tečky (.). To umožňuje aplikaci dotazovat se na hodnotu mnoha vlastností. Volejte funkci MQINQMP s volbou IPINQF k získání první odpovídající vlastnosti a znovu s volbou IPINQN pro získání další odpovídající vlastnosti. Nejsou-li k dispozici žádné další odpovídající vlastnosti, volání selže s hodnotou RC2471. Pokud je pole *ReturnedName* ve struktuře Opts InqPropinicializováno s adresou nebo offsetem pro vrácený název vlastnosti, je tento proces dokončen při návratu z MQINQMP s názvem vlastnosti, která se shoduje. Je-li pole *VSBufSize* v poli *ReturnedName* ve struktuře InqPropOpts menší než délka vráceného názvu vlastnosti, kód dokončení je nastaven CCFAIL s příčinou RC2465.

Vlastnosti, které mají známá synonyma, se vrátí takto:

1. Vlastnosti s předponou "mqps." jsou vráceny spolu s názvem vlastnosti IBM MQ . Například "MQTopicString" je spíše vrácený název než "mqps.Top".
2. Vlastnosti s předponou "jms." nebo "mcd." se vrátí 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 vrácen spíše než "usr.Color".

Vlastnosti se synonymy jsou vráceny pouze jednou.

V programovacím jazyce RPG jsou definovány následující proměnné makra určené pro dotazy na všechny vlastnosti a všechny vlastnosti začínající "usr.":

INQALL
Dotaz na všechny vlastnosti zprávy.

INQUSR
Zjišťovat všechny vlastnosti zprávy, které spouští "usr.". Vrácený název je vrácen bez parametru "usr." .

Je-li uvedeno IPINQN, ale název se změnil od předchozího volání, nebo je to první volání, pak IPINQF je implikovaná.

Další informace o použití názvů vlastností naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

PRPDSC (MQPD)-výstup

Tato struktura se používá k definování atributů vlastnosti, včetně toho, co se stane, pokud tato vlastnost není podporována, jaký kontext zprávy vlastnost patří a do jakých zpráv má být vlastnost zkopírována. Podrobnosti o této struktuře viz MQPD .

TYPE (10ciferné celé číslo se znaménkem)-vstupní/výstupní

Při návratu z volání MQINQMP je tento parametr nastaven na datový typ *Hodnota*. Datový typ může být libovolný z následujících:

TYBOL
Booleovský.

TYPBST
bajtový řetězec.

TYPI8
8bitové podepsané celé číslo.

TYPI16
16bitové podepsané celé číslo.

TYPI32
32bitové celé číslo se znaménkem.

TYPI64
64bitové podepsané celé číslo.

TYPF32
32-bitové číslo s pohyblivou řádovou čárkou.

TYPF64
64-bitové číslo s pohyblivou řádovou čárkou.

TYPSTR
Znakový řetězec.

TYPNUL
Vlastnost existuje, ale má hodnotu null.

Není-li datový typ hodnoty vlastnosti rozpoznán, je vrácen parametr TYPSTR a do oblasti *Hodnota* se umístí řetězcová reprezentace hodnoty. Řetězcovou reprezentaci datového typu lze nalézt v poli *IPTYP* parametru *IPOPT* . Kód dokončení varování je vrácen s kódem příčiny RC2467.

Navíc, je-li zadána volba IPCTYP, převod hodnoty vlastnosti je požadován. Použijte *Typ* jako vstup pro uvedení datového typu, který má vlastnost vracet jako. Podrobné informace o převodu datového typu naleznete v popisu volby IPCTYP v “MQIMPO (Inquire message property options) na IBM i” na stránce 1092 .

Pokud nevyžadujete převod typu, můžete na vstupu použít následující hodnotu:

PŘETYPOVÁ
Hodnota vlastnosti je vrácena bez převodu jeho datového typu.

VALLEN (10ciferné celé číslo se znaménkem)-vstup

Délka v bajtech oblasti Hodnota.

Uveďte nulu pro vlastnosti, pro které není požadována vrácená hodnota. Mohou to být vlastnosti, které jsou navrženy aplikací, aby měly hodnotu null nebo prázdný řetězec. Také uveďte nulu, pokud byla zadána volba IPQLEN; v tomto případě není vrácena žádná hodnota.

VALUE (1-bytový bit stringxVALLEN)-výstup

Toto je oblast, která má obsahovat dotazovanou hodnotu vlastnosti. Vyrovnávací paměť by měla být zarovnána na hranici vhodnou pro vracenou hodnotu. Pokud tak neučiníte, může to vést k chybě při pozdějším přístupu k této hodnotě.

Pokud je hodnota vlastnosti *VALLEN* menší než délka hodnoty vlastnosti, hodnota vlastnosti je přesunuta do hodnoty *VALUE* a volání selže s kódem dokončení CCFAIL a příčinou RC2469.

Znaková sada dat v hodnotě *VALUE* je dána polem IPRETCSI v parametru INQOPT. Kódování dat v hodnotě *VALUE* je dáno polem IPRETENC v parametru INQOPT.

Je-li parametr *VALLEN* nula, hodnota VALUE se neoznačuje.

DATLEN (10ciferné celé číslo se znaménkem)-výstup

Jedná se o délku skutečné hodnoty vlastnosti v bajtech, jak je vráceno v oblasti *Hodnota* .

Je-li hodnota *DataLength* menší než délka hodnoty vlastnosti, *DataLength* se stále zadává při návratu z volání MQINQMP. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné k umístění hodnoty vlastnosti, a pak znovu zadejte volání s vyrovnávací pamětí příslušné velikosti.

Mohou být vráceny také následující hodnoty.

Je-li parametr *Type* nastaven na hodnotu TYPSTR nebo TYPBST, postupujte takto:

VLEP
Vlastnost existuje, ale neobsahuje žádné znaky ani bajty.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CompCode*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* CCWARN:

RC2492
(2492, X'09BC') Vrácený název vlastnosti není 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 nebylo možné analyzovat.

Je-li *CMPCOD* CCFAIL:

RC2204
(2204, X'089C') Adaptér není k dispozici.

RC2130
(2130, X'0852 ') Nelze načíst modul služby adaptéru.

RC2157
(2157, X'086D') Primární a domovské ASID se liší.

RC2004
(2004, X'07D4') Hodnota parametru hodnoty není platná.

RC2005
(2005, X'07D5') Hodnota parametru délky hodnoty není platná.

RC2219
(2219, X'08AB') Volání MQI bylo zadáno 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') Dotaz na strukturu voleb vlastností zprávy není platný.

RC2460
(2460, X'099C') Popisovač zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

RC2064
(2046, X'07F8') Volby nejsou platné nebo nejsou konzistentní.

RC2482
(2482, X'09B2') Struktura deskriptoru vlastností 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 vracenou vyrovnávací paměť názvu.

RC2471
(2471, X'09A7) Vlastnost není k dispozici.

RC2469
(2469, X'09A5') Hodnota vlastnosti je příliš velká pro oblast Hodnota.

RC2472
(2472, X'09A8') Chyba formátu čísla zjištěna 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 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'0893 ') Vyskytla se neočekávaná chyba.

Podrobné informace o těchto kódech najdete v tématech:

- položky IBM MQ for z/OS zprávy, dokončení, a kódy příčiny pro IBM MQ for z/OS
- Zprávy a kódy příčiny pro všechny ostatní platformy IBM MQ

Deklarace 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 (Převod ovladače zpráv do vyrovnávací paměti) v systému IBM i

Hodnota MQMHBUF převádí popisovač zprávy do vyrovnávací paměti a je inverzní k volání MQBUFMH.

- "Syntaxe" na stránce 1304
- "Poznámky k použití" na stránce 1304
- "Parametry" na stránce 1304
- "Deklarace RPG" na stránce 1306

Syntaxe

MQMHBUF *(Hconn, Hmsg, MsgHBufOpts, Name, MsgDesc, BufferLength, Buffer, DataLength, CompCode, Reason)*

Poznámky k použití

MQMHBUF převádí popisovač zprávy do vyrovnávací paměti.

Můžete ji použít s uživatelskou procedurou rozhraní API MQGET k přístupu k určitým vlastnostem, pomocí rozhraní API vlastností zpráv, a poté tyto vlastnosti předat do vyrovnávací paměti zpět do aplikace určené k použití záhlaví MQRFH2 namísto obslužných rutin zpráv.

Toto volání je inverzní k volání MQBUFMH, které lze použít k analýze vlastností zpráv z vyrovnávací paměti do manipulátorů zpráv.

Parametry

Volání MQMHBUF má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota *HCONN* se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podvláknu, které odstraňuje popisovač zprávy. Není-li ustanoveno platné připojení, volání selže s chybou RC2009.

HMSG (20-digit signed integer)-vstup

Tento úchyt je popisovač zprávy, 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 (Vyrovnávací paměť pro volby zpracování vyrovnávací paměti) v systému IBM i" na stránce 1008.

PRNAME (MQCHARV)-vstup

Název vlastnosti nebo vlastností, které mají být vloženy do vyrovnávací paměti.

Není-li nalezena žádná vlastnost odpovídající názvu, volání selže s hodnotou RC2471.

zástupné znaky

Můžete použít zástupný znak pro vložení více než jedné vlastnosti do vyrovnávací paměti. Chcete-li to provést, použijte znak procenta (%) na konci názvu vlastnosti. Tento zástupný znak odpovídá nule nebo více znakům, včetně znaku tečky (.).

Další informace o použití názvů vlastností naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

MSGDSC (MQMD)-vstupní/výstupní

Struktura *MSGDSC* popisuje obsah oblasti vyrovnávací paměti.

Na výstupu jsou pole *Encoding*, *CodedCharSetId* a *Format* nastavena tak, aby správně popisovala kódování, identifikátor znakové sady a formát dat v oblasti vyrovnávací paměti tak, jak je zapsaly volání.

Data v této struktuře se nacházejí ve znakové sadě a kódování aplikace.

BUFLEN (10ciferné číslicové celé číslo)-vstup

BUFFLEN je délka oblasti vyrovnávací paměti, v bajtech.

BUFFER (1-bytový bitový řetězec x BUFLEN)-vstupní/výstupní

BUFFER definuje oblast obsahující vyrovnávací paměť zpráv. Pro většinu dat musíte zarovnat vyrovnávací paměť na 4bajtové hranici.

Pokud *BUFFER* obsahuje znaková nebo číselná data, nastavte pole *CodedCharSetId* a *Encoding* v parametru **MSGDSC** na hodnoty odpovídající datům; to umožní převod dat, je-li to nutné.

Jsou-li vlastnosti nalezeny ve vyrovnávací paměti zpráv, mohou být odebrány později. Později budou k dispozici od obslužné rutiny zprávy při návratu z volání.

V programovacím jazyku C je parametr deklarován jako ukazatel-to-void, což znamená, že adresa libovolného typu dat může být zadána jako parametr.

Pokud je argument **BUFLEN** nastaven na nulu, *BUFFER* se na něj neodkazuje. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

DATLEN (10ciferné celé číslo se znaménkem)-výstup

DATLEN je délka vrácených vlastností ve vyrovnávací paměti v bajtech. Je-li hodnota nula, žádné vlastnosti se neshodují s hodnotou uvedenou v *PRNAME* a volání selže s kódem příčiny RC2471.

Je-li *BUFLEN* menší než délka požadovaná pro uložení vlastností ve vyrovnávací paměti, volání MQMHUF selže s chybou RC2469, ale hodnota je stále zadána do *DATLEN*. To umožňuje aplikaci určit velikost vyrovnávací paměti potřebné pro přizpůsobení vlastností a pak znovu zadejte volání s požadovanou *BUFLEN*.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *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 domovské ASID se liší.

RC2501
(2501, X'095C') Popisovač zprávy pro strukturu vyrovnávací paměti není platný.

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 bylo zadáno 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 zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

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, aby mohla obsahovat zadané vlastnosti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Deklarace 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) v systému IBM i

Volání MQOPEN vytváří přístup k objektu.

Platné jsou tyto typy objektů:

- Fronta (včetně distribučních seznamů)
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Index

Syntaxe

MQOPEN *(HCONN, OBJDSC, OPTS, HOBJ, CMPCOD, REASON)*

Poznámky k použití

1. Otevíraný objekt je jeden z následujících:
   - Fronta, s cílem:
     - Získat nebo procházet zprávy (pomocí volání MQGET)
     - Vložit zprávy (pomocí volání MQPUT)
     - Dotazovat se na atributy fronty (pomocí volání MQINQ)
     - Nastavení atributů fronty (pomocí volání MQSET)

     Je-li uvedená fronta modelová fronta, vytvoří se dynamická lokální fronta.

Rozdělovník je speciální typ objektu fronty, který obsahuje seznam front. Lze ji otevřít pro vkládání zpráv, nikoli však k získání nebo procházení zpráv nebo k zjišťování či nastavení atributů. Další podrobnosti najdete v poznámce pod čarou 8.

Fronta, která má `QSGDISP(GROUP)` , je speciální typ definice fronty, kterou nelze použít s voláními MQOPEN nebo MQPUT1 .

- Seznam názvů, v němž lze provést následující akce:
  - Zjišťuje se o názvech front v seznamu (pomocí volání MQINQ).
- Definice procesu, s cílem:
  - Dotaz na atributy procesu (pomocí volání MQINQ).
- Správce front, aby:
  - Dotaz na atributy lokálního správce front (pomocí volání MQINQ).

2. Je platný, aby aplikace otevřela stejný objekt více než jednou. Pro každé otevření je vrácen jiný popisovač objektu. Každý vrácený popisovač může být použit pro funkce, pro které bylo provedeno odpovídající otevření.
3. Je-li otevíraný objekt fronta, ale ne fronta klastru, v době volání MQOPEN se v době volání MQOPEN použije všechna rozpoznání názvu v lokálním správci front. To může zahrnovat jednu nebo více z následujících akcí pro konkrétní volání MQOPEN:
   - Rozlišení aliasu pro název základní fronty
   - Vyřešení názvu lokální definice vzdálené fronty na název vzdáleného správce front a název, pod kterým je fronta známa ve vzdáleném správci front
   - Rozlišení názvu vzdáleného správce front na název lokální přenosové fronty

   Mějte však na paměti, že následující volání MQINQ nebo MQSET pro manipulátor se týkají výhradně názvu, který byl otevřen, a nikoli objektu, který je výsledkem rozlišení názvu. Je-li například otevřený objekt alias, jsou atributy vrácené voláním MQINQ atributy aliasu, nikoli atributy základní fronty, na které je alias interpretováno. Kontrola rozlišení názvů je stále prováděna, bez ohledu na to, co je určeno pro parametr **OPTS** v odpovídající MQOPEN.

   Je-li otevíraný objekt fronta klastru, může v době volání MQOPEN dojít k rozpoznání názvu nebo může být odloženo na později. Bod, ve kterém dochází k vyřešení problému, je řízen volbami OOBND* uvedeným v rámci volání MQOPEN:

   - OOBNDO
   - OOBNDN
   - OOBNDQ

   Další informace o rozlišování názvů pro fronty klastru najdete v tématu Rozlišování názvů .
4. Atributy objektu se mohou změnit, zatímco aplikace má otevřený objekt. V mnoha případech aplikace toto nezaznamenaní, ale u určitých atributů správce front označí popisovač jako již platný. Patří mezi ně:
   - Jakýkoli atribut, který ovlivňuje rozpoznání názvu objektu. To platí bez ohledu na použité otevřené volby a zahrnuje následující:
     - Změna na atribut **BaseQName** fronty aliasů, která je otevřená.
     - Změna atributů fronty **RemoteQName** nebo **RemoteQMgrName** pro všechny obslužné rutiny, které jsou otevřeny pro tuto frontu, nebo pro frontu, která se překládá prostřednictvím této definice jako alias správce front.
     - Jakákoli změna, která způsobí, že se aktuálně otevřený popisovač vzdálené fronty vyřeší na jinou frontu *transmission* , nebo aby se nevyhodnocla vůbec. To může například zahrnovat:
       - Změna atributu **XmitQName** lokální definice vzdálené fronty bez ohledu na to, zda je definice použita pro frontu nebo pro alias správce front.

Je zde jedna výjimka, konkrétně vytvoření nové přenosové fronty. Popisovač, který by byl interpretován jako tato fronta, byl při otevření popisovače přítomen, ale namísto toho vyřešen do výchozí přenosové fronty, není platný.

– Změna na atribut správce front produktu **DefXmitQName** . V tomto případě jsou všechny otevřené popisovače, které se vyřešily do dříve pojmenované fronty (které se na něj budou interpretovat pouze proto, že se jednalo o výchozí přenosovou frontu), označeny jako neplatné. Ošetřeny, které byly pro tuto frontu rozpoznáné z jiných důvodů, nejsou ovlivněny.

- Atribut fronty **Shareability** , pokud existují dva nebo více manipulátorů, které momentálně poskytují OOINPS přístup pro tuto frontu, nebo pro frontu, která je interpretována do této fronty. Pokud ano, *všechny* popisovače, které jsou otevřeny pro tuto frontu, nebo pro frontu, které se překládá do této fronty, jsou označeny jako neplatné, bez ohledu na volby otevření.
- Atribut fronty produktu **Usage** pro všechny manipulátory, které jsou otevřeny pro tuto frontu, nebo pro frontu, která se interpretuje jako tato fronta bez ohledu na volby otevření.

Je-li popisovač označen jako neplatný, všechna následná volání (jiná než MQCLOSE) používající tento manipulátor selžou s kódem příčiny RC2041; , aplikace by měla vydat volání MQCLOSE (pomocí původní obslužné rutiny) a poté znovu otevřít frontu. Všechny nepotvrzené aktualizace oproti starému popisovači z předchozích úspěšných volání lze stále potvrdit nebo odstranit, jak to 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 vyvolání volání MQOPEN, aby bylo možné ověřit, zda má identifikátor uživatele, pod kterým je aplikace spuštěna, příslušnou úroveň oprávnění, než je povolen přístup. Kontrola oprávnění se provádí na jméno objektu, který je otevíraný, a ne na názvu nebo názvech, výsledkem je, že byl vyřešen název.

   Je-li otevíraný objekt modelová fronta, provede správce front úplnou kontrolu zabezpečení proti názvu modelové fronty a názvu vytvořené dynamické fronty. Je-li výsledná dynamická fronta otevřena explicitně, provede se další kontrola zabezpečení prostředku proti názvu dynamické fronty.

6. Vzdálenou frontu lze zadat jedním ze dvou způsobů v parametru **OBJDSC** tohoto volání (viz pole *ODON* a *ODMN* popsaná v části “MQOD (Object descriptor) na systému IBM i” na stránce 1147):

   - Uvedením *ODON* název lokální definice vzdálené fronty. V tomto případě *ODMN* odkazuje na lokálního správce front a lze jej zadat jako mezery.

     Ověření zabezpečení provedené lokálním správcem front ověřuje, zda je uživatel oprávněn k otevření lokální definice vzdálené fronty.
   - Uvedením *ODON* název vzdálené fronty, jak je známo vzdálenému správci front. V tomto případě je *ODMN* názvem vzdáleného správce front.

     Ověření zabezpečení provedené lokálním správcem front ověřuje, zda je uživatel autorizován k odesílání zpráv do přenosové fronty, která je výsledkem procesu rozlišování názvů.

   V obou případech:

   - Lokální správce front odesílá do správce vzdálené fronty žádné zprávy, aby bylo možné zkontrolovat, 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 vytvořil, není autorizován.

7. Volání MQOPEN s volbou OOBRW ustanoví kurzor procházení, pro použití s voláními MQGET, které určují popisovač objektu a jednu z voleb procházení. To umožňuje skenování fronty, aniž by došlo ke změně jejího obsahu. Zpráva, která byla vyhledána procházením, může být později odstraněna z fronty pomocí volby GMMUC.

   Více kurzorů procházení může být aktivní pro jednu aplikaci vysláním 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 řetězec nebo řetězec s hodnotou null.
  - *ODMN* musí být prázdný řetězec nebo řetězec s hodnotou null.
  - *ODREC* musí být větší než nula.
  - Jeden z produktů *ODORO* a *ODORP* musí být nula a druhý nenulový.
  - Ne více než jeden z *ODRRO* a *ODRRP* může být nenulový.
  - Musí existovat *ODREC* záznamů objektů adresovaných buď *ODORO* nebo *ODORP*. Záznamy objektů musí být nastaveny na názvy cílových front, které se mají otevřít.
  - Je-li některý z produktů *ODRRO* a *ODRRP* nenulový, musí být přítomny záznamy odpovědí *ODREC* . Jsou nastavována správcem front, pokud je volání dokončeno s kódem příčiny RC2136.

  MQOD version-2 lze také použít k otevření jedné fronty, která není v distribučním seznamu, tím, že zajistíte, že *ODREC* je nula.
- V parametru **OPTS** jsou platné pouze následující volby otevření:
  - OOUT
  - OOPAS *
  - OSADA *
  - OOALTU@
  - OOFIQ
- Cílové fronty v rozdělovníku mohou být lokální, alias nebo vzdálené fronty, ale nemohou být modelové fronty. Je-li zadána modelová fronta, tato fronta se neotevřou, s kódem příčiny RC2057. To však nezabrání tomu, aby byly ostatní fronty v seznamu úspěšně otevřeny.
- Kód dokončení a parametry kódu příčiny jsou nastaveny takto:
  - Pokud jsou operace otevření pro fronty v seznamu distribuce úspěšné nebo selžou stejným způsobem, jsou nastaveny parametry dokončení kódu dokončení a kódu příčiny popisující společný výsledek. Záznamy odpovědí MQRR (nejsou-li zadány aplikací) nejsou v tomto případě nastaveny.

    Je-li například každé otevření úspěšné, kód dokončení je nastaven na CCOK a kód příčiny je RCNONE; pokud každé otevření selže, protože žádná z front neexistuje, parametry jsou nastaveny na CCFAIL a RC2085.
  - Pokud operace otevření pro fronty v rozdělovníku nejsou všechny úspěšné nebo selžou stejným způsobem:
    - Kód dokončení je nastaven na CCWARN, pokud byl alespoň jeden úspěšně otevřen, a do CCFAIL, pokud se všechny nezdařily.
    - Parametr kódu příčiny je nastaven na hodnotu RC2136.
    - Záznamy odpovědí (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.
- Když byl distribuční seznam úspěšně otevřen, popisovač *HOBJ* vrácený voláním lze použít při následných voláních MQPUT k vložení zpráv do front v rozdělovníku a na volání MQCLOSE, aby se uvoll přístup k rozdělovníku. Jediná platná volba zavření pro rozdělovník je CONONE.

  Volání MQPUT1 lze také použít k vložení zprávy do distribučního seznamu; struktura MQOD, která definuje fronty v seznamu, je uvedena jako parametr v tomto volání.
- Každý úspěšně otevřený cíl v distribučním seznamu se počítá jako *samostatný* při kontrole, zda aplikace překročila maximální povolený počet popisovačů (viz atribut správce front **MaxHandles**). To platí i v případě, že se dvě nebo více míst určení v seznamu distribucí skutečně vyřeší do stejné fyzické fronty. Pokud volání MQOPEN nebo MQPUT1 pro distribuční seznam způsobí, že počet popisovačů v aplikaci převýší *MaxHandles*, volání selže s kódem příčiny RC2017.

- Každé místo určení, které je úspěšně otevřeno, má hodnotu jeho atributu **OpenOutputCount** inkrementované o jednu. Pokud se dvě nebo více míst určení v seznamu distribucí skutečně vyřeší do stejné fyzické fronty, má tato fronta svůj atribut **OpenOutputCount** inkrementován počtem míst určení v seznamu distribucí, který se do této fronty rozejde.
- Jakákoli změna definic front, která by způsobila, že se popisovač stanou neplatnými, byly fronty otevřeny jednotlivě (například změna v cestě vyřešení), nezpůsobí zneplatnění rozdělovníku pro seznam distribucí. Výsledkem je však selhání této konkrétní fronty, je-li popisovač distribučního seznamu použit v následném volání MQPUT.
- Je platný, aby rozdělovník obsahoval pouze jedno místo určení.

9. Pro použití klastrových front se používají následující poznámky.
   - Je-li poprvé otevřena fronta klastru a lokální správce front není správce front úplného úložiště, obdrží lokální správce front informace o frontě klastru ze správce front úplného úložiště. Je-li síť zaneprázdněna, může správce lokální fronty přijmout několik sekund, aby obdržel potřebné informace od správce front úložiště. V důsledku toho může aplikace, která vydala volání MQOPEN, čekat až 10 sekund, než se řízení vrátí z volání MQOPEN. Pokud lokální správce front v rámci této doby neobdrží potřebné informace o frontě klastru, volání selže s kódem příčiny RC2189.
   - Když se otevře fronta klastru a v klastru je více instancí fronty, instance se skutečně otevře v závislosti na volbách uvedených v volání MQOPEN:
     – Pokud uvedené volby zahrnují některou z následujících voleb:
       - OOBRW
       - OOINPQ
       - OOINPX
       - OOINPS
       - OSADA

       Instance otevřené fronty klastru je nezbytná jako lokální instance. Pokud zde není žádná lokální instance fronty, volání MQOPEN selže.
     – Pokud uvedené volby nezahrnují žádnou z výše uvedených voleb, ale zahrnují jednu nebo obě z následujících možností:
       - OOINQ
       - OOUT

       otevřená instance je lokální instance, pokud existuje jedna, a vzdálená instance jinak. Instance zvolená správcem front však může být změněna uživatelskou procedurou pracovní zátěže klastru (je-li k tomu nějaká).

   Další informace o frontách klastru najdete v tématu Klastrové fronty.
10. Aplikace spuštěné monitorem spouštěčů jsou předány názvu fronty přidružené k aplikaci při spuštění aplikace. Tento název fronty může být zadán v parametru **OBJDSC** pro otevření fronty. Další podrobnosti naleznete v popisu struktury MQTMC.
11. Je-li použita volba OORLOQ, lokální fronta je již vrácena, pokud je otevřena buď lokální, alias nebo modelová fronta, ale v tomto případě se nejedná o případ, kdy je například otevřena vzdálená fronta nebo jiná než lokální fronta klastru. Název ResolvedQName a ResolvedQMgrse zadávají s názvem RemoteQName a RemoteQMgrNalezený název ve vzdálené definici fronty nebo podobně s vybranou vzdálenou frontou klastru. Je-li hodnota OORLOQ zadána při otevírání, například vzdálená fronta, ResolvedQName bude nyní přenosová fronta, do které budou zprávy vloženy. Název ResolvedQMgrbude zadán spolu s názvem lokálního správce front, který je hostitelem přenosové fronty. Je-li uživatel autorizován pro procházení, vstup nebo výstup ve frontě, mají oprávnění k zadání tohoto příznaku v rámci volání MQOPEN. Není třeba žádné zvláštní oprávnění.

Parametry

Volání MQOPEN má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OBJDSC (MQOD)-vstup/výstup

Deskriptor objektu.

Jedná se o strukturu, která identifikuje objekt, který má být otevřen; podrobnosti viz “MQOD (Object descriptor) na systému IBM i” na stránce 1147 .

Je-li pole *ODON* v parametru **OBJDSC** název modelové fronty, je dynamická lokální fronta je vytvořen s atributy modelové fronty; to se stává bez ohledu na volby otevření zadané argumentem **OPTS** . Následné operace používající příkaz *HOBJ* vrácené voláním MQOPEN jsou prováděny v nové dynamické frontě a nikoli ve frontě modelu. To platí i pro volání MQINQ a MQSET. Název modelové fronty v parametru **OBJDSC** se nahradí názvem vytvořené dynamické fronty. Typ dynamické fronty je určen hodnotou atributu **DefinitionType** v modelové frontě (viz “Atributy pro fronty” na stránce 1353). Informace o možnostech zavření použitelných pro dynamické fronty naleznete v popisu volání MQCLOSE.

OPTS (10ciferné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQOPEN.

Musí být uvedena alespoň jedna z následujících voleb:

- OOBRW
- OOINP* (pouze jeden z nich)
- OOINQ
- OOUT
- OSADA
- OORLQ

Další volby lze zadat podle potřeby. Je-li požadována více než jedna volba, lze hodnoty přidat (nepřidávat stejnou konstantu víckrát než jednou). Kombinace, které nejsou platné, jsou zaznamenány; všechny ostatní kombinace jsou platné. Povoleny jsou pouze volby, které jsou použitelné na typ objektu určeného parametrem *OBJDSC* (viz Platné volby MQOPEN pro každý typ fronty).

Volby přístupu: Následující volby řídí typ operací, které lze na objektu provést:

OOINPQ
Chcete-li získat zprávy pomocí výchozího nastavení fronty, otevřete frontu.

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 1353 .

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

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 , je-li fronta momentálně otevřená s OOINPX.

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

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 , je-li fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (OOINPS nebo OOINPX).

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty.

Pro tyto volby platí následující poznámky:

- Může být uvedena pouze jedna 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 následující volání MQGET selžou, zatímco je atribut nastaven na tuto hodnotu).
- Je-li fronta definovaná jako nesdílitelná (tedy atribut fronty **Shareability** má hodnotu QANSHR), pokusí se otevřít frontu pro sdílený přístup jako pokusy o otevření fronty s výlučným přístupem.
- Je-li alias fronta otevřena s jednou z těchto voleb, test pro výhradní použití (nebo pro to, zda má výlučnému použití jiná aplikace) je proti základní frontě, na kterou je alias interpretováno.
- Tyto volby nejsou platné, pokud *ODMN* je název alias správce front; to je pravda i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias 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:

- GMBRWFCH.
- GMBRWN
- GMBRWC

To je povoleno i v případě, že je fronta aktuálně otevřena pro OOINPX. Volání MQOPEN s volbou OOBRW vytvoří kurzor procházení a umístí jej logicky před první zprávou ve frontě. Další informace naleznete v poli *GMOPT* , které je popsáno v části “MQGMO (volby získání zpráv) v systému IBM i” na stránce 1066 .

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty. Je také neplatný, je-li *ODMN* název alias správce front; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

OOUT
Otevřete frontu pro vložení zpráv, nebo téma nebo 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 (ačkoli následné volání MQPUT se nezdaří, když je atribut nastaven na tuto hodnotu).

Tato volba je platná pro všechny typy front, včetně distribučních seznamů a témat.

OOINQ
Otevřít objekt k dotazu na atributy.

Fronta, seznam názvů, definice procesu nebo správce front je otevřen pro použití s dalšími voláními MQINQ.

Tato volba je platná pro všechny typy objektů jiných než distribuční seznamy. Není platná, pokud *ODMN* je název alias správce front; to platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

OSADA
Otevřete frontu pro nastavení atributů.

Fronta je otevřena pro použití s následnými voláními MQSET.

Tato volba je platná pro všechny typy front jiných než distribučních seznamů. Není platný, je-li *ODMN* název lokální definice vzdálené fronty. To platí i v případě, že hodnota atributu **RemoteQMgrName** v lokální definici vzdálené fronty použité pro alias správce front je názvem lokálního správce front.

Volby vázání: Při otevírání objektu z fronty klastru se používají následující volby: tyto volby řídí vázání manipulátoru fronty k instanci fronty klastru:

OOBNDO
Svázat popisovač do cíle při otevření fronty.

To způsobí, že lokální správce front sváže popisovač fronty s instancí cílové fronty při otevření fronty. V důsledku toho jsou všechny zprávy používající tento popisovač odeslány do stejné instance cílové fronty a stejnou přenosovou cestou.

Tato volba je platná pouze pro fronty a má vliv pouze na fronty klastru. Je-li tato volba zadána pro frontu, která není frontou klastru, je tato volba ignorována.

OOBNDN
Nepřipojujte se k určitému místu určení.

Tím se zastaví lokální správce front s vazbou manipulátoru fronty na instanci cílové fronty. Výsledkem je, že po sobě jdoucí volání MQPUT používající tento manipulátor mohou způsobit, že se zprávy odesílají do *různých* instancí cílové fronty nebo jsou odeslány do stejné instance, ale různými cestami. Umožňuje také, aby byla instance vybrána později lokálním správcem front, vzdáleným správcem front nebo agentem MCA (Message Channel Agent) v souladu se 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 do různých instancí serverové aplikace.

Je-li OOBRW nebo jedna z voleb OOINP* uvedena pro frontu klastru, správce front je nucen vybrat lokální instanci fronty klastru. V důsledku toho je vazba manipulátoru fronty opravena, a to i v případě, že je zadán objekt OOBNDN.

Je-li položka OOINQ uvedena s rozhraním OOBNDN, mohou následné volání MQINQ pomocí tohoto manipulátoru zjišťovat různé instance fronty klastru, ačkoli všechny instance mají stejné hodnoty atributu.

Hodnota OOBNDN je platná pouze pro fronty a má vliv pouze na fronty klastru. Je-li tato volba zadána pro frontu, která není frontou klastru, je tato volba ignorována.

OOBNDQ
Použít výchozí vazbu pro frontu.

To způsobí, že lokální správce front sváže manipulátor fronty tak, jak je definován atributem fronty **DefBind** . Hodnota tohoto atributu je buď BNDOPN nebo BNDNOT.

OOBNDQ je výchozí, nejsou-li zadány OOBNDO a OOBNDN.

OOBNDQ je definován v dokumentaci programu pomoci. Není určeno, že tato volba se používá při použití jedné z dalších dvou voleb vázání, ale protože její hodnota je nula, nelze takové použití detekovat.

Volby kontextu: Následující volby řídí zpracování kontextu zprávy:

OOSAVSKY
Uložit kontext při načítání zprávy.

Informace o kontextu jsou přidruženy k tomuto manipulátoru fronty. Tyto informace jsou nastaveny z kontextu jakékoli zprávy načtené pomocí tohoto popisovače. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tyto informace o kontextu mohou být předány ke zprávě, která je později vložena do fronty pomocí volání MQPUT nebo MQPUT1 . Viz volby PMPASI a PMPASA popsané v části “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161.

Do doby, kdy byla zpráva úspěšně načtena, nelze předat kontext do fronty, která je vložena do fronty.

Zpráva načtená pomocí jedné z voleb procházení GMBRW* nemá uložené informace o kontextu (ačkoli pole kontextu v parametru **MSGDSC** se nastavují po procházení).

Tato volba je platná pouze pro lokální fronty, alias a modelové fronty; není platná pro vzdálené fronty, distribuční seznamy a objekty, které nejsou fronty. Musí být zadána jedna z voleb OOINP*.

OOPASI
Povolit předávání kontextu identity.

Toto povoluje volbu PMPASI, která má být uvedena v parametru **PMO** , když je zpráva vložena do fronty; to dává zprávě informace o kontextu identity ze vstupní fronty, která byla otevřena pomocí volby OOSAVA. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

OOPASA
Povolit předávání všech kontextů.

To umožňuje uvedení volby PMPASA do parametru **PMO** , když je zpráva vložena do fronty; to dává zprávě informace o kontextu identity a původu z vstupní fronty, která byla otevřena pomocí volby OOSAVA. Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba určuje volbu OOPASI, která proto nemusí být zadána. Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

OOSETI
Povolit nastavení kontextu identity.

To umožňuje uvedení volby PMSETI do parametru **PMO** , když je zpráva vložena do fronty; to dává zprávě informace o kontextu identity obsažené v parametru **MSGDSC** uvedeném na volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba určuje volbu OOPASI, která proto nemusí být zadána. Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

OOSETA
Povolit nastavení veškerého kontextu.

To umožňuje uvedení volby PMSETA do parametru **PMO** , když je zpráva vložena do fronty; to dává zprávě informace o identitě a zdroji původu obsažené v parametru **MSGDSC** uvedeném na volání MQPUT nebo MQPUT1 . Další informace o kontextu zprávy najdete v tématu Kontext zprávy a Informace o řízení kontextu.

Tato volba zahrnuje následující volby, které proto nemusí být zadány:

- OOPASI
- OOPASA
- OOSETI

Musí být zadána volba OOOUT.

Tato volba je platná pro všechny typy front, včetně distribučních seznamů.

Další volby: Následující volby kontrolují kontrolu autorizace, a to, co se stane, když správce front přechází do klidového stavu:

OOALTU@
Validovat s uvedeným identifikátorem uživatele.

To znamená, že pole *ODAU* v parametru **OBJDSC** obsahuje identifikátor uživatele, který má být použit pro ověření tohoto volání MQOPEN. Volání může být úspěšné pouze v případě, že je *ODAU* autorizován k otevření objektu s uvedenými volbami přístupu, bez ohledu na to, zda je identifikátor uživatele, pod kterým je aplikace spuštěna, oprávněn tak učinit. To však neplatí pro žádné zadané volby kontextu, které jsou však vždy zkontrolovány proti identifikátoru uživatele, pod kterým je aplikace spuštěna.

Tato volba je platná pro všechny typy objektů.

OOFIQ
Selhání, pokud je správce front uváděn do klidového stavu.

Tato volba vynutí selhání volání MQOPEN, pokud se správce front nachází 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 by měla být v rámci struktury MQOD (je-li k dispozici) zadána hodnota ResolvedQName s názvem lokální fronty, která byla otevřena. Název ResolvedQMgrbude podobně zadán spolu s názvem lokálního správce front, který je hostitelem lokální fronty.

Tabulka 751. Platné volby MQOPEN pro každý typ fronty

Volba	Alias ("1" na stránce 1317)	Lokální a model	Vzdálený	Nelokální klastr	Distribuční seznam	Téma
OOINPQ	✓	✓	-	-	-	-
OOINPS	✓	✓	-	-	-	-
OOINPX	✓	✓	-	-	-	-
OOBRW	✓	✓	-	-	-	-
OOUT	✓	✓	✓	✓	✓	✓
OOINQ	✓	✓	"2" na stránce 1317	✓	-	-
OSADA	✓	✓	"2" na stránce 1317	-	-	-
OOBNDO ("3" na stránce 1317)	✓	✓	✓	✓	✓	-
OOBNDN ("3" na stránce 1317)	✓	✓	✓	✓	✓	-

Tabulka 751. Platné volby MQOPEN pro každý typ fronty (pokračování)						
Volba	**Alias ("1" na stránce 1317)**	**Lokální a model**	**Vzdálený**	**Nelokální klastr**	**Distribuční seznam**	**Téma**
OOBNDQ ("3" na stránce 1317)	✓	✓	✓	✓	✓	-
OOSAVSKY	✓	✓	-	-	-	-
OOPASI	✓	✓	✓	✓	✓	"5" na stránce 1317
OOPASA	✓	✓	✓	✓	✓	"5" na stránce 1317
OOSETI	✓	✓	✓	✓	✓	"5" na stránce 1317
OOSETA	✓	✓	✓	✓	✓	"5" na stránce 1317
OOALTU@	✓	✓	✓	✓	✓	✓
OOFIQ	✓	✓	✓	✓	✓	✓
OORLQ	✓	✓	✓	✓	-	-

Notes:

1. Platnost voleb pro aliasy závisí na platnosti volby pro frontu, na kterou je určen alias.
2. Tato volba je platná pouze pro lokální definici vzdálené fronty.
3. Tato volba může být uvedena pro jakýkoli typ fronty, ale je ignorována, pokud fronta není fronta klastru.
4. Tento atribut je ignorován pro téma.
5. Tyto atributy lze použít spolu s tématem, ale ovlivní pouze kontext nastavený pro uchovanou zprávu, nikoli pole kontextu odesílaná na libovolného odběratele.

HOTOBJ (10-digit signed integer)-výstup

Popisovač objektu.

Tento manipulátor představuje přístup, který byl vytvořen objektu. Musí být zadán v následných voláních front zpráv, které pracují s objektem. Přestane být platný, když je vydáno volání MQCLOSE, nebo když se jednotka zpracování, která definuje rozsah popisovače, ukončí.

Rozsah manipulátoru je omezen na nejmenší jednotku. paralelní zpracování podporované platformou, na které je aplikace spuštěna; popisovač není platný mimo jednotku paralelního zpracování, ze které bylo vydáno volání MQOPEN:

- V systému IBM ije rozsah manipulátoru úloha, která volá volání.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

Deklarace 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 (vložení zprávy) na IBM i

Volání MQPUT vloží zprávu do fronty, distribučního seznamu nebo do tématu. Fronta, distribuční seznam nebo téma musí být již otevřené.

Syntaxe

MQPUT *(HCONN, HOBJ, MSGDSC, PMO, BUFLEN, BUFFER, CMPCOD, REASON)*

Poznámky k použití

Témata

Pro použití témat se používají následující poznámky:

1. Pokud pomocí příkazu MQPUT publikujete zprávy v tématu, kde jeden nebo více účastníků daného tématu nemohou být předány publikování kvůli problému s frontou odběratele (například je úplný), kód příčiny vrácený do volání MQPUT a chování doručení závisí na nastavení atributů PMSGDLV nebo NPMSGDLV na TOPIC. Všimněte si, že doručení publikování do fronty nedoručených zpráv, je-li uveden RODLQ, nebo zahození zprávy při uvedení RODISC, je považováno za úspěšné doručení zprávy. Pokud není dodána žádná z příruček, vrátí se příkaz MQPUT s hodnotou RC2502. K tomu může dojít v následujících případech:
   - Zpráva se publikuje do TOPIC s PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavené na ALL a každý odběr (trvalý či nikoli) má frontu, která nemůže přijmout publikaci.
   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavenou na ALLDUR a trvalý odběr má frontu, která nemůže přijmout publikování.

   MQPUT se může vrátit s návratovým kódem RCNONE, i když publikování nebylo možné doručit některým odběratelům v následujících případech:
   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na trvání zprávy) nastavené na ALLAVAIL a jakýkoli odběr, trvalý či nikoli, má frontu, která nemůže přijmout publikaci.
   - Zpráva je publikována na TOPIC se PMSGDLV nebo NPMSGDLV (v závislosti na perzistenci zprávy) nastavené na ALLDUR a dočasné předplatné má frontu, která nemůže přijmout publikaci.
2. Pokud nejsou k používanovanému tématu žádné odběratele, publikovaná zpráva se neodešle do žádné fronty a nebude vyřazena. Neprovede žádný rozdíl v tom, zda je tato zpráva trvalá nebo trvalá, nebo zda má neomezené vypršení platnosti nebo nějakou malou dobu vypršení platnosti, je stále vyřazena, pokud nejsou k dispozici žádní odběratelé. Výjimkou je případ, kdy má být zpráva uchována, v takovém případě, ačkoli se neodesílá do front žádné odběratele, je uložena proti tématu, které má být doručeno na všechny nové odběry nebo na odběratele, kteří žádají o zachované publikace pomocí MQSUBRQ.

MQPUT a MQPUT1

Volání MQPUT i volání MQPUT1 lze použít k umístění zpráv do fronty. Volání, které má být použito, závisí na okolnostech.

- Volání MQPUT by mělo být použito v případě, že má být více zpráv umístěno ve stejné *frontě* .

  Bylo zadáno volání MQOPEN s určením volby OOOUT, za nímž následuje jeden nebo více požadavků MQPUT pro přidání zpráv do fronty. Nakonec je fronta uzavřena s voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
- Volání MQPUT1 by mělo být použito pouze v případě, že má být vložena do fronty pouze *jedna* zpráva.

  Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která musí být vydána.

Cílové fronty

Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny následující podmínky. Některé podmínky se vztahují na lokální i vzdálené cílové fronty; ostatní podmínky se vztahují pouze na vzdálené cílové fronty.

Podmínky pro lokální a vzdálené fronty místa určení

- Všechny volání MQPUT se nacházejí ve stejné pracovní jednotce, nebo žádný z nich není v rámci pracovní jednotky.

  Když jsou zprávy vloženy do konkrétní fronty v rámci jedné pracovní jednotky, zprávy z jiných aplikací mohou být promíchány s posloupností zpráv ve frontě.
- Všechna volání MQPUT jsou prováděna pomocí stejného popisovače objektu *H0BJ*.

  V některých prostředích je posloupnost zpráv také zachována při použití různých manipulátorů objektů, za předpokladu, že volání jsou prováděna ze stejné aplikace. Význam "stejné aplikace" je určen prostředím:

  - V systému IBM ije aplikací úloha.
- Všechny zprávy mají stejnou prioritu.

Další podmínky pro vzdálené cílové fronty

- Z odesílajícího správce front je k dispozici pouze jedna cesta ke správci cílové fronty.

  Existuje-li možnost, že některé zprávy v posloupnosti se mohou nacházet v jiné cestě (například kvůli změně konfigurace, vyrovnávání provozu nebo výběru cesty na základě velikosti zprávy), nelze zaručit pořadí zpráv v cílovém správci front.
- Zprávy nejsou dočasně umístěny do front nedoručených zpráv v odesílající, mezilehlé nebo cílové správci front.

  Je-li jedna nebo více zpráv dočasně umístěna do fronty nedoručených zpráv (například z důvodu dočasného zaplnění přenosové fronty nebo cílové fronty), mohou být zprávy doručeny do cílové fronty mimo pořadí.
- Zprávy jsou buď všechny trvalé, nebo všechny dočasné.

  Má-li kanál na trase mezi odesílajícím a cílovým správcem front nastaven atribut **CDNPM** na NPFAST, přechodné zprávy mohou skákat před trvalými zprávami, což vyústuje v pořadí trvalých zpráv vzhledem k neperzistentním zprávám, které se neuchovávají. Avšak pořadí trvalých zpráv ve vztahu k sobě navzájem a přechodných zpráv relativně k sobě navzájem je zachováno.

Pokud tyto podmínky nejsou splněny, mohou být skupiny zpráv použity k zachování pořadí zpráv, ale všimněte si, že toto vyžaduje, aby odesílající i přijímající aplikace používaly podporu seskupování zpráv. Další informace o skupinách zpráv viz:

- Pole *MDMFL* v deskriptoru MQMD
- Volba PMLOGO v MQPMO
- Volba GMLOGO v produktu MQGMO

Distribuční seznamy

Pro použití distribučních seznamů platí následující poznámky.

1. Zprávy lze vložit do rozdělovníku buď pomocí version-1 , nebo version-2 MQPMO. Je-li použita položka MQPMO version-1 MQPMO (nebo version-2 MQPMO s hodnotou *PMREC* rovnou nule), aplikace nebude moci poskytovat žádné záznamy vložení zpráv ani záznamy odpovědí. To znamená, že nebude možné identifikovat fronty, které se setkají s chybami, pokud je zpráva úspěšně odeslána do některých front v rozdělovníku a ne u jiných front.

   Pokud aplikace poskytuje záznamy zpráv nebo záznamy odpovědí, musí být pole *PMVER* nastaveno na PMVER2.

   MQPMO version-2 lze také použít k odeslání zpráv do jediné fronty, která není v rozdělovníku, tím, že zajistíte, že *PMREC* je nula.
2. Kód dokončení a parametry kódu příčiny jsou nastaveny takto:
   - Pokud se vložení do front v rozdělovníku všechny nezdaří nebo selžou stejným způsobem, nastaví se kód dokončení a parametry kódu příčiny, aby popisoval společný výsledek. Záznamy odpovědí MQRR (nejsou-li zadány aplikací) nejsou v tomto případě nastaveny.

     Je-li například každé vložení úspěšné, kód dokončení je nastaven na CCOK a kód příčiny je RCNONE; pokud každý z put selže, protože všechny fronty jsou blokovány pro vložení, parametry jsou nastaveny na CCFAIL a RC2051.
   - Pokud vložení do front v rozdělovníku není úspěšné nebo selže stejným způsobem:
     - Parametr kódu dokončení je nastaven na CCWARN, pokud byl alespoň jeden úspěšně proveden, a do CCFAIL, pokud se všechny nezdařily.
     - Parametr kódu příčiny je nastaven na hodnotu RC2136.

– Záznamy odpovědí (jsou-li poskytovány aplikací) jsou nastaveny na jednotlivé kódy dokončení a kódy příčiny pro fronty v rozdělovníku.

Pokud vložení do cíle selže, protože otevření pro toto místo určení se nezdařilo, pole v záznamu odezvy jsou nastavena na CCFAIL a RC2137; , že místo určení je zahrnuto v *PMIDC*.

3. Pokud se cíl v rozdělovníku interpretuje jako lokální fronta, zpráva se umístí do této fronty v normálním formátu (tj. ne jako zpráva rozdělovníku). Pokud se do stejné lokální fronty vyhodnotí více než jeden cíl, jedna zpráva se umístí do fronty pro každé takové místo určení.

   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 jediná zpráva distribučního seznamu obsahující taková místa určení, i když tyto cíle nesousedí v seznamu míst určení poskytovaného danou aplikací. To však lze provést pouze v případě, že přenosová fronta podporuje zprávy distribučních seznamů (viz atribut fronty **DistLists** popsaný v části “Atributy pro fronty” na stránce 1353).

   Pokud přenosová fronta nepodporuje distribuční seznamy, jedna kopie zprávy v normálním tvaru se umístí do přenosové fronty pro každé místo určení, které používá danou přenosovou frontu.

   Je-li distribuční seznam s daty zprávy aplikace příliš velký pro přenosovou frontu, rozdělí se zpráva distribučního seznamu do menších zpráv rozdělovníku, z nichž každá obsahuje méně míst určení. Pokud se data zprávy aplikací pouze hodí do fronty, zprávy distribučního seznamu nelze vůbec použít a správce front vygeneruje jednu kopii zprávy v normálním formátu pro každý cíl, který používá danou přenosovou frontu.

   Pokud různá místa určení mají jinou prioritu zprávy nebo perzistenci zpráv (může se vyskytnout, když aplikace specifikuje PRQDEF nebo PEQDEF), zprávy nejsou zadrženy ve stejné zprávě distribučního seznamu. Namísto toho správce front generuje tolik zpráv v seznamu distribuce, kolik je třeba k umístění různých hodnot priority a perzistence.

4. Typ vložení do distribučního seznamu může mít za následek:

   - jedna zpráva distribučního seznamu nebo
   - počet menších zpráv v seznamu rozdělení, nebo
   - Směs zpráv distribučního seznamu a normálních zpráv, nebo
   - Pouze normální zprávy.

   To, které z předchozích situací nastane, závisí na tom, zda

   - Místa určení v seznamu jsou lokální, vzdálená, nebo směs.
   - Místa určení mají stejnou prioritu zpráv a perzistenci zpráv.
   - Přenosové fronty mohou obsahovat zprávy distribučního seznamu.
   - Maximální délka zpráv pro přenosové fronty je dostatečně velká, aby pojmula zprávu do formuláře rozdělovníku.

   Avšak bez ohledu na to, která z výše uvedených situací nastane, každá *fyzická* zpráva (tj. každá normální zpráva nebo zpráva distribučního seznamu, která je výsledkem příkazu put) se počítá jako jediná zpráva *jedna* , když:

   - Kontrola, zda aplikace překročila povolený maximální počet zpráv v pracovní jednotce (viz atribut správce front produktu **MaxUncommittedMsgs**).
   - Kontrola, zda jsou podmínky spouštěče splněny.
   - Zvyšuje hloubku fronty a kontroluje, zda by byla překročena maximální hloubka fronty fronty.

5. Jakákoli změna definic front, která by způsobila, že se popisovač stanou neplatnými, byly fronty otevřeny jednotlivě (například změna v cestě vyřešení), nezpůsobí zneplatnění rozdělovníku pro seznam distribucí. Výsledkem je však selhání této konkrétní fronty, je-li popisovač distribučního seznamu použit v následném volání MQPUT.

Záhlaví

Je-li zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátku dat zprávy aplikace, provede správce front určité kontroly struktury záhlaví, aby ověřil, zda jsou platné. Pokud správce front zjistí chybu, volání selže s příslušným kódem příčiny. Provedená kontrola se liší v závislosti na konkrétních strukturách, které jsou přítomné. Kromě toho jsou kontroly provedeny pouze tehdy, je-li pro volání MQPUT nebo MQPUT1 použit version-2 nebo novější MQMD; kontroly se neprovedou, je-li použit version-1 MQMD, i když je MQMDE přítomen na začátku dat zprávy aplikace.

Následující struktury záhlaví IBM MQ jsou ověřovány zcela správcem front: MQDH, MQMDE.

Pro jiné struktury záhlaví IBM MQ provádí správce front určité ověření, ale nekontroluje každé pole. Struktury, které nejsou podporovány lokálním správcem front a strukturám následujících po prvním MQDLH ve zprávě, nejsou ověřeny.

Kromě obecných kontrol na polích 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 v rámci zprávy PCF se musí rovnat délce určené parametrem **BUFLEN** na volání MQPUT nebo MQPUT1 . Zpráva PCF je zpráva, která má jeden z následujících názvů formátů:
  - FMADMN
  - FMEVNT
  - FMPCF
- Struktury IBM MQ nesmí být zkráceny, s výjimkou následujících situací, kdy jsou povoleny oříznuté struktury:
  - Zprávy, které jsou zprávami sestavy.
  - Zprávy příkazu PCF.
  - Zprávy obsahující strukturu MQDLH. (Struktury *následující* první MQDLH mohou být oříznuty; struktury předcházející MQDLH nemohou.)

Vyrovnávací paměť

Parametr **BUFFER** uvedený v příkladu programování v RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li požadována větší vyrovnávací paměť, měl by být deklarovaný parametr deklarovaný jako struktura nebo jako pole ve fyzickém souboru. To zvýší maximální možnou délku přibližně na 32 kB.

Parametry

Volání MQPUT má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOTBJ (10ciferné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento popisovač představuje frontu, do níž je zpráva přidána, nebo téma, do kterého je zpráva publikována. Hodnota *H0BJ* byla vrácena předchozím voláním MQOPEN, které určuje volbu OOOUT.

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zpráv.

Tato struktura popisuje atributy odesílané zprávy a přijímá informace o zprávě po dokončení požadavku na vložení. Podrobnosti viz “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099.

Pokud aplikace poskytuje version-1 MQMD, mohou být data zprávy uvozen strukturou MQMDE, aby bylo možné zadat hodnoty pro pole, která existují v version-2 MQMD, ale ne v version-1. Pole *MDFMT* v deskriptoru MQMD musí být nastaveno na hodnotu FMMDE, aby bylo zřejmé, že je přítomen prvek MQMDE. Další informace viz část “MQMDE (rozšíření deskriptoru zpráv) na IBM i” na stránce 1141.

PMO (MQPMO)-vstup/výstup

Volby, které řídí akci MQPUT.

Podrobnosti viz “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161.

BUFLEN (10ciferné číslicové celé číslo)-vstup

Délka zprávy v produktu *BUFFER*.

Nula je platná a označuje, že zpráva neobsahuje žádná data aplikace. Horní mez pro *BUFLEN* závisí na různých faktorech:

- Je-li cílová fronta sdílenou frontou, horní limit je 63 kB (64 512 bajtů).
- Je-li cílem lokální fronta nebo je tento cíl přeložen do lokální fronty (ale není sdílenou frontou), závisí horní limit na tom, zda:
  - Lokální správce front podporuje segmentaci.
  - Odesílající aplikace uvádí příznak, který umožňuje správci front segmentovat zprávu. Tento parametr je MFSEGA a lze jej zadat buď v version-2 MQMD, nebo v MQMDE používaném s MQMD version-1 .

  Pokud jsou obě tyto podmínky splněny, *BUFLEN* nemůže překročit 999 999 999 minus hodnotu pole *MDOFF* v MQMD. Nejdelší logická zpráva, která může být vložena, je proto 999 999 999 bajtů (když *MDOFF* je nula). Omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, však může vést k nižšímu limitu.

  Pokud jedna nebo obě dříve popsané podmínky nejsou splněny, *BUFLEN* nemůže překročit menší hodnotu atributu **MaxMsgLength** fronty a atributu **MaxMsgLength** správce front.
- Je-li místem určení vzdálená fronta nebo je tento cíl převeden na vzdálenou frontu, platí podmínky pro lokální fronty, *ale v každém správci front, jehož prostřednictvím musí zpráva projít, aby dosáhla cílové fronty* ; zejména:
  1. Lokální přenosová fronta používaná k dočasnému ukládání zprávy v lokálním správci front
  2. Intermediační přenosové fronty (jsou-li nějaké) používané k ukládání zpráv ve správcích front na trase mezi lokálními a cílovými správci front.
  3. Cílová fronta v cílovém správci front

  Nejdelší zpráva, kterou lze vložit, se proto řídí nejrestriktivnějšími zprávami z těchto front a správců front.

  Je-li zpráva v přenosové frontě, jsou spolu s daty zprávy uloženy další informace a snižuje množství dat aplikace, které lze provést. V této situaci se doporučuje odečíst bajty LNMHD z hodnot *MaxMsgLength* přenosových front při určování limitu pro *BUFLEN*.

  Poznámka: Pouze selhání při dodržení podmínky 1 může být diagnostikováno synchronně (s kódem příčiny RC2030 nebo RC2031), když je zpráva vložena. Nejsou-li podmínky 2 nebo 3 splněny, je zpráva přesměrována do fronty nedoručených zpráv (nedoručené zprávy) buď v intermediačních správci front, nebo v cílovém správci front. Pokud k tomu dojde, vygeneruje se zpráva sestavy, pokud ji odesílatel požadoval.

BUFFER (1-bytový bitový řetězec x BUFLEN)-vstup

Data zprávy.

Jedná se o vyrovnávací paměť obsahující data aplikace, která se mají odeslat. 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 striktnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Pokud *BUFFER* obsahuje znaková data, číselná data nebo oboje, pole *MDCSI* a *MDENC* v parametru **MSGDSC** by měla být nastavena na hodnoty odpovídající datům; tím se umožní příjemci zprávy převést data (je-li to nutné) na znakovou sadu a kódování použité příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT musí být ve znakové sadě poskytnuté atributem správce front produktu **CodedCharSetId** a kódování lokálního správce front zadaného ENNAT.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2104
(2104, X'838 ') Volba sestavy v deskriptoru zprávy nebyla rozpoznána.

RC2136
(2136, X'858 ') Vráceno více kódů příčiny.

Je-li *CMPCOD* 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') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

RC2026
(2026, X'7EA') Deskriptor zprávy není platný.

RC2027
(2027, X'7EB') Chybí odpověď na frontu.

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 je větší než maximum pro správce front.

RC2039
(2039, X'7F7') Fronta není otevřena pro výstup.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

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 s blokováno pro frontu.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2056
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2061
(2061, X'80D') Volby sestav v deskriptoru zpráv nejsou platné.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2072
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

RC2093
(2093, X'82D') Fronta není otevřena pro předání všech kontextů.

RC2094
(2094, X'82E') Fronta není otevřena pro kontext předání identity.

RC2095
(2095, X'82F') Fronta není otevřena pro nastavení všech kontextů.

RC2096
(2096, X'830 ') Fronta není otevřena pro nastavení kontextu identity.

RC2097
(2097, X'831 ') Manipulátor fronty odkazovaný tak, aby neukládaný kontext.

RC2098
(2098, X'832 ') Kontext není k dispozici pro uvedený popisovač fronty.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2135
(2135, X'857 ') Struktura hlavičky distribuce není platná.

RC2136
(2136, X'858 ') Vráceno více kódů příčiny.

RC2137
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

RC2149
(2149, X'865 ') struktur PCF nejsou platné.

RC2154
(2154, X'86A') Počet záznamů přítomných záznamů není platný.

RC2156
(2156, X'86C') Záznamy odpovědí nejsou platné.

RC2158
(2158, X'86E') Příznaky vložení záznamu zprávy nejsou platné.

RC2159
(2159, X'86F') Položit záznamy zpráv nejsou platné.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2173
(2173, X'87D') Struktura volby vložení zprávy není platná.

RC2185
(2185, X'889 ') Nekonzistentní specifikace perzistence.

RC2188
(2188, X'88C') Volání bylo zamítnuto uživatelskou procedurou pracovní zátěže klastru.

RC2189
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

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 úplná.

RC2242
(2242, X'8C2') Logická zpráva není úplná.

RC2245
(2245, X'8C5') Nekonzistentní specifikace jednotky práce.

RC2248
(2248, X'8C8') Rozšíření deskriptoru zpráv není platné.

RC2249
(2249, X'8C9') Příznaky zprávy nejsou platné.

RC2250
(2250, X'8CA') Pořadové číslo zprávy není platné.

RC2251
(2251, X'8CB') Odsazení segmentu zprávy není platné.

RC2252
(2252, X'8CC') Původní délka není platná.

RC2253
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

RC2255
(2255, X'8CF') Unit of work not available for the queue manager to use.

RC2257
(2257, X'8D1') Chybná verze dodaných MQMD.

RC2258
(2258, X'8D2') Identifikátor skupiny není platný.

RC2266
(2266, X'8DA') Ukončení pracovní zátěže klastru se nezdařilo.

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 strukturu MQEPH, která není platná.

RC2479
(2479, X'9AF') Publikování nebylo možné zachovat.

RC2480
(2480, X'9B0') Cílový typ se změnil: fronta aliasů odkazuje na frontu, ale nyní odkazuje na téma.

RC2502
(2502, X'9C6') Publikování se nezdařilo a publikování nebylo doručeno žádnému odběratelům.

RC2551
(2551, X'9F7') Určený řetězec výběru není k dispozici.

RC2554
(2554, X'9FA') Obsah zprávy nemohl být analyzován za účelem určení, zda má být zpráva doručena odběrateli s rozšířeným voličem zpráv.

Deklarace 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 (Vložení jedné zprávy) v systému 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. Volání MQPUT i volání MQPUT1 lze použít k umístění zpráv do fronty. Volání, které má být použito, závisí na okolnostech:
   - Volání MQPUT by mělo být použito v případě, že má být více zpráv umístěno ve stejné *frontě* .

     Bylo zadáno volání MQOPEN s určením volby OOOUT, za nímž následuje jeden nebo více požadavků MQPUT pro přidání zpráv do fronty. Nakonec je fronta uzavřena s voláním MQCLOSE. To poskytuje lepší výkon než opakované použití volání MQPUT1 .
   - Volání MQPUT1 by mělo být použito pouze v případě, že má být vložena do fronty pouze *jedna* zpráva.

     Toto volání zapouzdřuje volání MQOPEN, MQPUT a MQCLOSE do jednoho volání a minimalizuje počet volání, která musí být vydána.
2. Pokud aplikace vkládá posloupnost zpráv do stejné fronty bez použití skupin zpráv, pořadí těchto zpráv je zachováno, jsou-li splněny určité podmínky. Avšak ve většině prostředí volání MQPUT1 nesplňuje tyto podmínky, a proto nezachovává pořadí zpráv. Místo toho v těchto prostředích musí být místo volání MQPUT použito volání MQPUT. Podrobnosti naleznete v poznámkách k použití v popisu volání MQPUT.
3. Volání MQPUT1 lze použít k umístění zpráv do distribučních seznamů. Obecné informace o tomto tématu najdete v poznámkách k použití pro volání MQOPEN a MQPUT.

   Při použití volání MQPUT1 se používají následující rozdíly:

   a. Jsou-li záznamy odpovědí MQRR poskytnuty aplikací, musí být poskytnuty pomocí struktury MQOD; nelze je poskytnout pomocí struktury MQPMO.

   b. Kód příčiny RC2137 není nikdy vrácen položkou MQPUT1 v záznamech odezvy; pokud se nepodaří otevřít frontu, obsahuje záznam odpovědi pro tuto frontu skutečný kód příčiny, který je výsledkem operace otevření.

      Pokud operace otevření fronty uspěje s kódem dokončení CCWARN, kód dokončení a kód příčiny v záznamu odpovědi pro danou frontu se nahradí kódem dokončení a kódem příčiny, které jsou výsledkem operace put.

      Stejně jako v případě volání MQOPEN a MQPUT správce front nastaví záznamy odpovědí (je-li k dispozici) pouze v případě, že výsledek volání není stejný pro všechny fronty v rozdělovníku; to je indikováno dokončením volání s kódem příčiny RC2136.
4. Je-li volání MQPUT1 použito k vložení zprávy do fronty klastru, volání se bude chovat, jako by byl v rámci volání MQOPEN zadán parametr OOBNDN.

5. Je-li zpráva vložena s jednou nebo více strukturami záhlaví IBM MQ na začátku dat zprávy aplikace, provede správce front určité kontroly struktury záhlaví, aby ověřil, zda jsou platné. Další informace o tomto tématu naleznete v poznámkách k použití volání MQPUT.
6. Pokud se objeví více než jedna z varovných situací (viz parametr **CMPCOD**), vrácený kód příčiny je *první* v následujícím seznamu, který se používá:
   a. RC2136
   b. RC2242
   c. RC2241
   d. RC2049 nebo RC2104
7. Parametr **BUFFER** uvedený v příkladu programování v RPG je deklarován jako řetězec; to omezuje maximální délku parametru na 256 bajtů. Je-li požadována větší vyrovnávací paměť, měl by být deklarovaný parametr deklarovaný jako struktura nebo jako pole ve fyzickém souboru. To zvýší maximální možnou délku přibližně na 32 kB.

Parametry

Volání MQPUT1 má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

OBJDSC (MQOD)-vstup/výstup

Deskriptor objektu.

Jedná se o strukturu, která identifikuje frontu, do níž je zpráva přidána. Podrobnosti viz “MQOD (Object descriptor) na systému IBM i” na stránce 1147.

Uživatel musí být autorizovaný, aby mohl otevřít frontu pro výstup. Fronta **nesmí** být modelovou frontou.

MSGDSC (MQMD)-vstupní/výstupní

Deskriptor zpráv.

Tato struktura popisuje atributy odesílané zprávy a přijímá informace zpětné vazby po dokončení požadavku na vložení. Podrobnosti viz “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099.

Pokud aplikace poskytuje version-1 MQMD, mohou být data zprávy uvozen strukturou MQMDE, aby bylo možné zadat hodnoty pro pole, která existují v version-2 MQMD, ale ne v version-1. Pole *MDFMT* v deskriptoru MQMD musí být nastaveno na hodnotu FMMDE, aby bylo zřejmé, že je přítomen prvek MQMDE. Další informace viz část “MQMDE (rozšíření deskriptoru zpráv) na IBM i” na stránce 1141.

PMO (MQPMO)-vstup/výstup

Volby, které řídí akci MQPUT1.

Podrobnosti viz “MQPMO (volby vkládání zpráv) v systému IBM i” na stránce 1161.

BUFLEN (10ciferné číslicové celé číslo)-vstup

Délka zprávy v produktu *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 (1-bytový bitový řetězec x BUFLEN)-vstup

Data zprávy.

Jedná se o vyrovnávací paměť obsahující data zprávy aplikace, která se mají odeslat. 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 IBM MQ struktur záhlaví), ale některé zprávy mohou vyžadovat striktnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Pokud *BUFFER* obsahuje znaková data, číselná data nebo oboje, pole *MDCSI* a *MDENC* v parametru **MSGDSC** by měla být nastavena na hodnoty odpovídající datům; tím se umožní příjemci zprávy převést data (je-li to nutné) na znakovou sadu a kódování použité příjemcem.

Poznámka: Všechny ostatní parametry volání MQPUT1 musí být ve znakové sadě poskytnuté atributem správce front **CodedCharSetId** a kódováním lokálního správce front poskytnutého produktem ENNAT.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCWARN:

RC2104
(2104, X'838 ') Volba sestavy v deskriptoru zprávy nebyla rozpoznána.

RC2136
(2136, X'858 ') Vráceno více kódů příčiny.

RC2049
(2049, X'801 ') Priorita zprávy překračuje maximální podporovanou hodnotu.

RC2241
(2241, X'8C1') Skupina zpráv není úplná.

RC2242
(2242, X'8C2') Logická zpráva není úplná.

Je-li *CMPCOD* CCFAIL:

RC2001
(2001, X'7D1') Alias základní fronty 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') Žádné další zprávy nelze v rámci aktuální jednotky práce zpracovat.

RC2026
(2026, X'7EA') Deskriptor zprávy není platný.

RC2027
(2027, X'7EB') Chybí odpověď na frontu.

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 je větší než maximum pro správce front.

RC2035
(2035, X'7F3') Chybí autorizace 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 s blokováno pro frontu.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2053
(2053, X'805 ') Fronta již obsahuje maximální počet zpráv.

RC2056
(2056, X'808 ') Na disku pro frontu není k dispozici žádné místo.

RC2057
(2057, X'809 ') Typ fronty není platný.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2061
(2061, X'80D') Volby sestav v deskriptoru zpráv nejsou platné.

RC2063
(2063, X'80F') Došlo k chybě zabezpečení.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2072
(2072, X'818 ') Podpora synchronizačních bodů není k dispozici.

RC2082
(2082, X'822 ') Neznámá alias základní fronty.

RC2085
(2085, X'825 ') Neznámý název objektu.

RC2086
(2086, X'826 ') Neznámý správce front objektu.

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 ') Manipulátor fronty odkazovaný tak, aby neukládaný kontext.

RC2098
(2098, X'832 ') Kontext není k dispozici pro uvedený popisovač fronty.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2135
(2135, X'857 ') Struktura hlavičky distribuce není platná.

RC2136
(2136, X'858 ') Vráceno více kódů příčiny.

RC2149
(2149, X'865 ') struktur PCF nejsou platné.

RC2154
(2154, X'86A') Počet záznamů 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 vložení záznamu zprávy nejsou platné.

RC2159
(2159, X'86F') Položit záznamy zpráv nejsou platné.

RC2161
(2161, X'871 ') Správce front je uváděn do klidového stavu.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2173
(2173, X'87D') Struktura volby vložení zprávy není platná.

RC2184
(2184, X'888 ') Název vzdálené fronty není platný.

RC2188
(2188, X'88C') Volání bylo zamítnuto uživatelskou procedurou pracovní zátěže klastru.

RC2189
(2189, X'88D') Rozpoznání názvu klastru se nezdařilo.

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 ') Chyba použití předvolené přenosové fronty.

RC2258
(2258, X'8D2') Identifikátor skupiny není platný.

RC2248
(2248, X'8C8') Rozšíření deskriptoru zpráv 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') Odsazení segmentu zprávy není platné.

RC2252
(2252, X'8CC') Původní délka není platná.

RC2253
(2253, X'8CD') Délka dat v segmentu zprávy je nula.

RC2255
(2255, X'8CF') Unit of work not available for the queue manager to use.

RC2257
(2257, X'8D1') Chybná verze dodaných MQMD.

RC2266
(2266, X'8DA') Ukončení pracovní zátěže klastru se nezdařilo.

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 strukturu MQEPH, která není platná.

RC2551
(2551, X'9F7') Určený řetězec výběru není k dispozici.

RC2554
(2554, X'9FA') Obsah zprávy nemohl být analyzován za účelem určení, zda má být zpráva doručena odběrateli s rozšířeným voličem zpráv.

Deklarace 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 (Nastavit atributy objektu) v systému IBM i

Volání MQSET se používá ke změně atributů objektu reprezentovaného popisovačem. Objekt musí být fronta.

Syntaxe

MQSET *(HCONN, HOBJ, SELCNT, SELS, IACNT, INTATR, CALEN, CHRATR, CMPCOD, REASON)*

Poznámky k použití

1. Při použití tohoto volání může aplikace určit pole celočíselných atributů nebo kolekci řetězců znakových atributů, nebo obojí. Pokud se nevyskytnou žádné chyby, uvedené atributy jsou všechny nastaveny současně. Pokud dojde k chybě (například, pokud je selektor neplatný nebo je proveden pokus o nastavení atributu na hodnotu, která není platná), volání selže a nejsou nastaveny žádné atributy.
2. Hodnoty atributů lze určit pomocí volání MQINQ; Podrobnosti viz "MQINQ (Dotaz na atributy objektů) v systému IBM i" na stránce 1291 .

   Poznámka: Ne všechny atributy s hodnotami, které mohou být dotazovány při použití volání MQINQ, mohou mít své hodnoty změněny pomocí volání MQSET. Pomocí tohoto volání lze například nastavit žádné atributy objektu procesu nebo správce front.
3. Změny atributů jsou zachovány po restartu správce front (jiné než změny dočasných dynamických front, které nepřečkají restarty správce front).
4. Atributy modelové fronty nelze změnit pomocí volání MQSET. Pokud však otevřete modelovou frontu pomocí volání MQOPEN s volbou MQOO_SET, můžete použít volání MQSET k nastavení atributů dynamické lokální fronty vytvořené voláním MQOPEN.

5. Je-li nastavovaný objekt fronta klastru, musí existovat lokální instance fronty klastru, aby byla otevřená úspěšná.

Další informace o atributech objektů najdete v tématech:

- “Atributy pro fronty” na stránce 1353
- “Atributy pro seznamy názvů” na stránce 1381
- “Atributy pro definice procesu v systému IBM i” na stránce 1382
- “Atributy pro správce front v systému IBM i” na stránce 1384

Parametry

Volání MQSET má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota HCONN byla vrácena předchozím voláním MQCONN nebo MQCONNX.

HOTBJ (10ciferné celé číslo se znaménkem)-vstup

Popisovač objektu.

Tento manipulátor představuje objekt fronty s atributy, které mají být nastaveny. Popisovač byl vrácen předchozím voláním MQOPEN, které určuje volbu OOSET.

SELCNT (10ciferné 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.

EL (desetimístné podepsané celé číslo x SELCNT)-vstup

Pole selektorů atributů.

Jedná se o pole selektorů atributů produktu **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. Jsou povoleny pouze určité hodnoty IA* a CA*; tyto hodnoty jsou vypsány později v této sekci.

Selektory mohou být zadány v libovolném pořadí. Hodnoty atributů odpovídající celočíselným selektorům atributů (selektory IA*) musí být určeny v produktu INTATR ve stejném pořadí, v jakém se tyto selektory vyskytují v produktu SELS. Hodnoty atributu, které odpovídají selektorům atributu znaku (selektory CA*), musí být určeny v produktu CHRATR ve stejném pořadí, ve kterém se tyto selektory vyskytují. Selektory IA* mohou být prokládané selektory CA*; důležitá je pouze relativní pořadí v rámci každého typu.

Není chybou uvádět stejný selektor více než jednou; pokud se tak stalo, poslední hodnota uvedená pro konkrétní selektor je ta, která se projeví.

Poznámka:

1. Selektory atributů celého čísla a znaku jsou přiděleny ve dvou různých rozsazích; selektory IA* jsou umístěny v rozsahu IAFRST až IALAST a selektory CA* v rozsahu CAFRST přes CALAST.

   Pro každý rozsah, konstanty IALSTU a CALSTU definují nejvyšší hodnotu, kterou správce front přijme.
2. Pokud se všechny selektory IA* vyskytnou jako první, mohou být použita stejná čísla prvků pro adresování příslušných prvků v polích SELS a INTATR .

Atributy, které lze nastavit, jsou vypsány v následující tabulce. Pomocí tohoto volání nelze nastavit žádné další atributy. Pro selektory atributu CA* konstanta, která definuje délku řetězce, která je požadována v CHRATR , je uvedena v závorkách.

Tabulka 752. Selektory atributů MQSET pro fronty

Selektor	Popis	Poznámka
KATRGD	Data spouštěče (LNTRGD).	“2” na stránce 1336
IADRIST	Podpora distribučního seznamu.	“1” na stránce 1336
IAIGET	Zda jsou povoleny operace get.	
IAIPUT	Zda jsou povoleny operace vložení.	
IATRAGC	Řízení spouštěče.	“2” na stránce 1336
IATRAGD	Hloubka spouątěče.	“2” na stránce 1336
IATRGP	Priorita zprávy prahové hodnoty pro spouątěče.	“2” na stránce 1336
SPRÁVA ITRGT	Typ spouątěče.	“2” na stránce 1336

Notes:

1. Podporováno pouze na následujících platformách:
   - AIX
   - IBM i
   - Windows

   a pro klienty IBM MQ připojené k těmto systémům.
2. Nepodporováno na VSE/ESA.

IACNT (10ciferné celé číslo se znaménkem)-vstup

Počet celočíselných atributů.

Jedná se o počet prvků v poli INTATR a musí být alespoň počtem selektorů IA* v parametru **SELS** . Nula je platná hodnota, pokud neexistují žádné.

INTATR (10místný podepsaný intrg x rxIACNT)-vstup

Pole celočíselných atributů.

Toto je pole celočíselných hodnot atributů IACNT . Tyto hodnoty atributů musí být ve stejném pořadí jako selektory IA* v poli SELS .

CALEN (10ciferné celé číslo se znaménkem)-vstup

Délka vyrovnávací paměti atributů znaků.

Toto je délka v bajtech parametru **CHRATR** a musí být alespoň součtem délek znakových atributů uvedených v poli SELS . Nula je platná hodnota, pokud v SELSnejsou žádné selektory CA*.

CHRATR (1-bytový znakový řetězec x CALEN)-vstup

Atributy znaků.

Jedná se o vyrovnávací paměť obsahující hodnoty atributu znaku zřetězené 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 nemezerových znaků, než je definovaná délka atributu, hodnota v CHRATR musí být směrem doprava vyplněna mezerami, aby se hodnota atributu shodovala s definovanou délkou atributu.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud má parametr CMPCOD hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li CMPCOD 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 atributů znaků 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 inhibit-get nebo inhibit-put queue není platná.

RC2021
(2021, X'7E5') Počet celočíselných atributů není platný.

RC2023
(2023, X'7E7') Pole celočíselné atributy není platné.

RC2040
(2040, X'7F8') Fronta není otevřena pro nastavení.

RC2041
(2041, X'7F9') Definice objektu byla od otevření změněna.

RC2101
(2101, X'835 ') Objekt je poškozen.

RC2052
(2052, X'804 ') Fronta byla odstraněna.

RC2058
(2058, X'80A') Název správce front není platný nebo je neznámý.

RC2059
(2059, X'80B') Správce front není k dispozici pro připojení.

RC2162
(2162, X'872 ') Správce front se vypíná.

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2065
(2065, X'811 ') Počet selektorů není platný.

RC2067
(2067, X'813 ') Selektor atributu není platný.

RC2066
(2066, X'812 ') Počet selektorů je příliš velký.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

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.

Deklarace 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 zpracování zpráv) v systému IBM i

Volání MQSETMP nastavuje nebo upravuje vlastnost obslužné rutiny zprávy.

Syntaxe

MQSETMP *(Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength, Value, CompCode, Reason)*

Poznámky k použití

- Toto volání můžete použít pouze v případě, že správce front sám koordinuje pracovní jednotku. To může být:
  - Lokální jednotka práce, kde se změny týkají pouze IBM MQ prostředků.
  - Globální jednotka práce, kde mohou změny ovlivnit prostředky patřící jiným správcům prostředků a které ovlivňují prostředky produktu IBM MQ .

  Další podrobnosti o lokálních a globálních jednotkách práce viz "MQBEGIN (Begin unit of work) na IBM i" na stránce 1240.
- V prostředích, kde správce front nekoordinuje jednotku práce, použijte místo MQBACK odpovídající zpětné volání. Prostředí může také podporovat implicitní vrácení zpět v důsledku abnormálního ukončení aplikace.
  - V systému z/OSpoužijte následující volání:
    - Dávkové programy (včetně dávkových DL/I programů produktu IMS) mohou použít volání MQBACK, pokud má jednotka práce vliv pouze na prostředky produktu IBM MQ . However, if the unit of work affects both IBM MQ resources and resources belonging to other resource managers (for example, Db2), use the SRRBACK call provided by the z/OS Recoverable Resource Service (RRS). Volání SRRBACK vrací změny prostředků náležejících ke správcům prostředků, kteří byli povoleni pro koordinaci RRS.
    - Aplikace produktu CICS musí použít příkaz `EXEC CICS SYNCPOINT ROLLBACK` k zálohování jednotky práce. Nepoužívejte volání MQBACK pro aplikace produktu CICS .
    - Aplikace produktu IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. produkt `ROLB` , aby odvrátila jednotku práce. Nepoužívejte volání MQBACK pro aplikace IMS (jiné než dávkové DL/I programy).
  - V systému IBM ipoužijte toto volání pro lokální jednotky práce koordinované správcem front. To znamená, že definice vázaného zpracování nesmí existovat na úrovni úlohy, to znamená, že příkaz STRCMTCTL s parametrem **`CMTSCOPE(*JOB)`** nesmí být vydán pro úlohu.
- Pokud aplikace skončí s nepotvrzenými změnami v pracovní jednotce, závisí odebrání těchto změn na tom, zda aplikace skončí normálně nebo abnormálně. Další podrobnosti naleznete v poznámkách k použití v příručce "MQDISC (Odpojení správce front) v systému IBM i" na stránce 1276 .

- Když aplikace vkládá nebo získává zprávy ve skupinách nebo segmentech logických zpráv, uchovává správce front informace vztahující se ke skupině zpráv a logické zprávě pro poslední úspěšné volání MQPUT a MQGET. Tyto informace jsou asociovány s manipulátorem fronty a zahrnují takové položky jako:
  - Hodnoty polí *GroupId*, *MsgSeqNumber*, *Offset*a *MsgFlags* v MQMD.
  - Zda je zpráva součástí jednotky práce.
  - Pro volání MQPUT: zda je zpráva trvalá nebo přechodná.

  Správce front uchovává tři skupiny informací o skupinách a segmentech, jednu sadu pro každou z následujících možností:
  - Poslední úspěšné volání MQPUT (může být součástí jednotky práce).
  - Poslední úspěšné volání MQGET, které odebrala zprávu z fronty (může být součástí jednotky práce).
  - Poslední úspěšné volání MQGET, které procházelo zprávu ve frontě (to nemůže být součástí pracovní jednotky).

  Pokud aplikace vkládá nebo získává zprávy jako součást pracovní jednotky, a pak se aplikace rozhodne zálohovat jednotku práce, informace o skupině a segmentu se obnoví na hodnotu, kterou předtím měla:
  - Informace přidružené k volání MQPUT se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQPUT pro tento popisovač fronty v aktuální transakci.
  - Informace přidružené k volání MQGET se obnoví na hodnotu, kterou měla před prvním úspěšným voláním MQGET pro daný popisovač fronty v aktuální pracovní jednotce.

  Fronty, které byly aktualizovány aplikací po spuštění jednotky práce, ale mimo rozsah jednotky práce, nemají obnovenou skupinovou a segmentovou informaci, pokud je jednotka práce zálohována.

  Obnova informace o skupině a segmentu na její předchozí hodnotu, když je zálohována jednotka práce, umožňuje aplikaci šířit velkou skupinu zpráv nebo velkou logickou zprávu skládající se z mnoha segmentů přes několik jednotek práce a restartovat ve správném bodu ve skupině zpráv nebo v logické zprávě, pokud se jedna z jednotek práce nezdaří.

  Použití několika jednotek práce může být výhodné v případě, že lokální správce front má pouze omezené množství paměti fronty. Aplikace však musí udržovat dostatečné informace, aby bylo možné restartovat vkládání nebo získání zpráv ve správném okamžiku, pokud dojde k selhání systému.

  Podrobnosti o restartování ve správném bodu po selhání systému najdete v tématu PMLOGO popsané v části PMOPT (10 číslic se znaménkem celého čísla)a s volbou GMLOGO popsané v souboru GMOPT (celé číslo se znaménkem 10).

  Ostatní poznámky k použití se použijí pouze tehdy, když správce front koordinuje jednotky práce:
- Jednotka práce má stejný rozsah jako manipulátor připojení. Všechna volání IBM MQ , která ovlivňují konkrétní jednotku práce, musí být prováděna pomocí stejného manipulátoru připojení. Volání vydaná pomocí jiného popisovače připojení (například volání vydaná jinou aplikací) ovlivňují jinou jednotku práce. Informace o rozsahu manipulátorů připojení viz HCONN (10ciferné hexadecimální číslo)-výstup .
- Pouze zprávy, které byly vloženy nebo načteny jako součást aktuální jednotky práce, jsou tímto voláním ovlivněny.
- Dlouhá-spuštěná aplikace, která vydává volání MQGET, MQPUT nebo MQPUT1 v rámci pracovní jednotky, ale která nikdy nevydává výzvu k potvrzení nebo vrácení, může vyplnit fronty zprávami, které nejsou k dispozici pro jiné aplikace. Pro ochranu proti této možnosti musí administrátor nastavit atribut správce front produktu **MaxUncommittedMsgs** na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale dostatečně vysoko, aby umožnily správné fungování očekávaných aplikací systému zpráv.

Parametry

Volání MQSETMP má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front.

Hodnota se musí shodovat s manipulátorem připojení, který byl použit k vytvoření manipulátoru zprávy zadaného argumentem **HMSG** .

Pokud byl popisovač zprávy vytvořen pomocí HCUNAS, musí být ustanoveno platné připojení na podprocesu nastaveném na vlastnost popisovače zprávy, jinak se volání nezdaří s kódem příčiny RC2009 .

HMSG (20-digit signed integer)-vstup

Jedná se o popisovač zprávy, který má být upraven. Hodnota byla vrácena předchozím voláním MQCRTMH.

SETOPT (MQSMPO)-vstup

Řídí, jak jsou nastaveny vlastnosti zpráv.

Tato struktura umožňuje aplikacím určit volby, které řídí způsob nastavení vlastností zpráv. Struktura je vstupním parametrem volání MQSETMP. Další informace viz MQSMPO .

PRNAME (MQCHARV)-vstup

Jedná se o název vlastnosti, která má být nastavena.

Další informace o použití názvů vlastností naleznete v tématech Názvy vlastností a Omezení názvů vlastností .

PRPDSC (MQPD)-vstup/výstup

Tato struktura se používá k definování atributů vlastnosti, včetně:

- co se stane, pokud vlastnost není podporována
- jaký kontext zprávy vlastnost patří
- Jaké zprávy je vlastnost kopírována do průběhu toku

Další informace o této struktuře viz MQPD .

TYPE (10ciferné celé číslo se znaménkem)-vstup

Datový typ nastavované vlastnosti. Může se jednat o jednu z následujících možností:

TYBOL
Booleovský. *ValueLength* musí být 4.

TYPBST
Řetězec bajtů. Hodnota *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
Celé 32bitové celé číslo se znaménkem. *ValueLength* musí být 4.

TYPI64
Celé 64bitové celé číslo se znaménkem. *ValueLength* musí být 8.

TYPF32
32bitové číslo s pohyblivou řádovou čárkou. *ValueLength* musí být 4.

TYPF64
Číslo 64bitové pohyblivé řádové čárky. *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 (10ciferné celé číslo se znaménkem)-vstup

Délka hodnoty vlastnosti v parametru *hodnota* v bajtech.

Nula je platná pouze pro hodnoty null, nebo pro řetězce nebo bajtové řetězce. Nula označuje, že vlastnost existuje, ale že tato hodnota neobsahuje žádné znaky ani bajty.

Hodnota musí být větší než nula nebo rovna nule nebo následující speciální hodnota, pokud má parametr *Type* nastavovací hodnotu TYPSTR:

HODNOTA VLNULL
Hodnota je oddělena první hodnotou null zjištěnou v řetězci. Hodnota null není zahrnuta jako součást řetězce. Tato hodnota je neplatná, pokud parametr TYPSTR také není nastaven.

Poznámka: Znak Null použitý k ukončení řetězce, pokud je hodnota VLNULL nastavena na hodnotu null ze znakové sady hodnoty.

VALUE (1-bytový 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 tento parametr deklarován jako ukazatel-to-void; adresa libovolného typu dat může být zadána jako parametr.

Je-li *ValueLength* nula, *Hodnota* není odkazována. V tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 asembleru null.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Má-li parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Má-li parametr *CMPCOD* hodnotu CCWARN:

RC2421
(2421, X'0975 ') Složku MQRFH2 obsahující vlastnosti nebylo možné analyzovat.

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 domovské ASID se liší.

RC2004
(2004, X'07D4') Hodnota parametru hodnoty není platná.

RC2005
(2005, X'07D5') Hodnota parametru délky hodnoty není platná.

RC2219
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2460
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

RC2499
(2499, X'09C3') Popisovač zprávy je již používán.

RC2046
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

RC2482
(2482, X'09B2') Struktura deskriptoru vlastností není platná.

RC2442
(2442, X'098A') Neplatný název vlastnosti.

RC2473
(2473, X'09A9') Neplatný typ dat vlastnosti.

RC2472
(2472, X'09A8') Chyba formátu čísla zjištěna v datech hodnoty.

RC2463
(2463, X'099F') Nastavení struktury voleb vlastností zprávy není platné.

RC2111
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

Další informace viz část "Návratové kódy pro IBM i (ILE RPG)" na stránce 1411.

Deklarace 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) v systému IBM i

Použijte volání MQSTAT k získání informací 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 vrací informace o předchozích asynchronních operacích MQPUT a operací MQPUT1 . Struktura MQSTAT předávaná na volání je dokončena s prvním zaznamenaným asynchronním varováním nebo s chybovým informací pro toto připojení. Pokud další chyby nebo varování následují za prvními, tyto hodnoty obvykle neupravují. Pokud však dojde k chybě s kódem dokončení CCWARN, dojde místo toho k následnému selhání s kódem dokončení CCFAIL.
2. Pokud se nevyskytly žádné chyby od té doby, kdy bylo připojení ustanoveno, nebo od posledního volání MQSTAT, pak se vrátí CMPCOD z CCOK a REASON z RCNONE.
3. Počty počtu asynchronních volání, která byla zpracována pod manipulátorem připojení, jsou vrácena pomocí tří čítačů: STSPSC, STSPWC a STSPFC. Tyto čítače jsou zvyšovány správcem front při každém zpracování asynchronní operace, která má varování nebo selže (všimněte si, že pro účely účtování se na distribuční seznam místo jednou na seznam rozdělení počítá na distribuční seznam jednou).
4. Úspěšné volání MQSTAT má za následek reset všech předchozích informací o chybě nebo resetování.

Parametry

Volání MQSTAT má následující parametry:

Hconn (MQHCONN)-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *Hconn* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

STYPE (10ciferné 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 (Status reporting structure) v systému IBM i" na stránce 1217.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení.

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CMPCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CMPCOD* CCFAIL:

RC2374
(2374, X' 946 ')-ukončení rozhraní API se nezdařilo

RC2183
(2183, X'887 ') Nelze načíst uživatelskou proceduru rozhraní API.

RC2219
(2219, X'8AB') Volání MQI bylo zadáno před dokončením předchozího volání.

RC2009
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

RC2203
(2203, X'89B') Spojení se vypíná.

RC2018
(2018, X'7E2') Popisovač připojení není platný.

RC2162
(2162, X'872 ') Zastavení správce front

RC2102
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

RC2430
(2430, X'97E') Chyba s typem MQSTAT.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2424
(2424, X' 978 ') Chyba struktury MQSTS

RC2195
(2195, X'893 ') Došlo k neočekávané chybě.

RC2298
(2298, X'8FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

Podrobné informace o těchto kódech najdete v tématech:

- Zprávy a kódy příčiny

Deklarace 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 753. 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.

Tabulka 753. Výstup MQSD z MQSUB (pokračování)	
Název pole v MQSD	**Speciální aspekty.**
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 1200.

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 754. Použití Hobj s různými možnostmi předplatného

Volby	**HOBJ**	**Popis**
SOCRT + SOMAN	Ignorováno na vstupu	Vytvoří odběr se spravovaným úložištěm zpráv správce front.
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.

Tabulka 754. Použití Hobj s různými možnostmi předplatného (pokračování)		
Volby	**HOBJ**	**Popis**
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) v systému IBM i

Volání MQSUBRQ provede požadavek na odběr.

Syntaxe

MQSUBRQ *(HCONN, HSUB, ACTION, SUBROPT, CMPCOD, REASON)*

Poznámky k použití

Pro použití požadavku na službu SRAPUB se vztahují následující poznámky k použití:

1. Je-li toto příkazové slovo dokončeno úspěšně, zachované publikace odpovídající uvedenému odběru byly odeslány na odběr a lze je přijmout pomocí příkazu MQGET nebo MQCB pomocí příkazu HOBJ vráceného v původním příkazu MQSUB, který vytvořil odběr.
2. Pokud téma přihlášené k odběru původního příkazu MQSUB, které vytvořilo daný odběr, obsahovalo zástupný znak, může být odeslán více než jeden zachovaný publikování. Počet publikování odeslaných jako výsledek tohoto volání se zaznamenává do pole *SRNMP* ve struktuře SBROPT.
3. Pokud je toto příkazové slovo dokončeno s kódem příčiny RC2437 , nebyly v současné době pro uvedené téma uvedeny žádné aktuálně zachované publikace.
4. Je-li toto slovo dokončeno s kódem příčiny RC2525 nebo RC2526 , jsou v současné době pro uvedené téma aktuálně zachované publikace, ale došlo k chybě, že to znamená, že nebylo možné doručit.
5. Aplikace musí mít aktuální odběr pro dané téma, než bude moci toto volání provést. Pokud byl odběr proveden v předchozí instanci aplikace a není k dispozici platný popisovač pro daný odběr, musí aplikace nejprve zavolat funkci MQSUB s volbou SORES, aby mohla získat popisovač pro použití v rámci tohoto volání.
6. Publikace se posílají na místo určení, které je registrováno pro použití s aktuálním odběrem této aplikace. Pokud by měla být publikování odeslána někde jinde, je třeba nejprve změnit odběr pomocí volání MQSUB s volbou SOALT.

Parametry

Volání MQSUBRQ má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Tento manipulátor představuje připojení ke správci front. Vrácena hodnota *HCONN* byla vrácena předchozím voláním MQCONN nebo MQCONNX.

V produktu z/OS pro aplikace CICS lze volání MQCONN vynechat a pro produkt *HCONN* je určena následující hodnota:

HCDEFH
Výchozí popisovač připojení.

HSUB (10ciferné celé číslo se znaménkem)-vstup

Tento popisovač představuje odběr, pro který má být požadována aktualizace. Hodnota *HSUB* byla vrácena z předchozího volání MQSUB.

ACTION (10ciferné celé číslo se znaménkem)-vstup

Tento parametr řídí konkrétní akci, která je požadována na odběru. Musí být zadán jeden (a pouze jeden) z následujících:

SRAPUBA

Tato akce požaduje odeslání publikování aktualizací pro uvedené téma. Tato hodnota se obvykle používá, pokud odběratel určil volbu SOPUBR při volání MQSUB při odběru odběru. Má-li správce front zachované publikování pro dané téma, odešle se tomuto odběrateli. Pokud tomu tak není, volání selže. Je-li aplikace odeslána publikování, která byla uchována, je tato publikace označena vlastností zprávy MQIsRetained této publikace.

Vzhledem k tomu, že téma ve stávajícím odběru představovaném argumentem **HSUB** může obsahovat zástupné znaky, může odběratel obdržet více zachovaných publikování.

SBROPT (MQSRO)-vstup/výstup

Tyto volby řídí akci MQSUBRQ, podrobnosti viz “MQSRO-Volby požadavku na odběr” na stránce 583 .

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení; je to jeden z následujících:

KEK
Úspěšné dokončení

CCWARN
Varování (částečné dokončení)

CCFIL
Volání se nezdařilo

Důvod (10ciferné celé číslo se znaménkem)-výstup

Kód příčiny kvalifikující *CMPCOD*.

Pokud má parametr *CPMCOD* hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CPMCOD* CCFAIL:

RC2298
2298 (X'08FA') Požadovaná funkce není k dispozici v aktuálním prostředí.

RC2437
2437 (X'0985 ') Pro toto téma nejsou aktuálně uložena žádná zachovaná publikování.

RC2046
2046 (X'07FE') Parametr nebo pole voleb obsahuje volby, které nejsou platné, nebo kombinace voleb, které nejsou platné.

RC2161
2161 (X'0871 ') Správce front-uvedení do klidového stavu

RC2438
2438 (X'0986 ') V rámci volání MQSUBRQ není volba MQSRO požadavku na odběr platná.

Deklarace 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

Tato kolekce témat uvádí pouze ty objekty IBM MQ , které mohou být předmětem volání funkce MQINQ, a uvádí podrobnosti o atributech, které lze požadovat, a selektory, které se mají použít.

Atributy pro fronty

Tyto informace použijte k seznámení se s různými typy definic front a s atributy, které jsou podporovány jednotlivými typy front.

Typy front: Správce front podporuje následující typy definic front:

Lokální fronta
Jedná se o fyzickou frontu, do které jsou ukládány zprávy. Fronta existuje v lokálním správci front.

Aplikace připojené k lokálnímu správci front mohou umísťovat zprávy a odebírat zprávy z front tohoto typu. Hodnota atributu fronty **QType** je QTLOC.

Sdílená fronta
Jedná se o fyzickou frontu, do které jsou ukládány zprávy. Fronta se nachází ve sdíleném úložišti, které je přístupné všem správcům front patřícím do skupiny sdílení front, která je vlastníkem sdíleného úložiště.

Aplikace připojené k libovolnému správci front ve skupině sdílení front mohou umisťovat zprávy do front tohoto typu a odebírat zprávy z fronty tohoto typu. Takové 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, do které jsou ukládány zprávy. Fronta existuje buď v lokálním správci front, nebo na jednom či více správcích front, které patří ke stejnému 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, chová se tato fronta stejným způsobem jako lokální fronta a aplikace připojené k lokálnímu správci front mohou z fronty odebírat zprávy. Hodnota atributu fronty **QType** je QTCLUS.

Fronta aliasů
Nejedná se o fyzickou frontu-jedná se o alternativní název pro lokální frontu. Název lokální fronty, do níž je rozlišen alias, je součástí definice alias fronty.

Aplikace připojené k lokálnímu správci front mohou umisťovat zprávy do front aliasů a odebírat je z fronty aliasů-zprávy jsou umístěny a odstraněny z lokální fronty, na kterou je alias interpretován. 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 říkají lokálnímu správci front, jak 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 odebrat zprávy ze vzdálených front. Hodnota atributu fronty **QType** je QTREM.

Definice vzdálené fronty může být také použita pro:

- Aliasy fronty odpovědí

V tomto případě je název definice názvem fronty pro odpověď. Další informace naleznete v tématu Definice aliasů pro fronty odpovědí.

- Aliasy správce front

  V tomto případě je název definice alias pro správce front, nikoli název 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 nemohou být uloženy ve frontách tohoto typu.

Některé atributy fronty platí pro všechny typy front. Ostatní atributy fronty se vztahují pouze na určité typy front. Typy front, na které se atribut vztahuje, jsou označeny "X" v Tabulka 755 na stránce 1354 a následných tabulkách.

Tabulka 755 na stránce 1354 shrnuje atributy, které jsou specifické pro fronty. Atributy jsou popsány v abecedním pořadí.

Názvy atributů, které jsou zobrazeny v tabulce, jsou názvy použité s voláními MQINQ a MQSET. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; podrobnosti najdete v příkazech MQSC .

V následující tabulce se sloupce použijí takto:

- Sloupec pro lokální fronty platí také pro sdílené fronty.
- Sloupec pro modelové fronty označuje, které atributy jsou děděny lokální frontou vytvořenou z modelové fronty.
- Sloupec pro fronty klastru označuje atributy, které mohou být dotazovány, když je fronta klastru otevřena pro dotaz samostatně nebo pro zjištění a výstup. Je-li fronta klastru otevřena pro dotaz s jedním nebo více vstupními, procházením nebo sadou, použije se místo toho sloupec pro lokální fronty.

Tabulka 755. 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	
BackoutRequeue	Nadměrný název fronty vrácených zpráv	X	X			
BackoutThreshold	Práh vrácení	X	X			
BaseQName	Název fronty, na kterou se rozlišuje alias			X		
ClusterChannelNázev	Název kanálu odesílatele klastru	✓	✓			
ClusterName	Název klastru, do kterého fronta patří	X		X	X	

Tabulka 755. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
ClusterNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, do kterých fronta patří	X		X	X	
CreationDate	Datum, kdy byla fronta vytvořena	X				
CreationTime	Čas, kdy byla fronta vytvořena	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			
MediaLog	Identita nejstaršího rozsahu protokolu (nebo nejstarší žurnálový zásobník na systému IBM i) potřebný pro zotavení média uvedené fronty	✓	✓			
MsgDeliverySequence	Pořadí doručení zpráv	X	X			

Tabulka 755. Atributy pro fronty (pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
OpenInputCount	Počet otevření pro vstup	X				
OpenOutputCount	Počet otevření pro výstup	X				
ProcessName	Název procesu	X	X			
QDepthHighEvent	Řídí, zda jsou generovány události vysoké hloubky fronty	X	X			
QDepthHighLimit	Horní mez hloubky fronty	X	X			
QDepthLowEvent	Řídí, zda jsou generovány události nízké hloubky fronty	X	X			
QDepthLowLimit	Dolní mez hloubky fronty	X	X			
QDepthMaxEvent	Řídí, zda jsou generovány úplné události 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			
UdálostQServiceInterval	Řídí, zda jsou generovány události OK intervalu služby nebo intervalu služby 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			
Obor	Určuje, zda položka pro frontu také existuje v adresáři buňky.	X		X	X	
Možnost sdílení	Možnost sdílení front	X	X			
TriggerControl	Řízení spouštěče	X	X			
TriggerData	Data spouątěče	X	X			
TriggerDepth	Hloubka spouątěče	X	X			

Tabulka 755. Atributy pro fronty (pokračování)						
Atribut	**Popis**	**Lokální**	**Model**	**Alias**	**Vzdálený**	**Klastr**
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) v systému IBM i*

Datum, kdy byla definice naposledy změněna.

Tabulka 756. Typy front, na které se vztahuje tento atribut				
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 jeden prázdný znak).

Hodnoty určitých atributů (například *`CurrentQDepth`*) se mění s tím, jak pracuje správce front. Změny těchto atributů nemají vliv na *`AlterationDate`*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTD s voláním MQINQ. Délka tohoto atributu je dána LNDATE.

IBM i *AlterationTime (8bajtový znakový řetězec) v systému IBM i*

Čas, kdy byla definice naposledy změněna.

Tabulka 757. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Jedná se o čas, kdy byla definice naposledy změněna. Formát času je `HH.MM.SS` pomocí 24hodinového formátu, s počáteční nulou, je-li hodina menší než 10 (například `09.10.20`). Čas je místní čas.

Hodnoty určitých atributů (například *`CurrentQDepth`*) se mění s tím, jak pracuje správce front. Změny těchto atributů nemají vliv na *`AlterationTime`*.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délku tohoto atributu dává LNTIME.

IBM i *BackoutRequeueQName (48-bajtový znakový řetězec) na IBM i*

Nadměrný název fronty vrácených zpráv.

Tabulka 758. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Aplikace spuštěné uvnitř WebSphere Application Server a ty, které používají IBM MQ Application Server Facilities, používají tento atribut k určení toho, kam se mají vrátit zprávy, které byly vráceny. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu, kromě toho, že umožňuje dotazování na jeho hodnotu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CABRQN s voláním MQINQ. Délka tohoto atributu je dána LNQN.

IBM i *BackoutThreshold (10ciferné celé číslo se znaménkem) v IBM i*

Prahová hodnota vyřazených zpráv.

Tabulka 759. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Aplikace běžící uvnitř produktu WebSphere Application Server a ty, které používají IBM MQ Application Server Facilities, používají tento atribut k určení, zda by měla být vrácena zpráva. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu, kromě toho, že umožňuje dotazování na jeho hodnotu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IABTHR s voláním MQINQ.

IBM i *BaseQName (48-bajtový znakový řetězec) v systému IBM i*

Název fronty, na kterou je alias vyřešen.

Tabulka 760. Typy front, na které se vztahuje tento atribut				
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 MQOD. Fronta je jedním 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 LNQN.

IBM i *BaseType (struktura parametrů celého čísla) v systému IBM i*

Typ objektu, na který je alias vyřešen.

Tabulka 761. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
		X		

Tento atribut může mít některou z následujících hodnot:

OTQ
Základní typ objektu je fronta

OTOPU
Základní typ objektu je téma

IBM i *CFStrucName (12bajtový znakový řetězec) v systému IBM i*

Název struktury prostředku Coupling Facility.

Tabulka 762. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o název struktury prostředku Coupling Facility, ve které jsou uloženy zprávy ve frontě. První znak jména je v rozsahu A až Z a zbývající znaky jsou v rozsahu A až Z, 0 až 9, nebo prázdné.

Úplný název struktury ve spojovacím zařízení je získán přidáním hodnoty atributu správce front **QSGName** s hodnotou atributu fronty produktu **CFStrucName** .

Tento atribut se používá 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 LNCFSN.

z/OS Tento atribut je podporován pouze v 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 763. Typy front, na které se vztahuje tento atribut				
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 (48-bajtový znakový řetězec) na IBM i*

Název klastru, do kterého fronta patří.

Tabulka 764. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Jedná se o název klastru, do kterého fronta patří. Pokud fronta patří do více než jednoho klastru, *ClusterNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a *ClusterName* je prázdný. 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 LNCLUN.

IBM i *ClusterNamelist (48-bajtový znakový řetězec) v systému IBM i*

Název objektu seznamu názvů obsahujícího názvy klastrů, do kterých fronta patří.

Tabulka 765. Typy front, na které se vztahuje tento atribut				
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, objekt seznamu názvů obsahuje pouze jeden název. Alternativně lze *ClusterName* použít k uvedení 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 LNNLN.

IBM i *CreationDate (12bajtový znakový řetězec) v systému IBM i*

Datum, kdy byla fronta vytvořena.

Tabulka 766. Typy front, na které se vztahuje tento atribut				
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¬¬, ¬ představuje jeden prázdný znak).

- V systému IBM ise datum vytvoření fronty může lišit od data vytvoření fronty základního operačního systému (soubor nebo uživatelská oblast), 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) v systému IBM i*

Čas, kdy byla fronta vytvořena.

Tabulka 767. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Toto je čas, kdy byla fronta vytvořena. Formát času je `HH.MM.SS` pomocí 24hodinového formátu, s počáteční nulou, je-li hodina menší než 10 (například `09.10.20`). Čas je místní čas.

- V systému IBM ise čas vytvoření fronty může lišit od času vytvoření entity základního operačního systému (soubor nebo uživatelský prostor), který představuje frontu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACRTT s voláním MQINQ. Délka tohoto atributu je dána LNCRTT.

IBM i *CurrentQDepth (10ciferné celé číslo se znaménkem) v IBM i*

Aktuální hloubka fronty.

Tabulka 768. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Jedná se o počet zpráv aktuálně uložených ve frontě. Během volání MQPUT se inkrementuje a během odvolání se volání MQGET znovu zobrazí. Je snižován během volání operace MQGET bez procházení a během odvolání volání MQPUT. Výsledkem je, že počet zahrnuje zprávy, které byly vloženy do fronty v rámci pracovní jednotky, ale které ještě nebyly potvrzeny, i když nejsou způsobilé k načtení voláním MQGET. Podobně vyloučí zprávy, které byly získány v rámci transakce pomocí volání MQGET, ale které ještě nebyly potvrzeny.

Počet také zahrnuje zprávy, které předaly svůj čas vypršení platnosti, ale ještě nebyly vyřazeny, ačkoli tyto zprávy nejsou způsobilé k načtení. Viz pole *MDEXP* popsané v části “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099.

Zpracování jednotek práce a segmentace zpráv může způsobit, že *CurrentQDepth* překročí *MaxQDepth*. To však neovlivňuje dostupnost zpráv- *všechny* zprávy ve frontě je možné načíst pomocí volání MQGET běžným způsobem.

Hodnota tohoto atributu kolísá, jak pracuje správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACDEP s voláním MQINQ.

IBM i *DefBind (10ciferné celé číslo se znaménkem) v IBM i*

Výchozí vazba.

Tabulka 769. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	X

Tento atribut je výchozí vazba, která se použije, je-li v volání MQOPEN zadán OOBNDQ a fronta je fronta klastru. DefBind může mít jednu z následujících hodnot:

BNDOPN
Vazba byla opravena voláním MQOPEN.

BNDNOT
Vazba nebyla opravena.

BNDGRP
Vazba není opravena voláním MQOPEN, ale je pevně nastavena na operaci MQPUT pro všechny zprávy v logické skupině.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADBND s voláním MQINQ.

IBM i *DefinitionType (10ciferné celé číslo se znaménkem) v IBM i*

Typ definice fronty.

Tabulka 770. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Označuje, jak byla fronta definována. Hodnota je jedna z následujících možností:

QDPRE
Předdefinovaná trvalá fronta.

Fronta je trvalá fronta vytvořená administrátorem systému; tuto frontu může odstranit pouze administrátor systému.

Předdefinované fronty se vytvářejí pomocí příkazu DEFINE MQSC a lze je odstranit pouze pomocí příkazu MQSC DELETE . Předdefinované fronty nelze vytvořit z modelových front.

Příkazy může být vydáno buď operátorem, nebo autorizovaným uživatelem odesláním zprávy příkazu do vstupní fronty příkazů (viz atribut **CommandInputQName** popsaný v "Atributy pro správce front v systému IBM i" na stránce 1384).

QDPERM
Dynamicky definovaná trvalá fronta.

Fronta je trvalá fronta, která byla vytvořena aplikací, která vydala volání MQOPEN s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má hodnotu QDPERM pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část "MQCLOSE (Zavření objektu) na IBM i" na stránce 1254.

Hodnota atributu **QSGDisp** pro trvalou dynamickou frontu je QSGDQM.

QDTEMP
Dynamicky definovaná dočasná fronta.

Fronta je dočasná fronta vytvořená aplikací, která vydala volání MQOPEN, s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má 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 vytvořená aplikací, která vydala volání MQOPEN, s názvem modelové fronty zadané v deskriptoru objektu MQOD. Definice modelové fronty má hodnotu QDSHAR pro atribut **DefinitionType** .

Tento typ fronty lze odstranit pomocí volání MQCLOSE. Další informace viz část "MQCLOSE (Zavření objektu) na IBM i" na stránce 1254.

Hodnota atributu **QSGDisp** pro sdílenou dynamickou frontu je QSGDSH.

Tento atribut v definici modelové fronty neukazuje, jak byla modelovaná fronta definována, protože modelové fronty jsou vždy předdefinované. Místo toho se hodnota tohoto atributu ve frontě modelu používá k určení *DefinitionType* každé z dynamických front vytvořených z definice modelové fronty pomocí volání MQOPEN.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADEFT s voláním MQINQ.

IBM i *DefInputOpenOption (10ciferné celé číslo se znaménkem) v IBM i*

Výchozí vstupní otevřená volba.

Tabulka 771. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o výchozí způsob, jak by se měla fronta otevřít pro vstup. Používá se, pokud je při volání MQOPEN zadána volba OOINPQ, když je fronta otevřena. 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 , je-li fronta aktuálně otevřena touto nebo jinou aplikací pro vstup libovolného typu (OOINPS 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 , je-li fronta momentálně otevřená s OOINPX.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADINP s voláním MQINQ.

IBM i *DefPersistence (10ciferné celé číslo se znaménkem) v IBM i*

Výchozí trvalost zpráv.

Tabulka 772. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	X

Jedná se o výchozí trvání zpráv ve frontě. Použije se, je-li hodnota PEQDEF uvedena v deskriptoru zpráv, když je zpráva vložena.

Pokud v cestě rozpoznání názvu fronty existuje více než jedna definice, bude použita výchozí perzistence z hodnoty tohoto atributu v cestě *první* v cestě v době volání MQPUT nebo MQPUT1 . To může být:

- Fronta aliasů
- Lokální fronta

- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*)

Může mít jednu z následujících hodnot:

PÍPČ
Zpráva je trvalá.

To znamená, že zpráva přečká 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 předdefinovaných front.

PENPER
Zpráva není trvalá.

To znamená, že zpráva normálně nepřežije selhání systému nebo restartuje správce front. To platí i v případě, že se během restartu správce front nachází neporušená kopie zprávy v pomocné paměti.

Ve speciálním případě sdílených front přežijí přechodné zprávy *do* restarty správců front ve skupině sdílení front, ale nepřežijí selhání prostředku Coupling Facility použitého k ukládání zpráv ve sdílených frontách.

Trvalé i přechodné zprávy mohou existovat ve stejné frontě.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADPER s voláním MQINQ.

IBM i *DefPriority (10ciferné celé číslo se znaménkem) v IBM i*

Výchozí priorita zprávy.

Tabulka 773. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Jedná se o výchozí prioritu zpráv ve frontě. To platí, je-li hodnota PRQDEF uvedena v deskriptoru zpráv, když je zpráva vložena do fronty.

Pokud je v cestě rozpoznání názvu fronty uvedena více než jedna definice, bude z hodnoty tohoto atributu použita výchozí priorita z hodnoty atributu v *první* definici v cestě v čase operace vložení. To může být:

- Fronta aliasů
- Lokální fronta
- Lokální definice vzdálené fronty
- Alias správce front
- Přenosová fronta (například fronta *DefXmitQName*)

Způsob, jakým je zpráva umístěna ve frontě, závisí na hodnotě atributu **MsgDeliverySequence** fronty:

- Je-li atribut **MsgDeliverySequence** MSPRIO, logická pozice, ve které je zpráva umístěna do fronty, závisí na hodnotě pole *MDPRI* v deskriptoru zpráv.
- Je-li atribut **MsgDeliverySequence** MSFIFO, jsou zprávy umístěny do fronty, jako by měly prioritu rovnající se *DefPriority* z vyřešené fronty, bez ohledu na hodnotu pole *MDPRI* v deskriptoru zpráv. Pole *MDPRI* si však zachovává hodnotu určenou aplikací, která vložila zprávu. Další informace naleznete v popisu atributu **MsgDeliverySequence** popsaného v tématu “Atributy pro fronty” na stránce 1353 .

Priority jsou v rozsahu nula (nejnižší) až *MaxPriority* (nejvyšší); viz atribut **MaxPriority** popsaný v “Atributy pro správce front v systému IBM i” na stránce 1384.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADPRI s voláním MQINQ.

IBM i *DefReadAhead (10-číslicové celé číslo se znaménkem) v IBM i*

Určuje výchozí chování dopředného čtení pro netrvalé zprávy doručené klientovi.

Tabulka 774. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X		

Volba DefReadAhead může být nastavena na jednu z následujících hodnot:

RAHNO
Netrvalé zprávy nejsou odeslány klientovi před tím, než je aplikace požaduje. Pokud klient skončí abnormálně, dojde ke ztrátě maximálně jedné netrvalé zprávy.

RAHYBY
Netrvalé zprávy jsou odeslány před klientem před tím, než je aplikace požaduje. Netrvalé zprávy mohou být ztraceny, pokud klient skončí abnormálně, nebo pokud klient nespotřebuje všechny zprávy, které odeslal.

RAHDIS
Čtení předem netrvalých zpráv pro tuto frontu není povoleno. Zprávy se do klienta neodesílají 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 podepsané celé číslo) v IBM i*

Atribut výchozí typ vložení odezvy (DEFPRESP) definuje hodnotu použitou aplikacemi, když byl PutResponseType v rámci MQPMO nastaven na PMRASQ. Tento atribut je platný pro všechny typy front.

Tabulka 775. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	X

Může mít jednu z následujících hodnot:

SYNC
Operace umístění je vydána synchronně po vrácení odezvy.

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 (10ciferné celé číslo se znaménkem) v IBM i*

Podpora distribučního seznamu.

Tabulka 776. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Označuje, zda mohou být do fronty umístěny zprávy distribučního seznamu. Atribut je nastaven agentem kanálu zpráv (MCA), který informuje lokálního správce front o tom, zda správce front na druhém konci

kanálu podporuje distribuční seznamy. Tento posledně jmenovaný správce front (nazývaný "partnering queue manager") je ten, který obdrží zprávu poté, co byla odebrána z lokální přenosové fronty odesílajícím programem MCA.

Atribut je nastaven odesílající agent MCA při každém vytvoření připojení k přijímajícímu agentovi MCA v rámci partnerského správce front. Tímto způsobem odesílající agent MCA může způsobit, že lokální správce front bude v přenosové frontě umístěn pouze zprávy, které může partnerský správce front zpracovat správně.

Tento atribut se primárně používá pro přenosové fronty, ale popsané zpracování se provede bez ohledu na využití definované pro frontu (viz atribut **Usage**).

Může mít jednu z následujících hodnot:

DLSUPP
Podporované seznamy distribucí.

Tato zpráva informuje o tom, že zprávy distribučních seznamů lze uložit do fronty a přenést do správce front partnera v daném formuláři. Tím se snižuje objem zpracování potřebný k odeslání zprávy do více míst určení.

PLNUP
Distribuční seznamy nejsou podporovány.

To znamená, že zprávy distribučního seznamu nelze uložit do fronty, protože partnerský správce front nepodporuje distribuční seznamy. Pokud aplikace umístí zprávu distribučního seznamu a tato zpráva má být umístěna do této fronty, správce front rozdělí zprávu distribučního seznamu a umístí jednotlivé zprávy do fronty místo ní. Tím se zvyšuje objem zpracování potřebný k odeslání zprávy do více míst určení, ale zajišťuje, že zprávy budou zpracovány správně správcem front partnering.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADIST s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *HardenGetBackout (10ciferné celé číslo se znaménkem) v IBM i*

Zda se má udržovat přesný počet vrácení.

Tabulka 777. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Pro každou zprávu je počet uchován z počtu případů, kdy je zpráva načtena pomocí volání MQGET v rámci pracovní jednotky a tato jednotka práce později byla vrácena. Tento počet je k dispozici v poli *MDBOC* v deskriptoru zpráv po dokončení volání MQGET.

Počet vrácení zprávy přežije, když se správce front restartuje. Chcete-li však zajistit, aby byl počet přesný, musí být informace "upřesněné" (zaznamenané na disku nebo jiné trvalé paměťové jednotce) při každém načtení zprávy voláním MQGET v rámci pracovní jednotky pro tuto frontu. Pokud k tomu nedojde a dojde-li k selhání správce front spolu s odvolání volání MQGET, může se počet zvýšit.

Zahazování informací pro každé volání MQGET v rámci jednotky práce však vynucuje náklady na výkon a atribut **HardenGetBackout** by měl být nastaven na hodnotu QABH pouze v případě, že má být tento počet přesný.

- V systému IBM ije počet odvolání zpráv vždy tvrzený, bez ohledu na nastavení tohoto atributu.

Možné jsou následující hodnoty:

QABH
Počet vrácení je zapamatován.

Zaměření se používá k ujištění, že počet vrácení pro zprávy v této frontě je přesný.

QABNH
Je možné, že nebude zapamatován počet vrácení.

Zahradničení se nepoužívá, aby se zajistilo, že počet vrácení pro zprávy v této frontě je přesný. Počet by proto mohl být nižší, než by měl být.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAHGB s voláním MQINQ.

IBM i *InhibitGet (10ciferné celé číslo se znaménkem) v IBM i*

Určuje, zda jsou povoleny operace get pro tuto frontu.

Tabulka 778. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X		

Je-li fronta alias fronta, musí být operace získání povoleny pro alias i pro základní frontu v době operace get, aby se volání MQGET mělo úspěšně provést. Hodnota je jedna z následujících možností:

QAGETINAME
Operace získání jsou blokovány.

Volání MQGET se nezdaří s kódem příčiny RC2016. To zahrnuje volání MQGET, která uvádí GMBRWF nebo GMBRWN.

Poznámka: Je-li operace MQGET pracující v rámci transakce úspěšně dokončena, změna hodnoty atributu **InhibitGet** po hodnotě QAGETI nezabrání tomu, aby byla jednotka práce potvrzena.

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 (10ciferné celé číslo se znaménkem) v IBM i*

Určuje, zda jsou povoleny operace vložení pro tuto frontu.

Tabulka 779. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X	X	X	X

Je-li v cestě rozpoznání názvu fronty uvedena více než jedna definice, musí být operace vložení povoleny pro *každou* definici v cestě (včetně všech definic aliasů správců front) v době operace vložení, aby bylo volání MQPUT nebo MQPUT1 úspěšné. Může mít jednu z následujících hodnot:

QAPUTI
Operace vložení jsou blokovány.

Volání MQPUT a MQPUT1 se nezdařily s kódem příčiny RC2051.

Poznámka: Je-li volání MQPUT fungující v rámci transakce úspěšně dokončeno, změna hodnoty atributu **InhibitPut** později na QAPUTI nezabrání tomu, aby byla jednotka práce potvrzena.

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 (48-bajtový znakový řetězec) v IBM i*

Název inicializační fronty.

Tabulka 780. Typy front, na které se vztahuje tento atribut				
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 do inicializační fronty zprávu spouštěče, je-li jako výsledek zprávy přicházející do fronty, do níž tento atribut náleží, vyžadováno spuštění aplikace. 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 LNQN.

IBM i *MaxMsgDélka (10ciferné celé číslo se znaménkem) v IBM i*

Maximální délka zprávy v bajtech.

Tabulka 781. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o horní limit délky nejdelší *fyzické* zprávy, kterou lze umístit do fronty. Vzhledem k tomu, že atribut fronty **MaxMsgLength** lze nastavit nezávisle na atributu správce front produktu **MaxMsgLength** , je menší z těchto dvou hodnot skutečný horní limit délky nejdelší fyzické zprávy, kterou lze umístit do fronty.

Pokud správce front podporuje segmentaci, je možné, aby aplikace umístila *logickou* zprávu, která je delší než menší než menší ze dvou atributů **MaxMsgLength** , ale pouze v případě, že aplikace určuje příznak MFSEGA v MQMD. Je-li tento parametr zadán, horní mez pro délku logické zprávy je 999 999 999 bajtů, obvykle však omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, vede k nižšímu limitu.

Pokus o umístění do fronty, která je příliš dlouhá, selže s kódem příčiny:

- RC2030 , je-li zpráva příliš velká pro frontu
- RC2031 , je-li zpráva příliš velká pro správce front, ale není příliš velká pro frontu

Dolní limit atributu **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 tématu “MQPUT (vložení zprávy) na IBM i” na stránce 1318.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMLEN s voláním MQINQ.

IBM i *MaxQDepth (10ciferné celé číslo se znaménkem) v IBM i*

Maximální hloubka fronty.

Tabulka 782. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o definovaný horní limit počtu fyzických zpráv, které mohou být ve frontě v daném okamžiku vůbec existovat. Pokus o vložení zprávy do fronty, která již obsahuje zprávy *MaxQDepth* , selže s kódem příčiny RC2053.

Zpracování jednotek práce a segmentace zpráv může způsobit, že skutečný počet fyzických zpráv ve frontě překročí *MaxQDepth*. To však neovlivňuje dostupnost zpráv- *všechny* zprávy ve frontě je možné načíst pomocí volání MQGET běžným způsobem.

Hodnota tohoto atributu je nula nebo větší. Horní limit je určen prostředím.

Poznámka: Je možné, aby byl úložný prostor dostupný pro frontu vyčerpán i v případě, že ve frontě je méně než *MaxQDepth* zpráv.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMDEP s voláním MQINQ.

IBM i *MediaLog (10ciferné celé číslo se znaménkem) v IBM i*

Identita rozsahu protokolu (nebo příjemce žurnálu na IBM i) potřebných k obnově média určité fronty.

Tabulka 783. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Ve správcích front, kde se používá kruhové protokolování, je hodnota vrácena jako prázdný řetězec.

IBM i *Posloupnost MsgDelivery(10ciferné celé číslo se znaménkem) v IBM i*

Sekvence doručení zpráv.

Tabulka 784. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

To určuje pořadí, ve kterém jsou zprávy vraceny do aplikace voláním MQGET:

MSFIFO
Zprávy jsou vraceny ve FIFO pořadí (první dovnitř, první ven).

To znamená, že volání MQGET vrátí zprávu *první* , která splňuje kritéria výběru uvedená ve volání, bez ohledu na prioritu zprávy.

MSPRIO
Zprávy jsou vraceny v pořadí priority.

To znamená, že volání MQGET vrátí zprávu *highest-priority* , která splňuje kritéria výběru zadaná ve volání. V rámci každé úrovně priority jsou zprávy vraceny ve FIFO pořadí (první dovnitř, první ven).

Pokud se příslušné atributy změní, když se ve frontě nacházejí zprávy, je posloupnost doručení následující:

- Pořadí, ve kterém jsou zprávy vraceny voláním MQGET, jsou určovány hodnotami atributů **MsgDeliverySequence** a **DefPriority** platných pro frontu v době, kdy zpráva dorazí do fronty:

– Má-li parametr *MsgDeliverySequence* hodnotu MSFIFO při doručení zprávy, bude zpráva vložena do fronty, jako by její priorita byla *DefPriority*. To nemá vliv na hodnotu pole *MDPRI* v deskriptoru zprávy této zprávy; v tomto poli je zachována hodnota, kterou měla při prvním vložení zprávy.
– Je-li *MsgDeliverySequence* MSPRIO při doručení zprávy, je zpráva umístěna do fronty na místě odpovídajícím prioritě zadané argumentem *MDPRI* v deskriptoru zprávy.

Pokud se změní hodnota atributu **MsgDeliverySequence** , zatímco se ve frontě nacházejí zprávy, pořadí zpráv ve frontě se nezmění.

Pokud se změní hodnota atributu **DefPriority** , zatímco ve frontě jsou zprávy, zprávy nebudou nutně doručeny v pořadí FIFO, i když je atribut **MsgDeliverySequence** nastaven na MSFIFO; ty, které byly umístěny do fronty při vyšší prioritě, jsou dodány jako první.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAMDS s voláním MQINQ.

IBM i *Počet OpenInputPočet (10ciferné celé číslo se znaménkem) v IBM i*

Počet otevření pro vstup.

Tabulka 785. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Jedná se o počet popisovačů, které jsou aktuálně platné pro odebrání zpráv z fronty s voláním MQGET. Jedná se o celkový počet těchto popisovačů známých pro *lokálního* správce front. Je-li fronta sdílenou frontou, tento počet nezahrne otevření pro vstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do níž patří lokální správce front.

Počet zahrnuje manipulátory, ve kterých byla pro vstup otevřena fronta aliasů, která byla rozpoznána pro tuto frontu. Počet nezahrnuje manipulátory, ve kterých byla fronta otevřena pro akce, které neobsahovaly vstup (například, fronta otevřená pouze pro procházení).

Hodnota tohoto atributu kolísá, jak pracuje správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAOIC s voláním MQINQ.

IBM i *OpenOutputPočet (10ciferné celé číslo se znaménkem) v IBM i*

Počet operací otevření pro výstup.

Tabulka 786. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X				

Jedná se o počet popisovačů, které jsou aktuálně platné pro přidání zpráv do fronty s voláním MQPUT. Jedná se o celkový počet takových manipulátorů, které jsou známy správci front *local* ; nezahrne se otevření pro výstup, který byl proveden pro tuto frontu ve vzdálených správcích front. Je-li fronta sdílenou frontou, tento počet nezahrnuje otevření pro výstup, který byl proveden pro frontu v jiných správcích front ve skupině sdílení front, do níž patří lokální správce front.

Počet zahrnuje manipulátory, ve kterých byla pro výstup otevřena fronta aliasů, která byla přeložena do této fronty. Počet nezahrnuje manipulátory, kde byla fronta otevřena pro akce, které neobsahovaly výstup (například, fronta byla otevřena pouze pro zjištění).

Hodnota tohoto atributu kolísá, jak pracuje správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAOOC s voláním MQINQ.

IBM i *ProcessName (48bajtový znakový řetězec) v systému IBM i*

Název procesu.

Tabulka 787. Typy front, na které se vztahuje tento atribut				
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 službu zařadit do fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAPRON s voláním MQINQ. Délka tohoto atributu je dána LNPRON.

IBM i *QDepthHighUdálost (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události vysoké hloubky fronty.

Tabulka 788. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Příliš dlouhá fronta označuje, že aplikace vložila zprávu do fronty, která způsobila, že se počet zpráv ve frontě stal 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
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDHE s voláním MQINQ.

IBM i *Limit QDepthHigh(10-číslicové celé číslo se znaménkem) v IBM i*

Horní mez hloubky fronty.

Tabulka 789. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o prahovou hodnotu, proti níž je porovnávána hloubka fronty pro generování události Příliš dlouhá fronta. Tato událost označuje, že aplikace umístila zprávu do fronty a způsobila, že se počet zpráv ve frontě stal větší nebo roven horní prahové hodnotě hloubky fronty. Viz atribut **QDepthHighEvent** .

Hodnota je vyjádřena jako procentní část z maximální hloubky fronty (atribut**MaxQDepth**) a je v rozsahu od nuly do 100. Výchozí hodnota je 80.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDHL s voláním MQINQ.

IBM i *Událost QDepthLow(10ciferné celé číslo se znaménkem) v systému IBM i*

Řídí, zda jsou generovány události nízké hloubky fronty.

Tabulka 790. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Příliš dlouhá fronta označuje, že aplikace načetla zprávu z fronty, která způsobila, že se počet zpráv ve frontě stal méně nebo roven dolní prahové hodnotě hloubky fronty (viz atribut **QDepthLowLimit**).

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

QDepthLowUdálost může mít jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDLE s voláním MQINQ.

IBM i *Limit QDepthLowLimit (10místný číslicový integer) na IBM i*

Dolní mez hloubky fronty.

Tabulka 791. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o prahovou hodnotu, proti níž je porovnávána hloubka fronty, aby se vygenerovala událost Nízká hloubka fronty. Tato událost označuje, že aplikace načetla zprávu z fronty, a to způsobilo, že se počet zpráv ve frontě stal méně než nebo roven dolní prahové hodnotě hloubky fronty. Viz atribut **QDepthLowEvent** .

Hodnota je vyjádřena jako procentní část z maximální hloubky fronty (atribut**MaxQDepth**) a je v rozsahu od nuly do 100. Výchozí hodnota je 20.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDLL s voláním MQINQ.

IBM i *QDepthMaxUdálost (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány úplné události fronty.

Tabulka 792. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Událost Plná fronta indikuje, že vložení do fronty bylo zamítnuto, protože fronta je plná, to znamená, že hloubka fronty již dosáhla maximální hodnoty.

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Může mít jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQDME s voláním MQINQ.

IBM i *QDesc (64bajtový znakový řetězec) v IBM i*

Popis fronty.

Tabulka 793. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X	X	X	X

Toto je pole, které lze použít pro popisný komentář. Obsah pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace 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.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQD s voláním MQINQ. Délka tohoto atributu je dána LNQD.

IBM i *QName (48-bajtový znakový řetězec) v IBM i*

Název fronty.

Tabulka 794. Typy front, na které se vztahuje tento atribut				
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 pojmenování objektů produktu IBM MQ. 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 LNQN.

IBM i *QServiceInterval (10ciferné celé číslo se znaménkem) v IBM i*

Cíl pro interval služby fronty.

Tabulka 795. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je interval služby použitý pro porovnání ke generování událostí Vysoká a servisní interval Interval služby OK. Viz atribut **QServiceIntervalEvent** .

Hodnota je v milisekundách, a je v rozsahu od nuly do 999 999 999.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSI s voláním MQINQ.

IBM i *Událost QServiceInterval(10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události vysokého nebo servisního intervalu servisního intervalu.

Tabulka 796. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

- Vysoká událost Interval služby se generuje, když kontrola označuje, že od fronty nebyly načteny žádné zprávy alespoň po dobu uvedenou atributem **QServiceInterval** .
- Událost Interval služby OK je generována, pokud kontrola indikuje, že zprávy byly získány z fronty v čase indikovaném atributem **QServiceInterval** .

Poznámka: Hodnota tohoto atributu se může dynamicky měnit.

Tento atribut může mít některou z následujících hodnot:

QSIEŠTINA
Události vysoké intervalu služby fronty povoleny.

- Události vysoké intervalu služby fronty jsou **povoleny** a
- Události servisního intervalu fronty OK jsou **zakázány**.

QSIEOK
Události OK intervalu služby fronty povoleny.

- Události vysoké intervalu služby fronty jsou **zakázány** a
- Události servisního intervalu fronty OK jsou **povoleny**.

QSIENJA
Nejsou povoleny žádné události intervalu služby fronty.

- Události vysoké intervalu služby fronty jsou **zakázány** a
- Události servisního intervalu fronty OK jsou také **zakázány**.

Pro sdílené fronty je hodnota tohoto atributu ignorována; předpokládá se hodnota QSIENO.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSIE s voláním MQINQ.

IBM i *QSGDisp (10ciferné celé číslo se znaménkem) v IBM i*

Dispozice skupiny sdílení front.

Tabulka 797. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Určuje dispozice fronty. Hodnota je jedna z následujících možností:

QSGDQM
Dispozice správce front.

Objekt má dispozice správce front. To znamená, že definice objektu je známa pouze lokálnímu správci front; definice není známa ostatním správcům front ve skupině sdílení front.

Každému správci front ve skupině sdílení front je možné mít objekt se stejným názvem a typem jako aktuální objekt, ale tyto objekty jsou samostatné objekty a mezi nimi neexistuje žádná korelace. Jejich atributy nejsou omezeny na to, aby byly stejné jako ostatní.

QSGDCP
Dispozice kopírovaného objektu.

Objekt je lokální kopií definice hlavního objektu, který existuje ve sdíleném úložišti. Každý správce front ve skupině sdílení front může mít vlastní kopii daného objektu. Zpočátku mají všechny kopie stejné atributy, ale pomocí příkazů MQSC lze každou kopii změnit tak, aby se její atributy odlišovaly od atributů ostatních kopií. Atributy kopií se znovu synchronizují, když se změní hlavní definice ve sdíleném úložišti.

QSGDSH
Sdílené odebrání.

Objekt má sdílené odebrání. To znamená, že ve sdíleném úložišti existuje jediná instance objektu, která je známá všem správcům front ve skupině sdílení front. Přistupuje-li správce front v dané skupině k objektu, bude přistupovat k jedné sdílené instanci objektu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAQSGD s voláním MQINQ.

z/OS Tento atribut je podporován pouze v systému z/OS.

IBM i *QType (10číslicové celé číslo se znaménkem) v IBM i*

Typ fronty.

Tabulka 798. Typy front, na které se vztahuje tento atribut

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 (48-bajtový znakový řetězec) v systému IBM i*

Název vzdáleného správce front.

Tabulka 799. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
			X	

Jedná se o název vzdáleného správce front, na kterém je definována fronta *RemoteQName* . Má-li 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 prázdná a nesmí se jednat o název lokálního správce front. Je-li parametr *XmitQName* prázdný, 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 produktu **DefXmitQName** .

Je-li tato definice použita pro alias správce front, *RemoteQMgrName* je název správce front, pro který je alias vytvořen. Může se jednat o název lokálního správce front. Jinak, je-li *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 odpovědi na alias, je tento název názvem správce front, který má být *MDRM*.

Poznámka: Při vytváření nebo úpravě definice fronty není prováděno žádné ověřování pro hodnotu určenou pro tento atribut.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARQMN s voláním MQINQ. Délka tohoto atributu je dána LNQMN.

IBM i *RemoteQName (48-bajtový znakový řetězec) v systému IBM i*

Název vzdálené fronty.

Tabulka 800. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
			X	

Jedná se o název fronty, jak je znám ve vzdáleném správci front *RemoteQMgrName*.

Pokud aplikace otevře lokální definici vzdálené fronty, když se otevřená vyskytuje, *RemoteQName* nesmí být prázdné.

Je-li tato definice použita pro definici aliasu správce front, musí být při otevření prázdná hodnota *RemoteQName* .

Je-li definice použita pro alias odpovědi na alias, je tento název názvem fronty, která má být *MDRQ*.

Poznámka: Při vytváření nebo úpravě definice fronty není prováděno žádné ověřování pro hodnotu určenou pro tento atribut.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CARQN s voláním MQINQ. Délka tohoto atributu je dána LNQN.

IBM i *RetentionInterval (10ciferné celé číslo se znaménkem) v IBM i*

Interval uchování.

Tabulka 801. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je doba, po kterou by měla být fronta zadržena. Po uplynutí této doby je fronta vhodná k odstranění.

Čas se měří v hodinách, počítáno od data a času, kdy byla fronta vytvořena. Datum vytvoření fronty je zaznamenáno v *CreationDate* a čas vytvoření fronty je zaznamenán v atributu **CreationTime** .

Tyto informace jsou poskytnuty, aby umožnily aplikaci úklidu nebo operátorovi identifikovat a odstranit fronty, které již nejsou zapotřebí.

Poznámka: Správce front se nikdy nepokusí o odstranění front na základě tohoto atributu nebo k zabránění odstranění front s intervalem uchování, jehož platnost dosud neskončila; je odpovědností uživatele, aby byla přijata veškerá požadovaná akce.

Realistický retenční interval by měl být použit k zabránění hromadění trvalých dynamických front (viz *DefinitionType*). Tento atribut lze však také použít s předdefinovanými frontami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IARINT pomocí volání MQINQ.

IBM i *Rozsah (10ciferné celé číslo se znaménkem) v IBM i*

Určuje, zda položka pro tuto frontu také existuje v adresáři buňky.

Tabulka 802. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X		X	X	

Adresář buňky je poskytován instalovatelnou službou názvů. Může mít jednu z následujících hodnot:

SCOQM
Obor správce front.

Definice fronty má obor správce front. To znamená, že definice fronty není rozšířena nad rámec správce front, který ji vlastní. Chcete-li otevřít frontu pro výstup z jiného správce front, je třeba zadat buď název vlastnícího správce front, nebo musí mít jiný správce front lokální definici fronty.

SKÚČ
Obor buňky.

Definice fronty má obor buňky. To znamená, že definice fronty je umístěna také v adresáři buňky, který je k dispozici všem správcům front v buňce. Frontu lze otevřít pro výstup z libovolného správce front v rámci buňky pouze zadáním názvu fronty. Název správce front, který tuto frontu vlastní, nemusí být zadán. Definice fronty však není k dispozici pro žádného správce front v buňce, která má také lokální definici fronty s tímto názvem, protože lokální definice má přednost.

Adresář buňky je poskytován instalovatelnou službou názvů, jako je LDAP (Lightweight Directory Access Protocol). Všimněte si, že produkt IBM MQ již nepodporuje službu názvů DCE (Distributed Computing Environment), která byla dříve použita pro vložení definic front do adresáře DCE (také již není podporováno).

Model 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.

Na podporu tohoto atributu se vztahují následující omezení:

- V systému IBM ije tento atribut podporován, je však platný pouze parametr SCOQM.

IBM i *Sdílitelnost (10ciferné celé číslo se znaménkem) v IBM i*

Zda lze frontu sdílet pro vstup.

Tabulka 803. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Označuje, zda lze frontu otevřít pro vstup vícenásobně souběžně. Může mít jednu z následujících hodnot:

QASHSTAR
Fronta je možné sdílet.

Vícenásobné otevření s volbou OOINPS je povoleno.

QANSHR
Fronta není možné sdílet.

Volání MQOPEN s volbou OOINPS je považováno za OOINPX.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASHAR s voláním MQINQ.

IBM i *TriggerControl (10ciferné celé číslo se znaménkem) v IBM i*

Řízení spouštěče.

Tabulka 804. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Tento příkaz určuje, zda se zprávy spouštěče zapisují do inicializační fronty, aby bylo možné spustit aplikaci ke zpracování této fronty. Toto je jedna z následujících možností:

TKOFF
Spouštěcí zprávy nejsou povinné.

Pro tuto frontu se nemají zapsat žádné zprávy spouštěče. Hodnota *TriggerType* je v tomto případě irelevantní.

TCON
Vyžadované zprávy spouštěče.

Zprávy spouštěče se mají zapsat pro tuto frontu, když dojde k odpovídajícím 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 (64-bajtový znakový řetězec) v systému IBM i*

Data spouątěče.

Tabulka 805. Typy front, na které se vztahuje tento atribut				
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, že zpráva spouštěče bude zapsána do inicializační fronty.

Obsah těchto dat nemá význam pro správce front. Je smysluplný buď pro aplikaci monitoru spouštěčů, která zpracovává inicializační frontu, nebo aplikaci, která je spuštěna monitorem spouštěčů.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se 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 LNTRGD.

IBM i *TriggerDepth (10ciferné celé číslo se znaménkem) v IBM i*

Hloubka spouątěče.

Tabulka 806. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Jedná se o počet zpráv s prioritou *TriggerMsgPriority* nebo vyšší, které musí být ve frontě, než se vypíše zpráva spouštěče. To platí, je-li parametr *TriggerType* nastaven na TTHDPTH. Hodnota *TriggerDepth* je jedna nebo více. Tento atribut se nepoužívá jinak.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGD s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *TriggerMsgPriorita (10ciferné celé číslo se znaménkem) v IBM i*

Prahová hodnota priority zpráv pro spouštěče v produktu IBM MQ for IBM i.

Tabulka 807. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Toto je priorita zprávy, pod níž zprávy nepřispívají ke generování zpráv spouštěče (to znamená, že správce front tyto zprávy ignoruje při zjišťování, zda by měla být generována zpráva spouštěče). *TriggerMsgPriority* může být v rozsahu nula (nejnižší) až *MaxPriority* (vysocet; viz “Atributy pro správce front v systému IBM i” na stránce 1384); hodnota nula způsobí, že všechny zprávy přispívají k generaci 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 (10ciferné celé číslo se znaménkem) v IBM i*

Typ spouątěče.

Tabulka 808. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
X	X			

Tím se řídí podmínky, za kterých jsou zprávy spouštěče zapisovány jako výsledek zpráv přicházejících do této fronty. Hodnota je jedna z následujících možností:

TTNONE
Žádné zprávy spouštěče.

Žádné zprávy spouštěče se nezapisují jako výsledek zpráv v této frontě. To má stejný účinek jako nastavení *TriggerControl* na TCOFF.

TTFRST
Spustit zprávu v případě, že hloubka fronty přejde od 0 do 1.

Zpráva spouštěče se zapisuje vždy, když se počet zpráv priority *TriggerMsgPriority* nebo vyšší ve frontě změní z 0 na 1.

TEVRY
Zpráva spouštěče pro každou zprávu.

Zpráva spouštěče se zapisuje vždy, když se do fronty dostane zpráva o prioritě *TriggerMsgPriority* nebo vyšší.

TDPTH
Spustit zprávu, když je překročena prahová hodnota hloubky.

Zpráva spouštěče se zapisuje vždy, když se počet zpráv priority *TriggerMsgPriority* nebo vyšší na frontě rovná nebo překročí *TriggerDepth*. Po zapsání zprávy spouštěče je produkt *TriggerControl* nastaven na hodnotu TCOFF, aby se zabránilo dalšímu spouštění, dokud nebude explicitně znovu zapnuto.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGT s voláním MQINQ. Chcete-li změnit hodnotu tohoto atributu, použijte volání MQSET.

IBM i *Použití (10ciferné celé číslo se znaménkem) v IBM i*

Použití fronty.

Tabulka 809. Typy front, na které se vztahuje tento atribut

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Označuje, pro kterou frontu se používá fronta. Hodnota je jedna z následujících možností:

UNORM
Normální využití.

Jedná se o frontu, kterou běžné aplikace používají při vkládání a získávání zpráv; fronta není přenosová fronta.

USTRAN.
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 tuto zprávu dočasně do příslušné přenosové fronty ve speciálním formátu. Agent kanálu zpráv poté přečte zprávu z přenosové fronty a odešle zprávu do vzdáleného správce front. Další informace o přenosových frontách najdete v tématu Přenosové fronty.

Pouze privilegované aplikace mohou otevřít přenosovou frontu pro OOOUT, aby se do ní vložila zprávy přímo. Za normálních okolností by se od těchto aplikací očekávalo, že to bude dělat. Je třeba dbát na to, aby formát dat zprávy byl správný (viz "MQXQH (záhlaví přenosové fronty) v systému IBM i" na stránce 1232), jinak by během procesu přenosu mohly nastat chyby. Kontext není předáván 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 (48-bajtový znakový řetězec) v systému IBM i*

Název přenosové fronty.

Tabulka 810. Typy front, na které se vztahuje tento atribut				
Lokální	**Model**	**Alias**	**Vzdálený**	**Klastr**
			X	

Je-li tento atribut neprázdný, když se vyskytne otevření, buď pro vzdálenou frontu, nebo pro definici alias správce front, uvádí jméno lokální přenosové fronty, která má být použita pro předání zprávy.

Je-li parametr *XmitQName* prázdný, 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 produktu **DefXmitQName** .

Tento atribut je ignorován, je-li definice použita 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 LNQN.

Atributy pro seznamy názvů

Toto téma shrnuje atributy, které jsou specifické pro seznamy názvů. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy uvedených 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, doplněno 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 LNDATE.

AlterationTime (8bajtový znakový řetězec)

Čas, kdy byla definice naposledy změněna.

Jedná se o čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délku tohoto atributu dává LNTIME.

NameCount (10ciferné celé číslo se znaménkem)

Počet názvů v seznamu názvů.

Tato hodnota je větší než nula 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 řetězec znaků)

Popis seznamu názvů.

Toto je pole, které může být použito pro popisný komentář; jeho hodnota je vytvořena definičním procesem. Obsah pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole

obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace DBCS může toto pole obsahovat znaky DBCS (s výhradou maximální délky 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.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CALSTD s voláním MQINQ.

Délka tohoto atributu je dána parametrem LNNLD.

NamelistName (48-bajtový znakový řetězec)

Název seznamu názvů.

Jedná se o název seznamu názvů, který je definován v lokálním správci front.

Každý seznam názvů má název odlišný od názvů jiných seznamů názvů náležejících ke správci front, ale mohou duplikovat názvy jiných objektů správce front různých typů (například front).

Chcete-li určit hodnotu tohoto atributu, použijte selektor CALSTN s voláním MQINQ.

Délka tohoto atributu je dána LNNLN.

Názvy (48-bajtový znakový řetězec x NameCount)

Seznam názvů *NameCount* .

Každý název představuje název objektu, který je definován pro lokálního správce front. Další informace o názvech objektů najdete v tématu Pojmenování objektů IBM MQ.

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 procesu v systému IBM i

Toto téma shrnuje atributy, které jsou specifické pro definice procesu. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy uvedených atributů jsou názvy použité s voláními MQINQ a MQSET. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; podrobnosti najdete v příkazech 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, doplněno 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 LNDATE.

AlterationTime (8bajtový znakový řetězec)

Čas, kdy byla definice naposledy změněna.

Jedná se o čas, kdy byla definice naposledy změněna. Formát času je HH.MM.SS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délku tohoto atributu dává LNTIME.

ApplId (256bajtový znakový řetězec)

Identifikátor aplikace.

Jedná se o znakový řetězec identifikující aplikaci, která má být spuštěna. Tyto informace používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě; informace se odesílají do inicializační fronty jako část zprávy spouštěče.

Význam *ApplId* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný serverem IBM MQ vyžaduje, aby byl *ApplId* název spustitelného programu.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAAPPI s voláním MQINQ. Délka tohoto atributu je dána LNPROA.

ApplType (10ciferné celé číslo se znaménkem)

Typ aplikace.

Označuje povahu programu, který má být spuštěn v odezvě na přijetí zprávy spouštěče. Tyto informace používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě; informace se odesílají do inicializační fronty jako část zprávy spouštěče.

ApplType může mít libovolnou hodnotu. Pro standardní typy můžete použít následující hodnoty; uživatelem definované typy aplikací jsou omezeny na hodnoty v rozsahu ATUFST přes ATULST:

rovnoCICS
CICS .

AT400
IBM i .

ATUFST
Nejnižší hodnota pro typ aplikace definovaný uživatelem.

ATULSTCITY
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 používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě; informace se odesílají do inicializační fronty jako část zprávy spouštěče.

Význam *EnvData* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytnutý produktem IBM MQ připojuje *EnvData* k seznamu parametrů předanému do spuštěné aplikace. Seznam parametrů se skládá ze struktury MQTMC2 , za nímž následuje jedna mezera, následované *EnvData* s odstraněnými koncovými mezerami.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAENVD s voláním MQINQ. Délka tohoto atributu je dána LNPROE.

ProcessDesc (64bajtový znakový řetězec)

Popis procesu.

Toto je pole, které lze použít pro popisný komentář. Obsah tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace 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.

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 ke správci front. Ale název definice procesu může být stejný jako názvy jiných objektů správce front různých typů (například fronty).

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAPRON s voláním MQINQ.

Délka tohoto atributu je dána 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 používá aplikace monitoru spouštěčů, která zpracovává zprávy v inicializační frontě, nebo aplikaci spouštěnou monitorem spouštěčů. Informace se odešlou do inicializační fronty jako část zprávy spouštěče.

Význam *UserData* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný produktem IBM MQ předává *UserData* do spuštěné aplikace jako součást seznamu parametrů. Seznam parametrů se skládá ze struktury MQTMC2 (obsahující *UserData*), za nímž následuje jedna mezera, za kterou následuje *EnvData* s odebranými koncovými mezerami.

Znakový řetězec nemůže obsahovat žádné hodnoty null. Je-li to nutné, doplní se vpravo mezerami.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAUSRD s voláním MQINQ. Délka tohoto atributu je dána 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`. Většina atributů správce front může být dotazovaná otevřením speciálního objektu OTQM a pomocí volání MQINQ s vráceným handle.

Následující tabulka shrnuje atributy, které jsou specifické pro správce front. Atributy jsou popsány v abecedním pořadí.

Poznámka: Názvy atributů, které jsou zobrazeny v této sekci, jsou názvy použité s voláními MQINQ a MQSET. Když se příkazy MQSC používají k definování, změně nebo zobrazení atributů, použijí se alternativní krátké názvy; další informace najdete v tématu Příkazy MQSC .

Tabulka 811. Atributy 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 (neautorizované)
BridgeEvent	Ovládá, zda jsou generovány události mostu IMS

Tabulka 811. Atributy správce front (pokračování)	
Atribut	**Popis**
ChannelAutoDef	Řídí, zda je povolena automatická definice kanálu
ChannelAutoDefEvent	Řídí, zda jsou generovány události automatické definice kanálu
ChannelAutoDefExit	Název uživatelské procedury pro automatické definování kanálů
ChannelEvent	Řídí, zda jsou generovány události kanálu
ClusterCacheTyp	Řídí, zda je mezipaměť klastru pevně nastavena ve velikosti nebo dynamicky.
ClusterWorkloadData	Uživatelská data pro uživatelskou proceduru pracovní zátěže klastru
ClusterWorkloadExit	Název uživatelské procedury pro správu pracovní zátěže klastru
ClusterWorkloadLength	Maximální délka dat zpráv předaných uživatelskou proceduru pracovní zátěže klastru
CodedCharSetId	Identifikátor znakové sady
CommandEvent	Řídí, zda jsou zprávy událostí příkazů zařazeny do fronty
CommandInputQName	Název fronty vstupu příkazů
CommandLevel	Úroveň příkazů
ConfigurationEvent	Událost konfigurace
DeadLetterQName	Název fronty nedoručených zpráv
DefClusterXmitQueueTyp	Výchozí typ přenosové fronty klastru
DefXmitQName	Výchozí název přenosové fronty
DistLists	Podpora seznamu distribuce
InhibitEvent	Řídí, zda jsou generovány události inhibice (Inhibit Get a Inhibit Put)
LocalEvent	Řídí, zda jsou generovány lokální chybové události
LoggerEvent	Řídí, zda jsou generovány události protokolu o zotavení
MaxHandles	Maximální počet popisovačů
MaxMsgLength	Maximální délka zprávy v bajtech
MaxPriority	Maximální priorita
MaxUncommittedMsgs	Maximální počet nepotvrzených zpráv v rámci jednotky práce
PerformanceEvent	Řídí, zda jsou generovány události související s výkonem
Platforma	Platforma, na které je správce front spuštěn.
PubSubMode	Zda je spuštěn stroj pro publikování/odběr 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 události vzdálené chyby
RepositoryName	Název klastru, pro který tento správce front poskytuje služby úložiště

Tabulka 811. Atributy správce front (pokračování)	
Atribut	**Popis**
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)
PočetSSLKeyResetCount	Určuje počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán šifrovací klíč
StartStopEvent	Řídí, zda jsou generovány události spuštění a zastavení
SyncPoint	Dostupnost synchronizačního bodu
TraceRouteRecording	Ovládá záznam informací o přenosové cestě trasování pro zprávy
TreeLifeTime	Životnost neadministrativních témat v sekundách
TriggerInterval	Trigger-interval zpráv

Notes:

1. Tento atribut nelze provést pomocí volání MQINQ a není popsán v této sekci. Další informace o tomto atributu naleznete v tématu Změna správce front.

IBM i *AlterationDate (12bajtový znakový řetězec) v systému IBM i*

Datum, kdy byla definice naposledy změněna.

Toto je datum, kdy byla definice naposledy změněna. Formát data je `YYYY-MM-DD`, doplněno 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 LNDATE.

IBM i *AlterationTime (8bajtový znakový řetězec) v systému IBM i*

Čas, kdy byla definice naposledy změněna.

Jedná se o čas, kdy byla definice naposledy změněna. Formát času je `HH.MM.SS`.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAALTT s voláním MQINQ. Délku tohoto atributu dává LNTIME.

IBM i *AuthorityEvent (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události autorizace (neautorizované).

Atribut AuthorityEvent musí být nastaven na jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAAUTE s voláním MQINQ.

IBM i *BridgeEvent (znakový řetězec) v systému IBM i*

Tento atribut určuje, zda jsou zprávy událostí mostu IMS vloženy do systému SYSTEM.ADMIN.CHANNEL.EVENT . Je podporován pouze v produktu z/OS.

IBM i *ChannelAutoDef (10-číslicové celé číslo se znaménkem)IBM i*

Řídí, 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 (10ciferné celé číslo se znaménkem) v IBM i*

Určuje, zda jsou generovány události automatické definice kanálu.

To platí pro kanály typu CTRCVR, CTSVCN a CTCLSD. Může mít jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech najdete v tématu Monitorování a výkon.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACADE s voláním MQINQ.

IBM i *ChannelAutoDefExit (20bajtový znakový řetězec) v systému IBM i*

Název uživatelské procedury pro automatické definování kanálu.

Pokud je tento název neprázdný a *ChannelAutoDef* má hodnotu CHADEN, je uživatelská procedura volána vždy, když se správce front chystá vytvořit definici kanálu. To platí pro kanály typu CTRCVR, CTSVCN a CTCLSD. Ukončení může poté provést jednu z následujících možností:

- Povolit vytvoření definice kanálu pro pokračování beze změn.
- Upravte atributy definice kanálu, která je vytvořena.
- Zcela potlačte vytvoření kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACADX s voláním MQINQ. Délka tohoto atributu je dána LNEXN.

IBM i *ChannelEvent (znakový řetězec) v systému IBM i*

Určuje, zda jsou generovány zprávy událostí kanálu.

Tento atribut určuje, zda jsou zprávy událostí kanálu vloženy do systému SYSTEM.ADMIN.CHANNEL.EVENT fronta, a pokud ano, jaký typ zpráv je zařazen do fronty (například 'kanál spuštěn', 'kanál zastaven', 'kanál není aktivován'). Před implementací tohoto atributu bylo jediným způsobem, jak zabránit ve frontě zpráv událostí kanálu, aby byla odstraněna cílová fronta.

Tento atribut vám také umožňuje shromažďovat pouze události mostu IMS (protože nyní můžete vypnout události kanálu, neukládejte se do stejné fronty). To samé platí pro události TLS, které lze také shromažďovat, aniž by bylo nutné shromažďovat také události kanálu.

Tento atribut vám také umožňuje shromažďovat pouze důležité události (například, když kanály obsahují chyby, ne když se spouští a zastavují normálně).

Hodnota atributu ChannelEvent může mít jednu z následujících hodnot:

- EVREXP (Jsou generovány pouze následující události kanálu: RC2279, RC2283, RC2284, RC2295, RC2296).
- EVRENA (všechny události kanálu jsou generovány; tj. kromě událostí vygenerovaných produktem 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) v systému IBM i*

Určuje, zda má mezipaměť klastru pevnou velikost nebo je dynamicky nastavena na velikost.

Jedná se o uživatelsky definovaný 32bajtový řetězec znaků, který je předán uživatelské proceduře pracovní zátěže klastru, když je volán. Nejsou-li k dispozici žádná data pro předání do procedury ukončení, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWD s voláním MQINQ.

IBM i *ClusterWorkloadData (32bajtový znakový řetězec) v systému IBM i*

Uživatelská data pro uživatelskou proceduru pracovní zátěže klastru.

Jedná se o uživatelsky definovaný 32bajtový řetězec znaků, který je předán uživatelské proceduře pracovní zátěže klastru, když je volán. Nejsou-li k dispozici žádná data pro předání do procedury ukončení, řetězec je prázdný.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWD s voláním MQINQ.

IBM i *ClusterWorkloadUkončení (20bajtový znakový řetězec) v systému IBM i*

Název uživatelské procedury pro správu pracovní zátěže klastru.

Pokud tento název není prázdný, je uživatelská procedura volána při každém vložení zprávy do fronty klastru nebo přesunu z jedné fronty odesílatele klastru do jiné fronty. Uživatelská procedura pak může buď přijmout instanci fronty vybranou správcem front jako místo určení zprávy, nebo vybrat jinou instanci fronty.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACLWX s voláním MQINQ. Délka tohoto atributu je dána LNEXN.

IBM i *ClusterWorkloadDélka (10ciferné celé číslo se znaménkem) v IBM i*

Maximální délka dat zpráv předaných uživatelskou proceduru pracovní zátěže klastru.

Jedná se o maximální délku dat zpráv, která jsou předána uživatelské proceduře pracovní zátěže klastru. Skutečná délka dat předaných do uživatelské procedury je minimálně:

- Délka zprávy.
- Atribut **MaxMsgLength** správce front.
- Atribut **ClusterWorkloadLength** .

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACLWL s voláním MQINQ.

IBM i *CodedCharSetId (10ciferné celé číslo se znaménkem) v IBM i*

Identifikátor kódované znakové sady.

Definuje znakovou sadu používanou správcem front pro všechna pole znakového řetězce, která jsou definována v rozhraní MQI, jako jsou například názvy objektů a datum a čas vytvoření fronty. Znaková sada musí být taková, která má jednobajtové znaky pro znaky, které jsou platné v názvech objektů. Nevztahuje se na data aplikace přenášené ve zprávě. Hodnota závisí na prostředí:

- V systému IBM ise jedná o hodnotu, která je 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) v systému IBM i*

Řídí, zda jsou zprávy vloženy do lokální fronty, když jsou vydány příkazy.

To řídí, zda se zprávy zapisují do nové fronty událostí, SYSTEM.ADMIN.COMMAND.EVENT, kdykoli jsou vydány příkazy. Tato funkce je užitečná pro oznámení o sledování příkazů a pro diagnostiku problémů. Chcete-li se dotázat na atribut správce front CommandEvent , použijte nový selektor atributu iacev s jednou z následujících hodnot:

- EVRENA-zprávy událostí příkazů jsou generovány a vloženy do fronty pro všechny úspěšné příkazy.
- EVND-zprávy událostí příkazů jsou generovány a vloženy do fronty pro všechny úspěšné příkazy jiné než příkaz DISPLAY (MQSC) a příkaz Inquire (PCF).
- EVRDIS-zprávy událostí příkazů nejsou generovány ani vloženy 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 (48-bajtový znakový řetězec) v 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. Avšak do této fronty lze odeslat příkaz MQSC, pokud je příkaz MQSC uzavřen v rámci příkazu PCF typu CMESC. Další informace o příkazu Escape naleznete v části Escape.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CACMDQ s voláním MQINQ. Délka tohoto atributu je dána LNQN.

IBM i *CommandLevel (10ciferné celé číslo se znaménkem) v IBM i*

Úroveň příkazů. Značí úroveň příkazů řízení 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 řídicích příkazů systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - verze 9.1

CML920
Úroveň 920 řídicích příkazů systému.

Tato hodnota je vrácena následujícími aplikacemi:

- IBM MQ for IBM i
  - verze 9.2

Nastavení řídicích příkazů systému, které odpovídají určité hodnotě atributu **CommandLevel** , se liší v závislosti na hodnotě atributu **Platform** ; oba musí být použity při rozhodování o tom, které řídicí příkazy systému jsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IACMDL s voláním MQINQ.

IBM i *ConfigurationEvent v systému IBM i*

Řídí, zda jsou generovány události konfigurace a odeslány do SYSTEM.ADMIN.CONFIG.EVENT Výchozí objekt fronty.

Atribut ConfigurationEvent může mít jednu z následujících hodnot:

- EVRENA
- EVRDIS

Je-li atribut ConfigurationEvent nastaven na hodnotu EVRENA a některé příkazy jsou úspěšně vydány příkazy runmqsc nebo PCF, jsou generovány a odeslány konfigurační události do systému SYSTEM.ADMIN.CONFIG.EVENT . Vydávají se události pro následující příkazy, a to i v případě, že příkaz alter nemění daný objekt. Příkazy, pro které jsou generovány a odesílány konfigurační události, jsou:

- DEFINOVAT/ZMĚNIT AUTHINFO
- DEFINOVAT/ZMĚNIT KANÁL
- DEFINOVAT/ZMĚNIT SEZNAM NÁZVŮ
- DEFINOVAT/ZMĚNIT PROCES
- DEFINE/ALTER QLOCAL (pokud se nejedná o dočasnou dynamickou frontu)
- DEFINOVAT/ZMĚNIT PARAMETR QMODEL/QALIAS/QREMOTE
- ODSTRANIT AUTHINFO
- Odstranit kanál
- Odstranit seznam názvů
- Odstranit proces
- DELETE QLOCAL (pokud se nejedná o dočasnou dynamickou frontu)
- ODSTRANIT QMODEL/QALIAS/QREMOTE
- ALTER QMGR (pokud je atribut CONFIGEV zakázán a není změněn na povolený)
- AKTUALIZOVAT SPRÁVCE FRONT
- Volání MQSET, jiné než pro dočasnou dynamickou frontu.

Události se negenerují (je-li povoleno) za následujících okolností:

- Zpracování příkazu nebo volání MQSET selže.
- Správce front nemůže umístit zprávu události do fronty událostí. Příkaz by měl být stále úspěšně dokončen.
- Dočasné dynamické fronty.
- Změny interního atributu byly provedeny přímo nebo implicitně (nikoli příkazem MQSET nebo příkazem); to ovlivní TRIGGER, CURDEPTH, IPPROCS, OPPROCS, QDPHIEV, QDPLOEV, QDPMAXEV, QSVCIEV.
- Když se změní fronta událostí konfigurace, i když je požadována zpráva události pro tuto změnu, když se požaduje zpráva o události, která se změní.
- Změny klastrování prováděné příkazy REFRESH/RESET CLUSTER a RESUME/SUSPEND QMGR.
- Vytvoření nebo odstranění správce front.

IBM i *DeadLetterQName (48-bajtový znakový řetězec) v IBM i*

Název fronty smrtelného dopisu (nedoručená zpráva).

Jedná se o název fronty definované v lokálním správci front. Zprávy se odesílají do této fronty, pokud nemohou být směrovány na jejich správné místo určení.

Například zprávy jsou vloženy do této fronty, když:

- Zpráva dorazí do správce front, který je určen pro frontu, která dosud není definována v daném správci front.
- Zpráva dorazí do správce front, ale fronta, pro kterou je určena, ji nemůže přijmout, protože pravděpodobně:
  - Fronta je plná
  - Požadavky PUT jsou blokovány
  - Odesílající uzel nemá oprávnění vkládat zprávy do fronty

Aplikace mohou také vkládat zprávy do fronty nedoručených zpráv.

Zprávy sestav se zpracovávají stejným způsobem jako běžné zprávy; pokud nelze zprávu sestavy doručit do cílové fronty (obvykle do fronty zadané v poli *MDRQ* v deskriptoru zprávy původní zprávy), bude zpráva sestavy umístěna do fronty nedoručených zpráv (nedoručená zpráva).

Poznámka: Zprávy, které předaly svůj čas vypršení platnosti (viz pole *MDEXP* popsané v publikaci "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099) **nejsou** převedeny do této fronty, když jsou zahozeny. Avšak, zpráva o vypršení platnosti sestavy (ROEXP) je stále generována a odeslána do fronty *MDRQ* , pokud ji požaduje odesílající aplikace.

Zprávy nejsou vloženy do fronty nedoručených zpráv (nedoručená zpráva), pokud byla aplikace, která vydala požadavek na vložení, oznámena synchronně s kódem příčiny vráceným voláním MQPUT nebo MQPUT1 (například zpráva vložena do lokální fronty, pro kterou jsou blokovány žádosti).

Zprávy ve frontě nedoručených zpráv (undelivered-message) mají někdy k dispozici data zpráv aplikace s předponou ve struktuře MQDLH. Tato struktura obsahuje další informace, které ukazují, proč byla zpráva vložena do fronty nedoručených zpráv (nedoručená zpráva). Další podrobnosti o této struktuře viz "MQDLH (záhlaví nedoručených zpráv) v systému IBM i" na stránce 1054 .

Tato fronta musí být lokální fronta, s atributem **Usage** USNORM.

Pokud správce front nepodporuje frontu nedoručených zpráv (nedoručená zpráva), nebo nebyla definována, je název prázdný. Všichni správci front produktu IBM MQ podporují frontu nedoručených zpráv (nedoručená zpráva), ale při výchozím nastavení není definována.

Není-li definována fronta nedoručených zpráv (nedoručená zpráva) nebo je-li plná nebo nepoužitelná z nějakého jiného důvodu, bude místo toho v přenosové frontě zadržena zpráva, která by byla agentem kanálu zpráv přenesena do tohoto agenta.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CADLQ s voláním MQINQ. Délka tohoto atributu je dána LNQN.

DefClusterXmitQueueTyp (10ciferné 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.

`SCTQ` se nastaví při definici správce front. Toto chování je implicitní ve verzích produktu IBM WebSphere MQ před verzí IBM WebSphere MQ 7.5. Ve starších verzích nebyl parametr správce front `DefClusterXmitQueueType` přítomen.

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.

Je-li 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 na příkaz MQINQnebo odešlete příkaz PCF produktu Inquire Queue Manager (MQCMD_INQUIRE_Q_MGR), nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE . Chcete-li změnit hodnotu, odešlete příkaz PCF správce front změn (MQCMD_CHANGE_Q_MGR) a nastavte selektor MQIA_DEF_CLUSTER_XMIT_Q_TYPE .

Související odkazy
Změnit správce front
Zjistit správce front
“MQINQ (Dotaz na atributy objektů) v systému IBM i” na stránce 1291
Volání MQINQ vrací pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

IBM i *DefXmitQName (48-bajtový znakový řetězec) v IBM i*
Výchozí název přenosové fronty.

Jedná se o název přenosové fronty, která se používá pro přenos zpráv do vzdálených správců front, pokud neexistuje žádná jiná indikace toho, jakou přenosovou frontu použít.

Pokud neexistuje žádná předvolená přenosová fronta, jméno je zcela prázdné. Počáteční hodnota tohoto atributu je prázdná.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CADXQN s voláním MQINQ. Délka tohoto atributu je dána LNQN.

IBM i *DistLists (10ciferné celé číslo se znaménkem) v IBM i*
Podpora distribučního seznamu.

To označuje, zda lokální správce front podporuje distribuční seznamy na volání MQPUT a MQPUT1 . Může mít jednu z následujících hodnot:

DLSUPP
Podporované seznamy distribucí.

PLNUP
Distribuční seznamy nejsou podporovány.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IADIST s voláním MQINQ.

IBM i *InhibitEvent (10ciferné celé číslo se znaménkem) v IBM i*
Řídí, zda jsou generovány události blokování (Inhibit Get a Inhibit Put).

Může mít jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech najdete v tématu Monitorování a výkon.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAINHE s voláním MQINQ.

IBM i *LocalEvent (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány lokální chybové události.

Hodnota je jedna z následujících možností:

EVRDIS
Vytváření sestav událostí je 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 (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události modulu protokolování pro zotavení.

Může mít jednu z následujících hodnot:

POVOLENO
Události modulu protokolování jsou generovány.

VYPNUTO
Události modulu protokolování se negenerují. Jedná se o počáteční výchozí hodnotu správců front.

Další informace o událostech najdete v tématu Monitorování a výkon.

IBM i *MaxHandles (10ciferné celé číslo se znaménkem) v IBM i*

Maximální počet popisovačů.

Jedná se o maximální počet otevřených popisovačů, které může jedna úloha používat souběžně. Každé úspěšné volání MQOPEN pro jednu frontu (nebo pro objekt, který není frontou) používá jeden popisovač. Tento popisovač bude k dispozici pro opětovné použití, když je objekt uzavřen. Když je však otevřen distribuční seznam, každá fronta v rozdělovníku je alokována jako samostatná obsluha, takže volání MQOPEN používá tolik popisovačů, kolik je ve frontách v rozdělovníku. To musí být vzato v úvahu při rozhodování o vhodné hodnotě pro *MaxHandles*.

Volání MQPUT1 provádí volání MQOPEN jako součást jeho zpracování; v důsledku toho hodnota MQPUT1 používá tolik obslužných rutin jako MQOPEN, ale manipulátory jsou použity pouze po dobu trvání volání MQPUT1 .

Hodnota je v rozsahu od 1 do 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 (10ciferné celé číslo se znaménkem) v IBM i*

Maximální délka zprávy v bajtech.

Jedná se o délku nejdelší *fyzické* zprávy, kterou může správce front zpracovat. Vzhledem k tomu, že atribut správce front produktu **MaxMsgLength** lze nastavit nezávisle na atributu fronty produktu **MaxMsgLength** , je tato nejdelší fyzická zpráva, kterou lze umístit do fronty, menší z těchto dvou hodnot.

Pokud správce front podporuje segmentaci, je možné, aby aplikace umístila *logickou* zprávu, která je delší než menší než menší ze dvou atributů **MaxMsgLength** , ale pouze v případě, že aplikace určuje příznak MFSEGA v MQMD. Je-li tento parametr zadán, horní mez pro délku logické zprávy je 999 999 999 bajtů, ale zpravidla omezení prostředků uložená operačním systémem nebo prostředím, v němž je aplikace spuštěna, bude mít za následek nižší omezení.

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 (10ciferné celé číslo se znaménkem) v 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 *MaxUncommittedMsgs (10-digit signed integer) na IBM i*

Maximální počet nepotvrzených zpráv v rámci jednotky práce.

Toto je maximální počet nepotvrzených zpráv, které mohou existovat v rámci jednotky práce. Počet nepotvrzených zpráv je součtem následujících od začátku aktuální transakce:

- Zprávy uvedené aplikací s volbou PMSYP
- Zprávy načtené aplikací s volbou GMSYP
- Zprávy spouštěče a zprávy sestav COA generované správcem front pro zprávy zařazené s volbou PMSYP
- Zprávy COD zprávy generované správcem front pro zprávy načtené pomocí volby GMSYP

Následující zprávy se nepočítají jako nepotvrzené:

- Zprávy vkládané nebo načtené aplikací mimo jednotku práce
- Zprávy spouštěče nebo zprávy sestav COA/COD generované správcem front jako výsledek zpráv vložených nebo načtených mimo jednotku práce
- Zprávy sestavy vypršení platnosti generované správcem front (i v případě, že volání způsobující zprávu hlášení o vypršení platnosti specifikovanou GMSYP)
- Zprávy událostí generované správcem front (i v případě, že volání způsobující uvedenou zprávu události PMSYP nebo GMSYP)

Poznámka:

1. Zprávy o výjimkách jsou generovány agentem MCA (Message Channel Agent) nebo aplikací a jsou s nimi nakládáno stejným způsobem jako s běžnými zprávami vkládané nebo načítány 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 ve skutečnosti pochází z vložení. (Může nastat více než jedna fyzická zpráva, je-li správce front nutné rozdělit zprávu nebo segment.)
3. Když je distribuční seznam vložen s volbou PMSYP, počet nepotvrzených zpráv se zvýší o jedničku o jednu *pro každou vygenerovanou fyzickou zprávu*. Může být tak malý jako jeden nebo velký jako počet míst určení v rozdělovníku.

Spodní 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 (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události související s výkonem.

PerformanceEvent může mít jednu z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IAPFME s voláním MQINQ.

IBM i *Platforma (10ciferné celé číslo se znaménkem) v IBM i*

Platforma, na které je správce front spuštěn.

Označuje operační systém, na kterém je spuštěný správce front. Hodnota je:

PL400
IBM i.

IBM i *Režim PubSub(10ciferné celé číslo se znaménkem) v IBM i*

Určuje, zda je stroj pro publikování/odběr a rozhraní publikování/odběru zařazené do fronty spuštěné, a proto umožňuje aplikacím publikovat/přihlásit se k odběru prostřednictvím 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 se žádná zpráva, která je vložena do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, nepostupuje. Toto nastavení se používá pro kompatibilitu s WebSphere Message Broker V6 nebo staršími verzemi pomocí tohoto správce front, protože musí číst stejné fronty, ze kterých normálně čte rozhraní publikování/odběru ve frontě.

PSDS
Stroj pro publikování/odběr a rozhraní pro publikování/odběr ve frontě nejsou spuštěny. Proto není možné publikovat/přihlásit se k odběru pomocí rozhraní API. Jakékoli zprávy publish/subscribe, které jsou vloženy do front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě, nepracují.

PSMEN
Stroj publikování/odběru a rozhraní publikování/odběru ve frontě jsou spuštěny. Proto je možné publikovat/přihlásit se k odběru pomocí rozhraní API a front, které jsou monitorovány rozhraním pro publikování/odběr ve frontě. 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 tohoto pole nemá význam pro správce front, ale správce front může vyžadovat, aby pole obsahovalo pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace DBCS může toto pole obsahovat znaky DBCS (s výhradou maximální délky 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.

V systému IBM ije výchozí hodnotou prázdná hodnota.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQMD s voláním MQINQ. Délka tohoto atributu je dána LNQMD.

IBM i *QMgrIdentifier (48-bajtový znakový řetězec) v 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 LNQMID.

IBM i *QMgrName (48-bajtový znakový řetězec) v systému 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 “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099). Správci front, kteří mohou komunikovat, musí mít proto názvy, které se v prvních 12 znacích liší, aby identifikátory zpráv byly jedinečné v síti správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor CAQMN s voláním MQINQ. Délka tohoto atributu je dána LNQMN.

IBM i *RemoteEvent (10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události vzdálené chyby.

Hodnota je jedna z následujících možností:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IARMTE s voláním MQINQ.

IBM i *RepositoryName (48-bajtový znakový řetězec) v 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, *RepositoryNamelist* určuje název objektu seznamu názvů, který identifikuje klastry, a *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 LNQMN.

IBM i *RepositoryNamelist (48-bajtový znakový řetězec) v systému 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, objekt seznamu názvů obsahuje pouze jedno jméno. Alternativně lze *RepositoryName* použít k uvedení 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 LNNLN.

IBM i *SSLEvent (znakový řetězec) v systému IBM i*

Určuje, zda jsou generovány události TLS.

Hodnota je jedna z následujících možností:

- EVRENA (událost MQINQ/PCF/config event) ENABLED (MQSC): Jsou generovány události TLS (tj. událost RC2371 je generována).
- EVRDIS (MQINQ/PCF/config event) DISABLED (MQSC): Události TLS se negenerují. Jedná se o počáteční výchozí hodnotu správce front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSLE s voláním MQINQ.

IBM i *Počet SSLKeyResetPočet (celé číslo) v IBM i*

Určuje celkový počet nešifrovaných bajtů, které jsou odeslány a přijaty v rámci konverzace TLS, než bude znovu vyjednán tajný klíč. Počet bajtů zahrnuje řídicí informace odeslané agentem kanálu zpráv (MCA).

Tato hodnota je používána pouze pro kanál MCU kanálu TLS, který iniciuje komunikaci od tohoto správce front (tedy kanálu MCA kanálu odesílatele v rámci párování odesílatele a příjemce kanálu).

Je-li hodnota tohoto atributu větší než 0 a prezenční signály kanálu jsou povoleny pro kanál, je tajný klíč také znovu vyjednán před odesláním nebo přijímáním dat po prezenční signál kanálu. Počet bajtů do obnovení dalšího opětovného domlouvání tajného klíče po každé úspěšné nové domlouvání.

Hodnota může být v rozsahu od 0 do 999 999 999. Hodnota 0 pro tento atribut označuje, že tajný klíč není nikdy znovu vyjednáván. Určíte-li počet obnovení tajných klíčů TLS v rozsahu od 1 bajtu do 32 KB, budou kanály TLS používat počet obnovení tajných klíčů 32 KB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se mohly vyskytnout u malých hodnot resetu tajného klíče TLS.

Je-li server SSL správcem front produktu IBM MQ a je povoleno obnovení tajného klíče i prezenční signály kanálu, dojde k novému domlouvání okamžitě po každém prezenčním signálu kanálu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSRC s voláním MQINQ.

IBM i *Událost StartStop(10ciferné celé číslo se znaménkem) v IBM i*

Řídí, zda jsou generovány události spuštění a zastavení.

Tento atribut může mít některou z následujících hodnot:

EVRDIS
Vytváření sestav událostí je zakázáno.

EVRENA
Vytváření sestav událostí je povoleno.

Další informace o událostech naleznete v tématu Monitorování událostí.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASSE s voláním MQINQ.

IBM i *SyncPoint (desetimístné podepsané celé číslo) v IBM i*

Dostupnost synchronizačního bodu.

To označuje, zda lokální správce front podporuje jednotky práce a syncpointing s voláními MQGET, MQPUT a MQPUT1 .

SPAVR
Jednotky práce a syncpointing jsou k dispozici.

SPNAVL
Jednotky práce a syncpointing nejsou k dispozici.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IASYNC s voláním MQINQ.

IBM i *TraceRouteZáznam (10-číslicové celé číslo se znaménkem) v IBM i*

Tento ovládací prvek určuje, zda mají být informace o zprávách zaznamenávány při jejich toku prostřednictvím správce front.

Hodnota je jedna z následujících možností:

- RECDD: Není povoleno žádné připojení ke zprávám přenosové cesty trasování
- RECDQ: zprávy jsou vloženy do pevné pojmenované fronty
- RECDM: určit pomocí zprávy (toto je výchozí výchozí nastavení)

Chcete-li zabránit tomu, aby zpráva přenosové cesty trasování zůstala v systému, nastavte hodnotu vypršení platnosti na ní větší než nula a uveďte volbu sestavy RODISC. Chcete-li zabránit tomu, aby zprávy nebo zprávy odpovědí zůstaly v systému, nastavte volbu sestavy ROPDAE. Další informace viz “Volby sestav a příznaky zpráv v systému IBM i” na stránce 1418.

Chcete-li určit hodnotu tohoto atributu, použijte selektor IATRGI s voláním MQINQ.

IBM i *TreeLifeČas (10-číslicové celé číslo se znaménkem) v IBM i*

Doba životnosti, v sekundách, neadministrativních témat.

Neadministrativní témata jsou ta, která jsou vytvářena, 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.

Uveďte 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 (10ciferné celé číslo se znaménkem) v IBM i*

Interval zprávy spouštěče.

Jedná se o časový interval (v milisekundách), který se používá k omezení počtu zpráv spouštěče. To je relevantní pouze v případě, že *TriggerType* je TTFRST. V takovém případě se zprávy spouštěče obvykle generují pouze tehdy, když do fronty dorazí vhodná zpráva a fronta byla dříve prázdná. Za určitých okolností však může být generována další zpráva spouštěče s aktivací TTFRST, i když nebyla fronta prázdná. Tyto další zprávy triggeru se negenerují častěji než každých *TriggerInterval* milisekund.

Další informace o spouštění najdete v tématu Spouštění kanálů.

Hodnota je v rozsahu 0 až 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 vzorové programy dodávané s produktem IBM MQ for IBM i for RPG. Také se dozvíte, jak vytvářet spustitelné aplikace z programů, které napíšete.

Vytváření vaší aplikace

Publikace IBM i popisují, jak sestavit spustitelné aplikace z programů, které napíšete. Toto téma popisuje další úlohy a změny standardních úloh, které musíte provést při sestavování aplikací produktu IBM MQ for IBM i , které mají být spuštěny v produktu IBM i.

Kromě kódování volání MQI ve vašem zdrojovém kódu musíte přidat příslušné jazykové příkazy, které budou obsahovat soubory kopií produktu IBM MQ for IBM i pro jazyk RPG. Měli byste se seznámit s obsahem těchto souborů; jejich názvy 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 kopírovací soubory, které vám pomohou s psaními aplikací v programovacím jazyce RPG. Jsou vhodné pro použití s produktem WebSphere Development toolset (5722 WDS) ILE RPG 4 Compiler.

Soubory kopií, které produkt IBM MQ for IBM i poskytuje jako pomůcka při psaní uživatelských procedur kanálů, jsou popsány v tématu Programy výstupních bodů kanálů pro kanály systému zpráv.

Názvy kopírovaných souborů IBM MQ for IBM i pro RPG mají předponu CMQ. Mají příponu G nebo H. Existují oddělené soubory kopií obsahující pojmenované konstanty a jeden soubor pro každou ze struktur. Kopírovaná soubory jsou uvedena v seznamu “Jazykové aspekty” na stránce 1000.

Poznámka: Pro ILE RPG/400jsou dodávány jako členy souboruQRPGLESRC v knihovně QMQM.

Deklarace struktury neobsahují příkazy DS . To umožňuje aplikaci deklarovat datovou strukturu (nebo strukturu dat s více výskyty) zakódováním příkazu DS a použitím příkazu /COPY na kopírování 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 vašeho objektu typu *MODULE je třeba v příkazu CRTPGM uvést BNDSRVPGM(QMQM/LIBMQM) . To zahrnuje různé procedury IBM MQ ve vašem programu.

Při kompilování se ujistěte, že je knihovna obsahující soubory kopií (QMQM) v seznamu knihoven.

Další informace týkající se aspektů programování včetně klientských režimů najdete v tématu “Jazykové aspekty” na stránce 1000.

Rozhraní pro externí správce synchronizačního bodu produktu IBM i

Produkt IBM MQ for IBM i používá nativní vázané zpracování obslužného programu IBM i jako externí koordinátor synchronizačního bodu.

Další informace o možnostech vázaného zpracování produktu IBM inaleznete v příručce *IBM i Programming: Backup and Recovery Guide* .

Chcete-li spustit zařízení pro vázané zpracování IBM i , použijte systémový příkaz STRCMTCTL. K ukončení vázaného zpracování použijte příkaz systému ENDCMTCTL.

Poznámka: Výchozí hodnota *Rozsah definice vázaného zpracování* je *ACTGRP. Toto musí být definováno jako *JOB pro IBM MQ pro IBM i. Příklad:

```
STRCMTCTL LCKLVL(*ALL) CMTSCOPE(*JOB)
```

Pokud voláte příkaz MQPUT, MQPUT1nebo MQGET, uvedete PMSYP nebo GMSYP po spuštění vázaného zpracování, produkt IBM MQ for IBM i se přidá jako prostředek vázaného zpracování API do definice vázaného zpracování. Obvykle se jedná o první takové volání v úloze. Zatímco v určité definici vázaného zpracování jsou registrovány prostředky vázaného zpracování API, nemůžete pro tuto definici ukončit vázané zpracování.

Produkt IBM MQ for IBM i při odpojování od správce front odebere jeho registraci jako prostředek závazku rozhraní API, pokud v aktuální jednotce práce nejsou žádné nevyřízené operace MQI.

Pokud se odpojíte od správce front, dokud nejsou v aktuální pracovní jednotce provedeny nevyřízené operace MQPUT, MQPUT1nebo MQGET, produkt IBM MQ for IBM i zůstane registrován jako prostředek vázaného zpracování API, takže bude upozorněn na další potvrzení nebo odvolání. Když je dosaženo dalšího synchronizačního bodu, IBM MQ potvrdí nebo odvolá změny podle potřeby. Aplikaci je možné odpojit a znovu se připojit ke správci front během aktivní transakce a provádět další operace MQGET a MQPUT uvnitř stejné jednotky práce (jedná se o nevyřízený odpojení).

Pokusíte-li se vydat systémový příkaz ENDCMTCTL pro tuto definici vázaného zpracování, zobrazí se zpráva CPF8355 označující, že nevyřízené změny byly aktivní. Tato zpráva se také objeví v protokolu úlohy při ukončení úlohy. Chcete-li tomu zabránit, ujistěte se, že jste potvrdili nebo odvolali všechny nevyřízené operace IBM MQ a že jste se odpojili od správce front. Proto použití příkazů COMMIT nebo ROLLBACK před ENDCMTCTL by mělo umožnit úspěšné dokončení zpracování konce vázaného zpracování.

Když je vázané zpracování IBM i použito jako externí koordinátor synchronizačního bodu, volání MQCMIT, MQBACK a MQBEGIN nemusí být vydána. Volání těchto funkcí selže s kódem příčiny RC2012.

Chcete-li potvrdit nebo odvolat (to znamená odvolat) vaši jednotku práce, použijte jeden z programovacích jazyků, které podporují vázané zpracování. Příklad:

- Příkazy CL: COMMIT a ROLLBACK
- Programovací funkce ILE C: _Rcommit a _Rrollback
- RPG/400: COMMIT a ROLBK
- COBOL/400: COMMIT a ROLLBACK

Synchronizační body v produktu CICS pro aplikace produktu IBM i

IBM MQ for IBM i se podílí na jednotkách práce s CICS. Rozhraní MQI lze použít v rámci aplikace produktu CICS k vložení a získání zpráv uvnitř aktuální jednotky práce.

Příkaz EXEC CICS SYNCPOINT můžete použít k vytvoření synchronizačního bodu, který zahrnuje operace produktu IBM MQ for IBM i . Chcete-li odvolat všechny změny až na předchozí synchronizační bod, můžete použít příkaz EXEC CICS SYNCPOINT ROLLBACK.

Použijete-li příkaz MQPUT, MQPUT1nebo MQGET s PMSYP nebo GMSYP, který je nastaven v aplikaci CICS , nemůžete se odhlásit CICS , dokud produkt IBM MQ for IBM i neodebere svou registraci jako prostředek závazku rozhraní API. Proto byste měli před odpojením od správce front potvrdit nebo zazálohovat všechny nevyřízené operace vložení nebo získání. To vám umožní odhlásit se CICS.

Ukázkové programy v systému IBM i

Toto téma popisuje vzorové programy dodávané s produktem IBM MQ for IBM i for RPG. Ukázky demonstrují typická použití rozhraní MQI (Message Queue Interface).

Ukázky nejsou určeny k demonstraci obecných programovacích technik, takže byla vynechána některá kontrola chyb, kterou byste mohli chtít zahrnout do produkčního programu. Tyto ukázky jsou však vhodné pro použití jako základ pro vlastní programy front zpráv.

Zdrojový kód pro všechny ukázky je dodáván spolu s produktem; tento zdroj zahrnuje komentáře, které vysvětlují techniky front zpráv demonstrovány v programech.

K dispozici je jedna sada ukázkových programů ILE:

1. **Programy používající prototypová volání do MQI (statické vázané volání)**

   Zdroj existuje v QMQMSAMP/QRPGLESRC. Členy jsou pojmenovány AMQ3xxx4, kde xxx označuje ukázkovou funkci. Kopírování členů existuje v QMQM/QRPGLESRC. Každý název členu má příponu G nebo H.

Produkt Tabulka 812 na stránce 1400 poskytuje úplný seznam ukázkových programů dodávaných s produktem IBM MQ for IBM ia uvádí názvy programů v každém z podporovaných programovacích jazyků. Všimněte si, že názvy všech jejich názvů začínají předponou AMQ, čtvrtý znak v názvu označuje programovací jazyk.

Tabulka 812. Názvy ukázkových programů

	RPG (ILE)
Vložit ukázky	AMQ3PUT4
Procházet ukázky	AMQ3GBR4
Získat ukázky	AMQ3GET4
Ukázky požadavků	AMQ3REQ4
Ukázky Echo	AMQ3ECH4

Tabulka 812. Názvy ukázkových programů (pokračování)	
	RPG (ILE)
Zjišťovat ukázky	AMQ3INQ4
Nastavit ukázky	AMQ3SET4
Ukázka monitoru spouštěčů	AMQ3TRG4
Ukázka spouštěcího serveru	AMQ3SRV4

Kromě těchto voleb ukázky produktu IBM MQ for IBM i obsahuje ukázkový datový soubor AMQSDATA, který lze použít jako vstup pro některé ukázkové programy a ukázkové programy CL, které demonstrují administrativní úlohy. Ukázky CL jsou popsány v části Administrace produktu IBM i . Vzorový CL program můžete použít k vytvoření front, které se mají použít s ukázkovými programy popsanými v tomto tématu.

Informace o tom, jak spustit ukázkové programy, najdete v tématu “Příprava a spuštění ukázkových programů v systému IBM i” na stránce 1402.

Funkce demonstrovány v ukázkových programech v systému IBM i

Tabulka, která zobrazuje techniky demonstrováno ukázkovými programy produktu 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 813. Ukázkové programy demonstrující použití rozhraní MQI	
Technika	**RPG (ILE)**
Použití volání MQCONN a MQDISC	AMQ3ECH4 nebo AMQ3INQ4
Implicitní připojování a odpojování	AMQ3PUT4
Vložení 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 (čekání 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
Vyžádání zpráv výjimek	AMQ3REQ4
Přijímání oříznuté zprávy	AMQ3GBR4
Použití vyřešeného názvu fronty	AMQ3GBR4

Tabulka 813. Ukázkové programy demonstrující použití rozhraní MQI (pokračování)	
Technika	**RPG (ILE)**
Zpracování spouštěče	AMQ3SRV4 nebo AMQ3TRG4

Poznámka: Všechny vzorové programy vytvářejí soubor pro souběžný tisk, který obsahuje výsledky zpracování.

Příprava a spuštění ukázkových programů v systému IBM i

Než budete moci spustit ukázkové programy produktu IBM MQ for IBM i , musíte je zkompilovat stejně jako všechny ostatní aplikace produktu IBM MQ for IBM i . Chcete-li tak učinit, můžete použít příkazy IBM i CRTRPGMOD a CRTPGM.

Když vytváříte programy AMQ3xxx4 , musíte zadat BNDSRVPGM (QMQM/LIBMQM) v příkazu CRTPGM. Tento krok zahrnuje různé procedury produktu IBM MQ ve vašem programu.

Vzorové programy jsou poskytovány v knihovně QMQMSAMP jako členy QRPGLESRC. Používají kopírovací soubory poskytnuté v knihovně QMQM, takže se ujistěte, že je tato knihovna v seznamu knihoven, když je kompilujete. Kompilátor jazyka RPG poskytuje informační zprávy, protože ukázky nevyužívají mnoho proměnných, které jsou deklarovány v souborech kopií.

Spuštění ukázkových programů

Při spouštění ukázek můžete použít vlastní fronty nebo můžete kompilovat a spustit příkaz AMQSAMP4 za účelem vytvoření některých ukázkových front. Zdroj pro tento program je dodáván v souboru QCLSRC v knihovně QMQMSAMP. Lze jej kompilovat 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 být 48 znaků dlouhé, čehož dosáhnete vyplněním Queue_Name a Queue_Manager_Name s požadovaným počtem mezer.

V případě ukázkových programů produktu Inquire and Set jsou ukázkové definice vytvořené pomocí příkazu AMQSAMP4 vyvolány spuštěním těchto ukázek jazyka C. Chcete-li spouštět verze RPG, musíte změnit definice procesu SYSTEM.SAMPLE.ECHOPROCESS a SYSTEM.SAMPLE.INQPROCESS a SYSTEM.SAMPLE.SETPROCESS. Můžete použít příkaz CHGMQMPRC (popsaný v tématu Změna procesu MQ (CHGMQMPRC)) chcete-li to provést, nebo upravte a spusťte příkaz AMQSAMP4 s alternativní definicí.

Ukázkový program vložení v systému IBM i

Ukázkový program Put AMQ3PUT4umisťuje zprávy do fronty pomocí volání MQPUT.

Chcete-li spustit program, zavolejte program a zadejte jméno cílové fronty jako parametr programu. Program umístí sadu pevných zpráv do fronty; tyto zprávy jsou převzaty z datového bloku na konci zdrojového kódu programu. Ukázkový program put je AMQ3PUT4 v knihovně QMQMSAMP.

Pomocí tohoto vzorového programu je příkaz:

```
CALL PGM(QMQMSAMP/AMQ3PUT4) PARM('Queue_Name','Queue_Manager_Name')
```

Kde Queue_Name a Queue_Manager_Name musí být 48 znaků dlouhé, čehož dosáhnete vyplněním Queue_Name a Queue_Manager_Name s požadovaným počtem mezer.

Návrh ukázkového programu Put

Program používá volání MQOPEN s volbou OOOUT k otevření cílové fronty pro vkládání zpráv. Výsledky jsou výstupem do souboru pro souběžný tisk. Pokud tuto frontu nemůže otevřít, program zapíše chybovou

zprávu obsahující kód příčiny vrácený voláním MQOPEN. Chcete-li program ponechat jednoduchý, a to i při následných voláních MQI, program použije výchozí hodnoty pro celou řadu voleb.

Pro každý řádek dat obsažených ve zdrojovém kódu program přečte text do vyrovnávací paměti a použije volání MQPUT k vytvoření datagramové zprávy obsahující text této řádky. Program pokračuje, dokud nedojde k dosažení konce vstupu nebo volání MQPUT selže. Pokud se program dostane na konec vstupu, zavře frontu pomocí volání MQCLOSE.

Ukázkový program Procházet v systému IBM i

Ukázkový program Procházet AMQ3GBR4prochází zprávy ve frontě pomocí volání MQGET.

Program načte kopie všech zpráv ve frontě, kterou uvedete při volání programu; zprávy zůstanou ve frontě. Je možné použít dodanou frontu SYSTEM.SAMPLE.LOCAL; nejprve spusťte vzorový program, abyste vložili nějaké zprávy do fronty. Můžete použít frontu SYSTEM.SAMPLE.ALIAS, což je název aliasu pro stejnou lokální frontu. Program pokračuje, dokud nedosáhne konce fronty, nebo se nezdaří volání MQI.

Příklad příkazu k volání programu RPG je:

```
CALL PGM(QMQMSAMP/AMQ3GBR4) PARM('Queue_Name','Queue_Manager_Name')
```

Kde `Queue_Name` a `Queue_Manager_Name` musí být 48 znaků dlouhé, čehož dosáhnete vyplněním `Queue_Name` a `Queue_Manager_Name` s požadovaným počtem mezer. Proto, pokud používáte SYSTEM.SAMPLE.LOCAL jako cílová fronta, budete potřebovat 29 prázdných znaků.

Návrh ukázkového programu pro procházení

Program otevře cílovou frontu pomocí volání MQOPEN s volbou OOBRW. Pokud nemůže frontu otevřít, program zapíše chybovou zprávu do svého souboru pro souběžný tisk, který obsahuje kód příčiny vrácený voláním MQOPEN.

Pro každou zprávu ve frontě používá program 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 zprávy *první* při prvním spuštění volání.

GMNWT.
Program nečeká, pokud ve frontě nejsou žádné zprávy.

GMATM
Volání MQGET určuje vyrovnávací paměť pevné velikosti. Je-li zpráva delší než tato vyrovnávací paměť, program zobrazí oříznutou zprávu spolu s varováním, že zpráva byla zkrácena.

Tento program demonstruje, jak musíte vymazat pole *MDMID* a *MDCID* struktury MQMD po každém volání MQGET, protože volání nastavuje tato pole na hodnoty obsažené ve zprávě, kterou načítá. Vymazání těchto polí znamená, že po sobě jdoucí příkazy MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

Program pokračuje do konce fronty; zde volání MQGET vrátí kód příčiny RC2033 (není k dispozici žádná zpráva) a program zobrazí varovnou zprávu. Pokud se volání MQGET nezdaří, program zapíše chybovou zprávu, která obsahuje kód příčiny v jeho souboru pro souběžný tisk.

Program poté zavře frontu pomocí volání MQCLOSE.

Ukázkový program Get na systému IBM i

Ukázkový program Get, AMQ3GET4, získá zprávy z fronty pomocí volání MQGET.

Když je program volán, odstraní zprávy z uvedené fronty. Je možné použít dodanou frontu SYSTEM.SAMPLE.LOCAL; nejprve spusťte vzorový program, abyste vložili nějaké zprávy do fronty. Můžete použít SYSTEM.SAMPLE.ALIAS fronta, což je název aliasu pro stejnou lokální frontu. Program pokračuje do doby, než je fronta prázdná, nebo selže volání MQI.

Příklad příkazu k volání programu RPG je:

```
CALL PGM(QMQMSAMP/AMQ3GET4) PARM('Queue_Name','Queue_Manager_Name')
```

kde `Queue_Name` a `Queue_Manager_Name` musí být 48 znaků dlouhé, čehož dosáhnete vyplněním `Queue_Name` a `Queue_Manager_Name` s 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 získání zpráv; používá volání MQOPEN s volbou OOINPQ. Pokud nemůže frontu otevřít, program zapíše chybovou zprávu obsahující kód příčiny vrácený voláním MQOPEN ve svém souboru pro souběžný tisk.

Pro každou zprávu ve frontě program používá volání MQGET k odebrání zprávy z fronty. Poté se zobrazí data obsažená ve zprávě. Volání MQGET používá volbu GMWT uvádějící interval čekání (*GMWI*) o 15 sekundách, takže program čeká na toto období, pokud ve frontě není žádná zpráva. Pokud žádná zpráva nepřijde před uplynutím tohoto intervalu, volání selže a vrátí kód příčiny RC2033 (není k dispozici žádná zpráva).

Tento program demonstruje, jak musíte vymazat pole *MDMID* a *MDCID* struktury MQMD po každém volání MQGET, protože volání nastavuje tato pole na hodnoty obsažené ve zprávě, kterou načítá. Vymazání těchto polí znamená, že po sobě jdoucí příkazy MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

Volání MQGET určuje vyrovnávací paměť pevné velikosti. Je-li 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 žádný zpráva), nebo selže volání MQGET. Pokud se volání nezdaří, zobrazí program chybovou zprávu, která obsahuje kód příčiny.

Program poté zavře frontu pomocí volání MQCLOSE.

Ukázkový program požadavku v systému IBM i

Ukázkový program Požadavek AMQ3REQ4demonstruje zpracování klienta/serveru. Ukázka je klient, který ukládá zprávy požadavků do fronty, která je zpracována programem serveru. Čeká na to, aby serverový program vložil zprávu odpovědi do fronty pro odpověď.

Ukázka požadavku ukládá posloupnosti zpráv požadavků ve frontě pomocí volání MQPUT. Tyto zprávy uvádějí SYSTEM.SAMPLE.REPLY jako fronta odpovědí. Program čeká na zprávy odpovědi a pak je zobrazí. Odpovědi se odesílají pouze v případě, že cílová fronta (která bude nazývat *fronta serveru*). je zpracováván serverovou aplikací nebo je-li pro tento účel spuštěna aplikace (jsou navrženy ukázkové programy Inquire and Set), které mají být spuštěny. Ukázka čeká 5 minut na příchod první odpovědi (k povolení času pro aplikaci serveru) a 15 sekund pro následné odpovědi, ale to může skončit bez získání odpovědí.

Chcete-li spustit program, zavolejte program a zadejte jméno cílové fronty jako parametr programu. Program umístí sadu pevných zpráv do fronty; tyto zprávy jsou převzaty z datového bloku na konci zdrojového kódu programu.

Návrh ukázkového programu Požadavek

Program otevře frontu serveru tak, aby mohla vkládat zprávy. Vyvolá volání MQOPEN s volbou OOOUT. Nemůže-li frontu otevřít, zobrazí se v programu chybová zpráva obsahující kód příčiny vrácený voláním MQOPEN.

Program pak otevře frontu pro odpověď s názvem SYSTEM.SAMPLE.REPLY , aby bylo možné získat zprávy odpovědí. Za tímto způsobem program používá volání MQOPEN s volbou OOINPX. Nemůže-li frontu otevřít, zobrazí se v programu chybová zpráva obsahující kód příčiny vrácený voláním MQOPEN.

Pro každý řádek vstupu program pak přečte text do vyrovnávací paměti a použije volání MQPUT k vytvoření zprávy požadavku obsahující text této řádky. Na tomto volání program používá volbu sestavy

ROEXCD k požadavku, aby všechny zprávy sestavy odeslané o zprávě požadavku obsahovaly prvních 100 bajtů dat zprávy. Program pokračuje, dokud nedojde k dosažení konce vstupu nebo volání MQPUT selže.

Program pak použije volání MQGET k odstranění zpráv odpovědí z fronty a zobrazí data obsažená v odpovědích. Volání MQGET používá volbu GMWT, která uvádí čekací interval (*GMWI*) 5 minut pro první odpověď (což umožní spuštění serverové aplikace) a 15 sekund pro následné odpovědi. Program čeká na tato období, pokud ve frontě není žádná zpráva. Pokud žádná zpráva nepřijde před uplynutím tohoto intervalu, volání selže a vrátí kód příčiny RC2033 (není k dispozici žádná zpráva). Volání také používá volbu GMATM, takže zprávy delší než deklarovaná velikost vyrovnávací paměti jsou oříznuty.

Tento program demonstruje, jak musíte vymazat pole *MDMID* a *MDCOD* struktury MQMD po každém volání MQGET, protože volání nastavuje tato pole na hodnoty obsažené ve zprávě, kterou načítá. Vymazání těchto polí znamená, že po sobě jdoucí příkazy MQGET načítají zprávy v pořadí, ve kterém jsou zprávy zadrženy ve frontě.

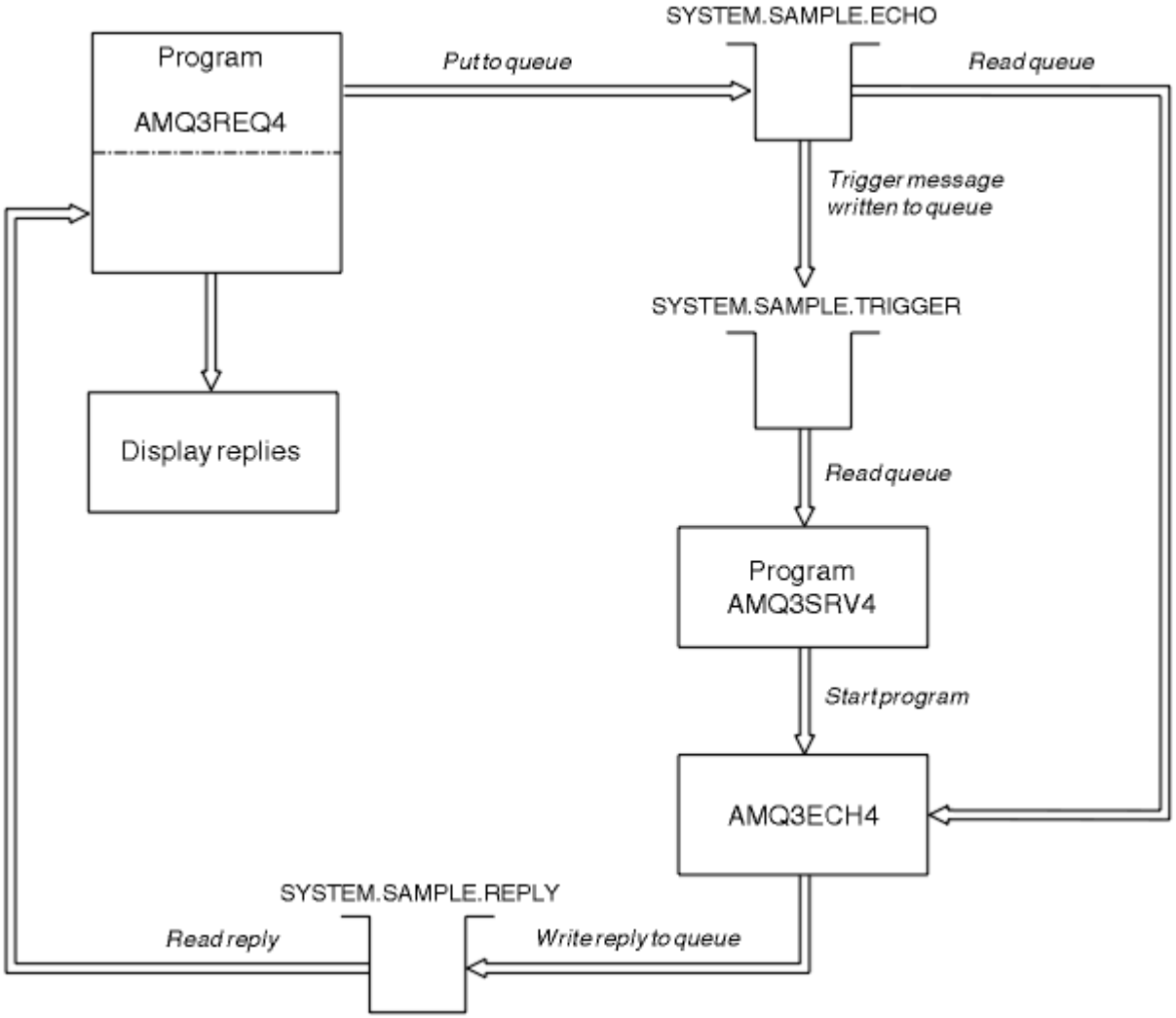
Program pokračuje, dokud buď volání MQGET nevrátí kód příčiny RC2033 (není k dispozici žádný zpráva), nebo selže volání MQGET. Pokud se volání nezdaří, zobrazí program chybovou zprávu, která obsahuje kód příčiny.

Program pak zavře jak frontu serveru, tak frontu pro odpověď pomocí volání MQCLOSE. Produkt Tabulka 814 na stránce 1405 zobrazuje změny ukázkového programu Echo, které jsou nezbytné ke spuštění ukázkových programů pro zjišťování a nastavení.

Poznámka: Podrobnosti o ukázkovém programu Echo jsou zahrnuty jako reference.

Tabulka 814. Podrobnosti ukázkového programu klienta/serveru

Název programu	**Fronta SYSTEM/SAMPLE**	**Program spuštěn**
Ozvěna	OZVĚNA	AMQ3ECH4
Zjišťovat	inq	AMQ3INQ4
Nastavit	SET	AMQ3SET4



Obrázek 9. Ukázkový diagram vývojového diagramu Client/Server (Echo)

IBM i *Použití spouštěče s ukázkou požadavku v systému IBM i*

Chcete-li spustit ukázku pomocí spouštěče, spusťte program spouštěcího serveru AMQ3SRV4proti požadované inicializační frontě v jedné úloze, poté spusťte příkaz AMQ3REQ4 v jiné úloze.

To znamená, že spouštěcí server je připraven, když vzorový program Požadavek 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 svou vlastní inicializační frontu.
2. Ukázkové definice vytvořené pomocí příkazu AMQSAMP4 způsobí, že se spustí verze jazyka C ukázky. Chcete-li spustit verzi RPG, musíte změnit definice procesu SYSTEM.SAMPLE.ECHOPROCESS a SYSTEM.SAMPLE.INQPROCESS a SYSTEM.SAMPLE.SETPROCESS. Můžete použít příkaz CHGMQMPRC (viz Změna procesu MQ (CHGMQMPRC) pro další podrobnosti), abyste to mohli provést, nebo můžete upravit a spustit vlastní verzi AMQSAMP4.
3. Program spouštěcího serveru musíte zkompilovat ze zdroje poskytnutého v QMQMSAMP/QRPGLESRC.

V závislosti na spouštěcím procesu, který chcete spustit, by měl být příkaz AMQ3REQ4 volán s parametrem určujícím zprávy požadavku, které mají být umístěny na jednu z těchto ukázkových front serveru:

- SYSTEM.SAMPLE.ECHO (pro ukázkové programy Echo)

- SYSTEM.SAMPLE.INQ (pro ukázkové programy pro zjišťování)
- SYSTEM.SAMPLE.SET (pro sadu ukázkových programů)

Graf toku pro SYSTEM.SAMPLE.ECHO je zobrazen v souboru Obrázek 9 na stránce 1406. Pomocí tohoto příkladu zadáte příkaz k vydání požadavku na program RPG na tento server:

```
CALL PGM(QMQMSAMP/AMQ3REQ4) PARM('SYSTEM.SAMPLE.ECHO
+ 30 blank characters','Queue_Manager_Name')
```

protože název fronty a správce front musí mít délku 48 znaků.

Poznámka: Tato ukázková fronta má typ spouštěče FIRST, takže pokud již ve frontě existují zprávy, než spustíte ukázku Požadavek, serverové aplikace se nespustí pomocí zpráv, které odešlete.

Pokud se chcete pokusit o další příklady, můžete zkusit následující varianty:

- Use AMQ3TRG4 instead of AMQ3SRV4 to submit the job instead, but potential job submission delays could make it less easy to follow what is happening.
- Použijte adresu SYSTEM.SAMPLE.INQ a SYSTEM.SAMPLE.SET . Pomocí vzorového datového souboru jsou příkazy k 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 odeslaného do fronty odpovědí. Program se nazývá AMQ3ECH4 .

Chcete-li spustit spouštěcí proces, musíte se ujistit, že ukázkový program Echo, který chcete použít, je spuštěn zprávami přicházejícími do fronty SYSTEM.SAMPLE.ECHO. Chcete-li tak učinit, zadejte název ukázkového programu Echo, který chcete použít, v poli *ApplId* v definici procesu SYSTEM.SAMPLE.ECHOPROCESS. (Pro tento parametr můžete použít příkaz CHGMQMPRC, který je popsán v části Administrace produktu IBM i .) Ukázková fronta má typ spouštěče FIRST, takže pokud již ve frontě existují zprávy, než spustíte ukázku požadavku, ukázka Echo se nespustí pomocí zpráv, které odešlete.

Pokud jste definici správně nastavili, nejprve spusťte příkaz AMQ3SRV4 v jedné úloze a poté spusťte příkaz AMQ3REQ4 v jiném. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální zpoždění při odeslání úlohy by mohla méně snadno následovat, co se děje.

Ukázkové programy požadavku slouží k odesílání zpráv do fronty SYSTEM.SAMPLE.ECHO. Ukázkové programy Echo odešlou zprávu s odpovědí obsahující data ve zprávě požadavku do fronty odpovědi určené ve zprávě s požadavkem.

Návrh ukázkového programu Echo

Když se program spustí, explicitně se připojí k výchozímu správci front pomocí volání MQCONN. Ačkoli to v produktu IBM inení nezbytné, znamená to, že byste mohli používat stejný program na jiných platformách, aniž byste změnili zdrojový kód.

Poté program otevře frontu pojmenovanou ve struktuře zpráv spouštěče, která byla předána při spuštění. (Pro srozumitelnost zavoláme tuto *frontu požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odstranění zpráv z této fronty. Toto volání používá volby GMATM a GMWT, s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy za účelem zjištění, zda se jedná o zprávu požadavku; pokud není, program zahodí zprávu a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program používá volání MQPUT k vložení zprávy odpovědi do fronty pro odpovědi. Tato zpráva obsahuje obsah zprávy požadavku.

Pokud ve frontě požadavků nejsou 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 je IBM i, ačkoli pro tuto situaci není dodána žádná ukázka. Chcete-li provést práci programu ECHO, postupujte takto:

- Napište program, který správně uvádí pole *Format*, *Encoding*a *CCSID* k odeslání zpráv s požadavky na text.

  Program ECHO požaduje, aby správce front provedl převod dat zpráv, pokud je to potřeba.
- Uveďte CONVERT (*YES) na odesílajícím kanálu IBM MQ for IBM i , pokud program, který jste napsali, nezajišťuje podobnou konverzi pro odpověď.

Ukázkový program Inquire na systému IBM i

Ukázkový program Inquire, AMQ3INQ4, inquires about some of the attributes of a queue using the MQINQ call.

Program je určen ke spuštění jako spouštěný program, takže jeho jediným vstupem je struktura MQTMC (trigger message). Tato struktura obsahuje název cílové fronty s atributy, které mají být zjišťovány.

Aby spouštěcí proces fungoval, musíte se ujistit, že je ukázkový program pro zjišťování spuštěn zprávami, které přicházejí do fronty SYSTEM.SAMPLE.INQ. Chcete-li tak učinit, uveďte název ukázkového programu Inquire v poli *ApplId* SYSTEM.SAMPLE.INQPROCESS definice procesu. (Pro tento parametr můžete použít příkaz CHGMQMPRC, který je popsán v tématu Změna procesu MQ (CHGMQMPRC)). Ukázková fronta má typ spouštěče FIRST, takže pokud již ve frontě existují zprávy, než spustíte ukázku Požadavek, nevyvolá se ukázka Inquire zpráv, které odešlete.

Pokud jste definici správně nastavili, nejprve spusťte příkaz AMQ3SRV4 v jedné úloze a poté spusťte příkaz AMQ3REQ4 v jiném. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální zpoždění při odeslání úlohy může být méně jednoduchá na to, co se děje.

Ukázkový program Žádost použijte k odeslání zpráv požadavků, z nichž každý obsahuje pouze název fronty, do fronty SYSTEM.SAMPLE.INQ. Pro každou zprávu požadavku program Inquire sample odešle zprávu s odpovědí obsahující informace o frontě zadané ve zprávě požadavku. Odpovědi se posílají do fronty odpovědí uvedené ve zprávě s požadavkem.

Návrh ukázkového programu Inquire

Když se program spustí, explicitně se připojí k výchozímu správci front pomocí volání MQCONN. Tato funkce návrhu sice není v produktu IBM inezbytná, ale znamená to, že stejný program můžete používat i na jiných platformách, aniž byste změnili zdrojový kód.

Poté program otevře frontu pojmenovanou ve struktuře zpráv spouštěče, která byla předána při spuštění. (Pro srozumitelnost zavoláme tuto *frontu požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odstranění zpráv z této fronty. Toto volání používá volby GMATM a GMWT, s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy za účelem zjištění, zda se jedná o zprávu požadavku; pokud není, program zahodí zprávu a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program přečte název fronty (která bude nazývat *cílovou frontou*). obsažené v datech a otevření této fronty pomocí volání MQOPEN s volbou OOINQ. Program potom použije volání MQINQ k dotazům na hodnoty atributů **InhibitGet**, **CurrentQDepth**a **OpenInputCount** cílové fronty.

Je-li volání MQINQ úspěšné, program použije volání MQPUT k vložení zprávy odpovědi do fronty pro odpovědi. Tato zpráva obsahuje hodnoty tří atributů.

Je-li volání MQOPEN nebo MQINQ neúspěšné, program použije volání MQPUT k vložení zprávy *report* do fronty pro odpovědi. V poli *MDFB* v deskriptoru zprávy této zprávy je kód příčiny vrácený voláním MQOPEN nebo MQINQ, v závislosti na tom, který z nich selhal.

Po volání MQINQ program zavře cílovou frontu pomocí volání MQCLOSE.

Pokud ve frontě požadavků nejsou žádné zprávy, program tuto frontu zavře a odpojí se od správce front.

Ukázkový program Nastavit v systému IBM i

Ukázkový program Set AMQ3SET4zabraňuje operacím vložení ve frontě pomocí volání MQSET za účelem změny atributu **InhibitPut** fronty.

Program je určen ke spuštění jako spouštěný program, takže jeho jediným vstupem je struktura MQTMC (spouštěcí zpráva), která obsahuje název cílové fronty s atributy, které se mají dotazovat.

Aby spouštěcí proces fungoval, musíte se ujistit, že ukázkový program pro nastavení spouští zprávy přicházející do fronty SYSTEM.SAMPLE.SET. Chcete-li to provést, zadejte název ukázkového programu v poli *ApplId* definice procesu SYSTEM.SAMPLE.SETPROCESS. (Pro tento parametr můžete použít příkaz CHGMQMPRC, který je popsán v publikaci Administrace IBM i .) Ukázková fronta má typ spouštěče FIRST, takže pokud již ve frontě existují zprávy, než spustíte ukázku Požadavek, nespustí se ukázka Nastavit zprávy, které odešlete.

Pokud jste definici správně nastavili, nejprve spusťte příkaz AMQ3SRV4 v jedné úloze a poté spusťte příkaz AMQ3REQ4 v jiném. Můžete použít AMQ3TRG4 místo AMQ3SRV4, ale potenciální zpoždění při odeslání úlohy by mohla méně snadno následovat, co se děje.

Ukázkový program Žádost použijte k odeslání zpráv požadavků, z nichž každá obsahuje pouze název fronty, do fronty SYSTEM.SAMPLE.SET. Pro každou zprávu požadavku program Set sample odešle zprávu odpovědi obsahující potvrzení, že operace put byly na zadané frontě zablokovány. Odpovědi se posílají do fronty odpovědí uvedené ve zprávě s požadavkem.

Návrh ukázkového programu Set

Když se program spustí, explicitně se připojí k výchozímu správci front pomocí volání MQCONN. I když to není nezbytné pro IBM i, znamená to, že byste mohli používat stejný program na jiných platformách, aniž byste změnili zdrojový kód.

Poté program otevře frontu pojmenovanou ve struktuře zpráv spouštěče, která byla předána při spuštění. (Pro srozumitelnost zavoláme tuto *frontu požadavků*.) Program používá volání MQOPEN k otevření této fronty pro sdílený vstup.

Program používá volání MQGET k odstranění zpráv z této fronty. Toto volání používá volby GMATM a GMWT, s intervalem čekání 5 sekund. Program testuje deskriptor každé zprávy za účelem zjištění, zda se jedná o zprávu požadavku; pokud není, program zahodí zprávu a zobrazí varovnou zprávu.

Pro každou zprávu požadavku odebranou z fronty požadavků program přečte název fronty (která bude nazývat *cílovou frontou*). obsažené v datech a otevření této fronty pomocí volání MQOPEN s volbou OOSET. Program potom použije volání MQSET k nastavení hodnoty atributu **InhibitPut** cílové fronty na QAPUTI.

Je-li volání MQSET úspěšné, program použije volání MQPUT k vložení zprávy odpovědi do fronty pro odpovědi. Tato zpráva obsahuje řetězec `PUT inhibited`.

Je-li volání MQOPEN nebo MQSET neúspěšné, program použije volání MQPUT k vložení zprávy *report* do fronty pro odpovědi. V poli *MDFB* v deskriptoru zprávy této zprávy je kód příčiny vrácený voláním MQOPEN nebo MQSET, v závislosti na tom, který z nich selhal.

Po volání MQSET program zavře cílovou frontu pomocí volání MQCLOSE.

Pokud ve frontě požadavků nejsou žádné zprávy, program tuto frontu zavře a odpojí se od správce front.

Spouštěcí ukázkové programy v systému IBM i

IBM MQ for IBM i dodává dva spouštěcí ukázkové programy, které jsou napsány v ILE/RPG.

Programy jsou:

AMQ3TRG4
: Toto je monitor spouštěčů pro prostředí IBM i . Předává úlohu IBM i pro spuštění aplikace, ale to znamená, že jsou k dispozici dodatečné náklady na zpracování spojené s každou zprávou spouštěče.

AMQ3SRV4
Toto je spouštěcí server pro prostředí IBM i . Pro každou zprávu spouštěče spouští tento server ve své vlastní úloze spouštěcí příkaz ke spuštění zadané aplikace. Spouštěcí server může volat transakce CICS .

Verze jazyka C těchto ukázek jsou také k dispozici jako spustitelné programy v knihovně QMQM, nazývané AMQSTRG4 a AMQSERV4.

Ukázkový monitor spouštěčů AMQ3TRG4 v systému IBM i
AMQ3TRG4 je monitor spouštěčů. Tento parametr má jeden parametr: název inicializační fronty, která má sloužit. AMQSAMP4 definuje ukázkovou inicializační frontu SYSTEM.SAMPLE.TRIGGER, který můžete použít, když zkoušíte ukázkové programy.

Příkaz AMQ3TRG4 odešle úlohu IBM i pro každou platnou zprávu spouštěče, kterou obdrží od inicializační fronty.

Návrh monitoru spouštěčů

Monitor spouštěčů otevře inicializační frontu a získává zprávy z fronty a určuje neomezený interval čekání.

Monitor spouštěčů odešle úlohu IBM i ke spuštění aplikace zadané ve zprávě spouštěče a předává strukturu MQTMC (znaková verze zprávy spouštěcího impulsu). Data prostředí ve zprávě spouštěče se používají jako parametry odeslání úlohy.

Nakonec program zavře inicializační frontu.

Ukázkový server spouštěče AMQ3SRV4
AMQ3SRV4 je spouštěcí server. Tento parametr má jeden parametr: název inicializační fronty, která má sloužit. AMQSAMP4 definuje ukázkovou inicializační frontu SYSTEM.SAMPLE.TRIGGER, který můžete použít, když zkoušíte ukázkové programy.

Pro každou zprávu spouštěče příkaz AMQ3SRV4 spustí ve své vlastní úloze příkaz ke spuštění určené aplikace.

Pomocí vzorového spouštěcího impulsu, který má příkaz vydat, je:

```
CALL PGM(QMQM/AMQ3SRV4) PARM('Queue Name')
```

Kde `Queue Name` musí být 48 znaků dlouhé, což lze dosáhnout vyplněním názvu fronty s požadovaným počtem mezer. Proto, pokud používáte SYSTEM.SAMPLE.TRIGGER jako cílová fronta, budete potřebovat 28 prázdných znaků.

Návrh spouštěcího serveru

Návrh spouštěcího serveru je podobný jako u spouštěcího monitoru s výjimkou spouštěcího serveru:

- Umožňuje CICS stejně jako IBM i aplikací
- Nepoužije data prostředí ze zprávy spouštěče
- Volá aplikaci IBM i ve své vlastní úloze (nebo používá příkaz STRCICSUSR ke spuštění aplikací CICS) namísto zadávání úlohy IBM i .
- Otevře inicializační frontu pro sdílený vstup, takže mnoho serverů spouštěčů může být spuštěno ve stejnou dobu.

Poznámka: Programy, které spustil AMQ3SRV4 , nesmějí používat volání MQDISC, protože se tím zastaví spouštěcí server. Pokud programy spouštěné příkazem AMQ3SRV4 používají volání MQCONN, získají kód příčiny RC2002 .

Ukončení spouštění ukázkových programů v systému IBM i
Program monitoru spouštěčů může být ukončen volbou sysrequest 2 (ENDRQS) nebo tím, že se replikace získá ze spouštěcí fronty.

Je-li použita vzorová spouštěcí fronta, příkaz je:

```
CHGMQMQ QNAME('SYSTEM.SAMPLE.TRIGGER') GETENBL(*NO)
```

Poznámka: Chcete-li spustit spuštění znovu 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), která používá vzdáleného správce front s názvem 0THER. Chcete-li použít tuto vzorovou definici, změňte 0THER na název druhého správce front zpráv, který chcete použít. Musíte také nastavit kanál zpráv mezi dvěma správci front zpráv; chcete-li získat informace o tom, jak to provést, přečtěte si téma Kanály pro zasílání zpráv kanálů pro kanály systému zpráv.

Ukázkový program požadavku umístí své vlastní lokální jméno správce front do pole *MDRM* zpráv, které odesílá. Ukázky Inquire a Set odešle zprávy odpovědi do fronty a správce front zpráv pojmenované v polích *MDRQ* a *MDRM* v požadavcích na zprávy, které zpracovávají.

Návratové kódy pro IBM i (ILE RPG)

Tyto informace popisují návratové kódy přidružené k rozhraní 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 jazyka C++.

Pro každé volání se správce front nebo uživatelská procedura vrací kód dokončení a kód příčiny, který označuje úspěch nebo selhání volání.

Aplikace nesmí záviset na chybách, které jsou kontrolovány ve specifickém pořadí, kromě případů, kdy je to výslovně uvedeno. Pokud by z volání mohlo dojít k více než jednomu kódu dokončení nebo kódu příčiny, závisí konkrétní hlášená chyba na implementaci.

Kódy dokončení pro IBM i (ILE RPG)

Parametr kódu dokončení (*CMPCOD*) umožňuje volajícímu rychle zjistit, zda bylo volání úspěšně dokončeno, dokončeno částečně, nebo selhalo.

KEK
(MQCC_OK na jiných platformách)

Úspěšné dokončení.

Volání bylo dokončeno plně; všechny výstupní parametry byly nastaveny. Parametr **REASON** má v tomto případě vždy hodnotu RCNONE.

CCWARN
(MQCC_WARN na jiných platformách)

Varování (částečné dokončení).

Volání bylo dokončeno částečně. Některé výstupní parametry mohly být nastaveny spolu s výstupními parametry *CMPCOD* a *REASON* . Parametr **REASON** poskytuje další informace o částečném dokončení.

CCFIL
(MQCC_FAIL na jiných platformách)

Volání se nezdařilo.

Zpracování volání nebylo dokončeno a stav správce front je normálně nezměněn. Výjimky jsou výslovně zaznamenány. Výstupní parametry *CMPCOD* a *REASON* byly nastaveny; ostatní parametry jsou nezměněny, kromě případů, kdy byly zaznamenány.

Příčinou může být chyba v aplikačním programu, nebo může být výsledkem nějaké situace mimo program, například oprávnění uživatele mohlo být odvoláno. Parametr **REASON** udává další informace o chybě.

Kódy příčiny pro IBM i (ILE RPG)

Parametr kódu příčiny (*REASON*) je kvalifikace na parametr kódu dokončení (*CMPCOD*).

Není-li k dispozici žádná speciální příčina, vrátí se hodnota RCNONE. Úspěšné volání vrátí hodnotu CCOK a RCNONE.

Je-li kód dokončení buď CCCWARN nebo CCFAIL, správce front vždy nahlásí kvalifikovanou příčinu; podrobnosti jsou uvedeny pod každým popisem volání.

Pokud rutiny uživatelských procedur nastavují kódy dokončení a důvody, měly by se řídit těmito pravidly. Dále platí, že všechny speciální hodnoty důvodu definované uživatelskými procedurami by měly být menší než nula, aby se zajistilo, že se nekolidují s hodnotami nadefinovanými správcem front. Uživatelské procedury mohou nastavit příčiny, které jsou již definovány správcem front, kde jsou tyto důvody vhodné.

Kódy příčiny se také vyskytují v:

- Pole *DLREA* struktury MQDLH
- Pole *MDFB* struktury MQMD

Úplný seznam kódů příčiny najdete v tématu Kódy dokončení a příčin rozhraní API.

Chcete-li vyhledat kód příčiny IBM i v tomto seznamu, odeberte z přední části text "RC", například RC2002 se stane 2002. Kódy dokončení se také zobrazí, protože se nacházejí na jiných platformách:

Tabulka 815. Názvy kódů příčiny na serveru IBM i a na jiných platformách

IBM i	**Ostatní platformy**
KEK	MQCC_OK
CCWARN	MQCC_WARN
CCFIL	SELHÁNÍ MQCC_FAIL

Pravidla pro ověření platnosti voleb MQI pro produkt IBM i (ILE RPG)

Toto téma poskytuje informace o situacích, které produkují kód příčiny RC2046 z volání MQOPEN, MQPUT, MQPUT1, MQGET nebo MQCLOSE.

Volání MQOPEN v systému IBM i

Volby volání MQOPEN:

- Musí být zadán *Nejméně jedna* z následujících:
  - OOBRW
  - OOINPQ
  - OOINPX
  - OOINPS
  - OOINQ
  - OOUT

  - OSADA
- Povolen je pouze *jeden* z následujících možností:
  - OOINPQ
  - OOINPX
  - OOINPS
- Povolen je pouze *jeden* z následujících možností:
  - OOBNDO
  - OOBNDN
  - OOBNDQ

  Poznámka: Volby uvedené dříve se vzájemně vylučují. Avšak, protože hodnota OOBNDQ je nula a její zadání s některou z dalších dvou voleb vázání, nebude výsledkem kód příčiny RC2046. OOBNDQ je k dispozici pro dokumentaci programu podpory.
- Je-li uvedeno OOSAVA, musí být také uvedena jedna z voleb OOINP*.
- Je-li zadán jeden z voleb OOSET* nebo OOPAS*, musí být také uvedena volba OOOUT.

Volání MQPUT v systému IBM i

Pro volby put-message:

- Kombinace PMSYP a PMNSYP není povolena.
- Povolen je pouze *jeden* z následujících možností:
  - PMDEFC
  - PMNOC
  - PMPASA
  - PMPASI
  - PMSETA
  - PMSETI
- PMALTU není povolen (je platný pouze na volání MQPUT1).

Volání MQPUT1 v systému IBM i

U voleb vložení zpráv jsou pravidla stejná jako pro volání MQPUT, s výjimkou následujících voleb:

- PMALTU je povolen.
- PMLOGO není povoleno.

Volání MQGET v systému IBM i

Pro volby get-message:

- Povolen je pouze *jeden* z následujících voleb:
  - GMNSYP
  - GMSYP
  - GMPSYP
- Povolen je pouze *jeden* z následujících voleb:
  - GMBRWFCH.
  - GMBRWC
  - GMBRWN

  - GMMUC
- GMSYP není povolen s žádnou z následujících voleb:
  - GMBRWFCH.
  - GMBRWC
  - GMBRWN
  - GMLK
  - GMUNK
- Parametr GMPSYP není povolen s žádnou z následujících voleb:
  - GMBRWFCH.
  - GMBRWC
  - GMBRWN
  - GMCMPM
  - GMUNK
- Je-li uveden GMLK, musí být uvedena jedna z následujících voleb:
  - GMBRWFCH.
  - GMBRWC
  - GMBRWN
- Je-li uveden GMUNLK, jsou povoleny pouze následující volby:
  - GMNSYP
  - GMNWT.

Volání MQCLOSE v systému IBM i

- Volby pro volání MQCLOSE. Kombinace CODEL a COPURG není povolena.
- Je povolena pouze jedna z následujících možností:
  - COKPSSCOMMENT
  - CORMSB

Volání MQSUB na IBM i

Volby volání MQSUB:

- Musí být uvedena alespoň jedna z následujících možností:
- Musí být uvedena alespoň jedna z následujících možností:
  - SOALT
  - TORIE
  - PONOŽKA
- Je povolena pouze jedna z následujících možností:
  - SODUR
  - SONDURŠTINA

  Poznámka: Volby uvedené dříve se vzájemně vylučují. Protože však hodnota parametru SOnDUR je nula, zadání hodnoty SODUR nebude mít za následek kód příčiny RC2046. SONDUR je k dispozici pro dokumentaci programu podpory.
- Kombinace SOGRP a SOMAN není povolena.
- SOGRP vyžaduje uvedení SOSCID.
- Je povolena pouze jedna z následujících možností: SOAUID SOFUID

- Kombinace SONEWP a SOPOUBR není povolena.
- SONEWP je povolen pouze v kombinaci se SOCRT.
- Je povolena pouze jedna z následujících možností:
  - SWCHRŠTINA
  - SNOWTOP

Kódování počítače v systému IBM i

Tyto informace použijte, chcete-li se dozvědět více o struktuře pole *MDENC* v deskriptoru zpráv.

Další informace o deskriptoru zpráv viz "MQMD (Message Descriptor) na serveru IBM i" na stránce 1099.

Pole *MDENC* je 32bitové celé číslo, které je rozděleno do čtyř samostatných podpolí; tato podpole identifikují:

- Kódování použité pro binární celá čísla
- Kódování použité pro packed-decimal celá čísla
- Kódování použité pro čísla s pohyblivou řádovou čárkou
- Vyhrazené bity

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Jsou definovány následující masky:

CSIMSK
Maska pro kódování binary-integer.

Toto podpole zabírá v poli *MDENC* bitové pozice 28 až 31.

KONCOVKA
Maska pro kódování packed-decimal-integer.

Toto podpole zabírá v poli *MDENC* bitové pozice 24 až 27.

ENFMSK
Maska pro kódování s pohyblivou řádovou čárkou

Toto podpole zaujímá bitové pozice 20 až 23 v poli *MDENC* .

PRASK_PROSTŘEDÍ
Maska pro rezervované bity.

Toto podpole zabírá v poli *MDENC* bitové pozice 0 až 19.

IBM i Binární-celočíselné kódování v IBM i

Platné hodnoty pro kódování binary-integer.

Pro kódování binary-integer jsou platné následující hodnoty:

ENIUNDOVÁ
Nedefinované celočíselné kódování.

Binární celá čísla jsou znázorněna pomocí nedefinovaného kódování.

ENINOR
Normální celočíselné kódování.

Binární celá čísla jsou reprezentována konvenčním způsobem:

- Nejméně významný bajt v čísle má nejvyšší adresu kteréhokoli z bajtů v daném čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu se nachází vedle bajtu s nejbližší vyšší adresou; nejvíce významný bit v každém bajtu se nachází vedle bajtu s nejbližší nižší adresou.

ENIREV
Obrácené celočíselné kódování.

Binární celá čísla jsou znázorněna stejným způsobem jako ENINOR, ale s byty uspořádanými v obráceném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENINOR.

IBM i Packed-decimal-integer encoding on IBM i

Platné hodnoty pro kódování packed-decimal-integer

Pro kódování packed-decimal-integer jsou platné následující hodnoty:

KONCOVKA
Nedefinované kódování packed-decimal.

Pakovaný-desítková čísla jsou reprezentována pomocí nedefinovaného kódování.

KONCOVKA
Běžné zapakované kódování desetinných čísel.

Packed-decimal celá čísla jsou reprezentována v konvenčním způsobem:

- Každá desetinná číslice v tisknutelném tvaru čísla je vyjádřena v pakovaném desítkovém zápisu jedinou hexadecimální číslicí v rozsahu X' 0 'až X' 9 '. Každá hexadecimální číslice zabírá 4 bity, a tak každý bajt v pakovaném dekadickém čísle představuje dvě desetinná místa v tisknutelném tvaru čísla.
- Nejméně významný bajt v pakovaném-decimálním čísle je bajt, který obsahuje nejméně výraznou dekadickou číslici. V tomto bajtu obsahují nejvýznamnější 4 bity nejméně významnou desítkovou číslici a nejméně významné 4 bity obsahují znaménko. Znaménko je buď X'C '(kladné), X 'D' (negativní), nebo X'F ' (nepodepsaný).
- Nejméně významný bajt v čísle má nejvyšší adresu kteréhokoli z bajtů v daném čísle; nejvýznamnější bajt má nejnižší adresu.
- Nejméně významný bit v každém bajtu se nachází vedle bajtu s nejbližší vyšší adresou; nejvíce významný bit v každém bajtu se nachází vedle bajtu s nejbližší nižší adresou.

ENDREV
Obrácené kódování packed-decimal.

Packed-decimal celá čísla jsou znázorněna stejným způsobem jako ENDNOR, ale s bajty uspořádané v opačném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENDNOR.

IBM i Kódování čísel s pohyblivou řádovou čárkou v systému IBM i

Platné hodnoty pro kódování s pohyblivou řádovou

Pro kódování s pohyblivou řádovou čárkou jsou platné následující hodnoty:

ENFUNDACE
Nedefinované kódování s pohyblivou řádovou

Čísla s pohyblivou řádovou čárkou jsou reprezentována nedefinovaným kódováním.

ENFNOR
Normální kódování IEEE (Institute of Electrical and Electronics Engineers).

Čísla s pohyblivou řádovou čárkou jsou znázorněna pomocí standardního formátu IEEE s pohyblivou řádovou čárkou, přičemž bajty jsou uspořádány následujícím způsobem:

- Nejméně významný bajt v mantisy má nejvyšší adresu libovolného z bajtů v počtu; bajt obsahující exponent má nejnižší adresu.
- Nejméně významný bit v každém bajtu se nachází vedle bajtu s nejbližší vyšší adresou; nejvíce významný bit v každém bajtu se nachází vedle bajtu s nejbližší nižší adresou.

Podrobnosti o kódování typu float se standardem IEEE lze nalézt ve standardu IEEE Standard 754.

ENFREV
Reverzní kódování IEEE s plovoucí řádovou čárkou.

Čísla s pohyblivou řádovou čárkou jsou znázorněna stejným způsobem jako ENFNOR, ale s převráceným bajtům v opačném pořadí. Bity v každém bajtu jsou uspořádány stejným způsobem jako ENFNOR.

ENF390
Kódování float architektury System/390 .

Čísla s pohyblivou řádovou čárkou jsou reprezentována pomocí standardního formátu s pohyblivou řádovou čárkou System/390 . Používá se také v systému System/370.

IBM i Konstruování kódování v systému IBM i

Chcete-li vytvořit hodnotu pro pole *MDENC* v MQMD, měly by být přidány příslušné konstanty, které popisují požadovaná kódování.

Ujistěte se, že jste spojili pouze jedno z kódování ENIs jedním kódováním END* a jedním kódováním ENF*.

IBM i Analýza kódování v systému IBM i

Pole *MDENC* obsahuje podpole; z toho důvodu by aplikace, které potřebují prozkoumat celé číslo, pakované desetinné číslo nebo plovoucí kódování, měly používat techniku popsanou v tomto tématu.

Použití aritmetiky

Následující kroky by měly být provedeny pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu požadovaného kódování:
   - 1 pro kódování binárního celého čísla
   - 16 pro pakované dekadické celé kódování
   - 256 pro kódování čísel s pohyblivou řádovou čárkou

   Volejte hodnotu A.
2. Dělí hodnotu pole *MDENC* hodnotou A ; volejte výsledek B.
3. Dělí se B o 16; volejte výsledek C.
4. Multiply C od 16 a odečítat od B ; volejte výsledek D.
5. Násobení D podle A ; volejte výsledek E.
6. E je požadované kódování a lze jej testovat pro rovnost s každou z hodnot, které jsou platné pro daný typ kódování.

IBM i Souhrn kódování architektury počítače v systému IBM i

Tabulka shrnující kódování pro počítačové architektury.

Kódování pro počítačové architektury jsou zobrazeny v Tabulka 816 na stránce 1417.

Tabulka 816. Souhrn kódování pro počítačové architektury

Architektura počítače	Kódování binárních celých čísel	Packed-desítkové celočíselné kódování	Kódování čísel s pohyblivou řádovou čárkou
IBM i	normální	normální	Normální IEEE
Intel x86	Převrácené	Převrácené	IEEE převrácen
PowerPC	normální	normální	Normální IEEE

Tabulka 816. Souhrn kódování pro počítačové architektury (pokračování)			
Architektura počítače	**Kódování binárních celých čísel**	**Packed-desítkové celočíselné kódování**	**Kódování čísel s pohyblivou řádovou čárkou**
System/390	normální	normální	System/390

IBM i Volby sestav a příznaky zpráv v systému IBM i

Toto téma se týká polí *MDREP* a *MDMFL* , která jsou součástí deskriptoru zpráv MQMD určeného na voláních MQGET, MQPUT a MQPUT1 .

Další informace o deskriptoru zpráv viz “MQMD (Message Descriptor) na serveru IBM i” na stránce 1099. Tyto informace popisují:

- Struktura pole sestavy a způsob, jakým je správce front zpracovává.
- Jak by měla aplikace analyzovat pole sestavy
- Struktura pole s příznaky 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 zamítnuty, pokud je lokální správce front nerozpozná
- Volby sestavy, které jsou vždy akceptovány, i tehdy, když je lokální správce front nerozpozná
- Volby sestavy, které jsou akceptovány pouze v případě splnění určitých dalších podmínek

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Všimněte si, že bity v podpoli nejsou nutně sousedící. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Pro identifikaci podpolí jsou definovány následující masky:

RORUM
Maska pro nepodporované volby sestavy, které jsou zamítnuty.

Tato maska určuje 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 akceptovány.

Tato maska určuje bitové pozice v poli *MDREP* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, přesto akceptovány v rámci volání MQPUT nebo MQPUT1 . V tomto případě se vrací kód dokončení CCWARN s kódem příčiny RC2104 .

Toto podpole zabírá bitové pozice 0 až 2, 4 až 10, a 24 až 31.

Do tohoto podpole jsou zahrnuty následující volby sestavy:

- ROMTMTCA
- RODLQ
- RODISK
- ROEXC
- REXCD
- VÝMLUVU

- VÝMĚNNÉ
- ROEXPD
- ROEXPF
- RONAN
- RONMI
- RONAN
- ROPAN
- ROPCI
- ROPMI

ROAUXM
Maska pro nepodporované volby sestavy, které jsou přijímány pouze za určitých okolností.

Tato maska určuje bitové pozice v poli *MDREP* , kde jsou volby sestavy, které nejsou podporovány lokálním správcem front, nicméně přijaty na základě 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 (tedy fronta určená poli *ODMN* a *ODON* v deskriptoru objektu uvedeném v volání MQOPEN nebo MQPUT1 není lokální přenosová fronta).

Kód dokončení CCWARN s kódem příčiny RC2104 je vrácen, pokud jsou tyto podmínky splněny, a CCFAIL s kódem příčiny RC2061 , pokud ne.

Toto podpole zabírá bitové pozice 14 až 23.

Do tohoto podpole jsou zahrnuty následující volby sestavy:

- ROCOA
- ROCOAD
- ROCOAF
- TRESKA OBECNÁ
- RODOKD
- RODCODF

Pokud jsou v poli *MDREP* zadány nějaké volby, které správce front nerozpozná, zkontroluje správce front každé dílčí pole postupně pomocí bitové operace AND a zkombinuje pole *MDREP* s maskou pro toto dílčí pole. Není-li výsledek této operace nula, vrátí se kód dokončení a kódy příčiny popsané dříve.

Je-li vráceno CCWARN, není nadefinováno, který kód příčiny je vrácen, pokud existují jiné varovné podmínky.

Možnost zadat a přijmout volby sestavy, které nejsou rozpoznány lokálním správcem front, je užitečné, když je nutné odeslat zprávu s volbou sestavy, která bude rozpoznána a zpracována *vzdáleným* správcem front.

IBM i Analýza pole sestavy v systému IBM i

Pole MDREP obsahuje podpole. Z tohoto důvodu je třeba, aby některé aplikace zkontrolovaly, zda odesilatel zprávy požadoval konkrétní sestavu. Tyto aplikace by měly používat techniku popsanou v tomto tématu.

Použití aritmetiky

Následující kroky by měly být provedeny pomocí celočíselné aritmetiky:

1. Vyberte jednu z následujících hodnot podle typu sestavy, která má být zkontrolována:
   - ROCOA pro zprávu COA

- ROCOD pro hlášení COD
- ROEXC pro hlášení výjimek
- Sestava ROEXP pro vypršení platnosti

Volejte hodnotu A.

2. Rozdělte pole *MDREP* hodnotou A ; volejte výsledek B.
3. Rozdělit B podle 8 ; volejte výsledek C.
4. Násobení C od 8 a odečtením od B ; volejte výsledek D.
5. Násobení D podle A ; volejte výsledek E.
6. Testujte E for equality s každou z hodnot, které jsou pro tento typ sestavy možné.

   Například, pokud je A ROEXC, testujte E pro rovnost s každou z následujících možností, abyste určili, co bylo uvedeno odesilatelem zprávy:

   - RONAN
   - ROEXC
   - REXCD
   - VÝMLUVU

   Testy lze provádět v libovolném pořadí, které je nejvhodnější pro logiku aplikace.

Následující pseudokód ilustruje tuto techniku pro zprávy hlášení výjimek:

```
A = ROEXC
B = Report/A
C = B/8
D = B - C*8
E = D*A
```

A similar method can be used to test for the ROPMI or ROPCI options; select as the value A whichever of these two constants is appropriate, and then proceed as described previously, but replacing the value 8 in the previous steps by the value 2.

IBM i Struktura pole zpráv-flags v systému IBM i

Pole *MDMFL* je 32bitové celé číslo, které je rozděleno do tří samostatných dílčích polí.

Tato podpole identifikují:

- Příznaky zpráv, které jsou zamítnuty v případě, že je lokální správce front nerozpozná
- Příznaky zpráv, které jsou vždy akceptovány, i když je lokální správce front nerozpozná
- Příznaky zpráv, které jsou akceptovány pouze v případě splnění určitých dalších podmínek

Poznámka: Všechna podpole v produktu *MDMFL* jsou vyhrazena pro použití správcem front.

Každé dílčí pole je označeno bitovou maskou, která má 1-bity v pozicích odpovídajících podpoli, a 0-bity jinde. Bity jsou očíslovány tak, že bit 0 je nejvíce významný bit, a bit 31 je nejméně významný bit. Pro identifikaci podpolí jsou definovány následující masky:

MFRUM
Maska pro nepodporované příznaky zpráv, které jsou zamítnuty.

Tato maska určuje 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.

Do tohoto podpole jsou zahrnuty následující příznaky zpráv:

- MFLMIG

- MFLSEG
- MFMIG
- MFSEG
- MFSEGA
- MFSEGŠTINA

MFAUM
Maska pro nepodporované příznaky zpráv, které jsou akceptovány.

Tato maska určuje bitové pozice v poli *MDMFL* , kde zprávy o znacích, které nejsou podporovány lokálním správcem front, však budou přijaty na základě volání MQPUT nebo MQPUT1 . Kód dokončení je CCOK.

Toto podpole zabírá bitové pozice 0 až 11.

MFAUXM
Maska pro nepodporované příznaky zpráv, které jsou přijímány pouze za určitých okolností.

Tato maska určuje bitové pozice v poli *MDMFL* , kde zprávy o znacích, které nejsou podporovány lokálním správcem front, však budou přijaty na základě 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 (tedy fronta určená poli *ODMN* a *ODON* v deskriptoru objektu uvedeném v volání MQOPEN nebo MQPUT1 není lokální přenosová fronta).

Kód Completion CCOK je vrácen, pokud jsou tyto podmínky splněny, a CCFAIL s kódem příčiny RC2249 , pokud tomu tak není.

Toto podpole zabírá bitové pozice 12 až 19.

Pokud jsou v poli *MDMFL* zadány parametry, které správce front nerozpoznal, zkontroluje správce front každé dílčí pole postupně pomocí bitové operace AND, aby zkombinoval pole *MDMFL* s maskou pro toto podpole. Není-li výsledek této operace nula, vrátí se kód dokončení a kódy příčiny popsané dříve.

IBM i Převod dat v systému IBM i

Toto téma popisuje rozhraní pro uživatelskou proceduru pro převod dat a zpracování prováděné správcem front při požadavku na převod dat.

Uživatelská procedura pro převod 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 na volání MQGET.

Popsána jsou popsány následující aspekty konverze dat:

- Zpracování prováděné správcem front v odpovědi na volbu GMCONV, viz “Zpracování konverze v systému IBM i” na stránce 1422.
- Konvence zpracování použité správcem front při zpracování vestavěného formátu; tyto konvence se doporučují také pro uživatelské procedury zápisu. Viz “Konvence zpracování v systému IBM i” na stránce 1423.
- Speciální aspekty konverze zpráv hlášení; viz “Převod zpráv sestav v systému IBM i” na stránce 1426.
- Parametry předané uživatelské proceduře pro převod dat; viz “MQCONVX (Ukončení převodu dat) v systému IBM i” na stránce 1437.
- Volání, které lze použít z uživatelské procedury ke konverzi znakových dat mezi různými reprezentacemi; viz “MQXCNVC (Konverze znaků) na IBM i” na stránce 1432.
- Parametr datové struktury, který je specifický pro uživatelskou proceduru; viz “MQDXP (parametr uživatelské procedury konverze dat) v systému IBM i” na stránce 1427.

IBM i Zpracování konverze v systému IBM i

Tyto informace popisují zpracování prováděné správcem front jako odpověď na volbu GMCONV.

Správce front provede následující akce, je-li volba GMCONV určena v rámci volání MQGET a zpráva má být vrácena do aplikace:

1. Je-li splněna jedna nebo více z následujících podmínek, není převod nutný:
   - Data zprávy jsou již ve znakové sadě a kódování požadované aplikací, která vydala volání MQGET. Aplikace musí nastavit pole *MDCSI* a *MDENC* v parametru **MSGDSC** v rámci volání MQGET na vyžadované hodnoty před zadáním volání.
   - Délka dat zprávy je nula.
   - Délka parametru **BUFFER** volání MQGET je nulová.

   V těchto případech je zpráva vrácena bez převodu na aplikaci, která vydala volání MQGET; hodnoty *MDCSI* a *MDENC* v parametru **MSGDSC** jsou nastaveny na hodnoty v řídicích informacích ve zprávě a volání je dokončeno s jednou z následujících kombinací kódu dokončení a kódu příčiny:

   Kód dokončení
   Kód příčiny

   KEK
   RCNONE

   CCWARN
   RC2079

   CCWARN
   RC2080

Následující kroky se provádějí pouze v případě, že znaková sada nebo kódování dat zprávy se liší od odpovídající hodnoty v parametru **MSGDSC** a že jsou data k převedení:

1. Pokud má pole *MDFMT* v informacích o ovládacím prvku ve zprávě hodnotu FMNONE, je vrácena nekonvertovaný zpráva s kódem dokončení CCWARN a kódem příčiny RC2110.

   Ve všech ostatních případech zpracování konverze pokračuje.
2. Zpráva se odstraní z fronty a umístí se do dočasné vyrovnávací paměti, která má stejnou velikost jako parametr **BUFFER** . Pro operace procházení je zpráva kopírována do dočasné vyrovnávací paměti místo toho, aby byla odebrána z fronty.
3. Pokud má být zpráva oseknuta tak, aby se vešla do vyrovnávací paměti, provede se následující:
   - Pokud nebyla uvedena volba GMATM, vrátí se nekonvertované zprávy s kódem dokončení CCWARN a kódem příčiny RC2080.
   - Je-li uvedena volba GMATM *byla* , kód dokončení je nastaven na CCWARN, kód příčiny je nastaven na RC2079a zpracování konverze pokračuje.
4. Pokud může být zpráva umístěna ve vyrovnávací paměti bez oříznutí nebo byla zadána volba GMATM, je provedeno následující:
   - Je-li formát vestavěným formátem, vyrovnávací paměť se předá do služby pro převod dat správce front.
   - Pokud formát není vestavěný formát, vyrovnávací paměť se předává uživatelské proceduře, která má stejný název jako formát. Pokud nelze nalézt uživatelskou proceduru, vrátí se nekonvertovaný zpráva s kódem dokončení CCWARN a kódem příčiny RC2110.

   Pokud se nevyskytne žádná chyba, výstup ze služby pro převod dat nebo z uživatelem napsaného ukončení je převedená zpráva a kód dokončení a kód příčiny, které se vrátí do aplikace, která volala příkaz MQGET.
5. Je-li konverze úspěšná, správce front vrátí převedenou zprávu na aplikaci. V takovém případě bude kód dokončení a kód příčiny vrácený voláním MQGET obvykle jedna z následujících kombinací:

Kód dokončení
Kód příčiny

KEK
RCNONE

CCWARN
RC2079

Je-li však konverze provedena uživatelskou procedurou, mohou být vráceny jiné kódy příčiny, i když je konverze úspěšná.

Pokud konverze selže (z jakéhokoli důvodu), správce front vrátí nekonvertovaný zprávu do aplikace, s poli *MDCSI* a *MDENC* v parametru **MSGDSC** se nastaví na hodnoty v řídicích informacích ve zprávě a s kódem dokončení CCWARN.

IBM i Konvence zpracování v systému IBM i

Při převádění vestavěného formátu postupuje správce front konvencemi zpracování popsanými v tomto tématu.

Zvažte použití těchto konvencí na uživatelské procedury, které jsou zapsány uživatelem, ačkoli správce front toto oprávnění nevynucuje. Vestavěné formáty převedené správcem front jsou následující:

- FMADMN
- FMMUDE
- FMCKA
- FMPCF
- FMCMD1
- FMRMHCACH
- FMCMD2
- FMRFH
- FMDLH
- FMRFH2
- FMDH/
- FMSTR
- FMEVNT
- FMTM
- FMIMY
- FMXQH
- FMIMVS

1. Pokud se zpráva během převodu rozbalí a překročí velikost parametru **BUFFER** , provede se následující akce:
   - Pokud nebyla uvedena volba GMATM, vrátí se nekonvertované zprávy s kódem dokončení CCWARN a kódem příčiny RC2120.
   - Pokud byla uvedena volba GMATM *was* , zpráva je zkrácena, kód dokončení je nastaven na CCWARN, kód příčiny je nastaven na RC2079, a zpracování konverze pokračuje.
2. Dojde-li k oříznutí (buď před nebo během převodu), je možné, aby počet platných bajtů vrácených v parametru **BUFFER** byl *menší než* délku vyrovnávací paměti.

   K tomu může dojít například v případě, že se jedná o 4bajtové celé číslo nebo o znak DBCS, který je strdles na konec vyrovnávací paměti. Neúplný prvek informací není převeden, a proto tyto bajty ve vrácené zprávě neobsahují platné informace. K tomu může dojít také v případě, že byla během převodu oříznuta zpráva, která byla oříznuta před konverzí převodu.

Pokud je počet vrácených platných bajtů menší než délka vyrovnávací paměti, nepoužité bajty na konci vyrovnávací paměti jsou nastaveny na hodnotu null.

3. Pokud pole nebo řetězec obsahuje konec vyrovnávací paměti, je konvertováno tolik dat, kolik je možné; pouze konkrétní prvek pole nebo znak DBCS, který je nekompletní, nekonvertuje- předcházející prvky pole nebo znaky jsou převedeny.
4. Dojde-li k oříznutí (před nebo během převodu), délka vrácená pro parametr **DATLEN** je délka zprávy *nepřevedené* před oříznutím.
5. Pokud se řetězce převádějí mezi jednobajtovými znakovými sadami (SBCS), dvoubajtovými znakovými sadami (DBCS) nebo vícebajtovými znakovými sadami (MBCS), řetězce se mohou rozšiřovat nebo uzavírat smlouvy.
   - Ve formátech PCF FMADMN, FMEVNT a FMPCF se řetězce v strukturách MQCFST a MQCFSL rozšiřují nebo podle potřeby přizpůsobí řetězci po převodu.

     Pro strukturu seznamu řetězců MQCFSL se mohou řetězce v seznamu rozšiřovat nebo uzavírat podle různých částek. Pokud k tomu dojde, správce front vycpává kratší řetězce mezerami tak, aby jejich délka byla stejná jako nejdelší řetězec po převodu.
   - Ve formátu FMRMH jsou řetězce adresované poli RMSEO, RMSNO, RMDEOa RMDNO rozšiřovány nebo podle potřeby podle potřeby akceptovány za účelem umístění řetězců po konverzi.
   - Ve formátu FMRFH se pole RFNVS rozšiřuje nebo podle potřeby rozšiřuje tak, aby bylo možné přizpůsobit dvojice názvu a hodnoty po převodu.
   - Ve strukturách s pevnými velikostmi polí umožňuje správce front rozšiřovat nebo uzavírat smlouvy v rámci svých pevných polí, pokud nejsou 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 rozvine, ale pouze nevýznamné znaky je třeba zahodit, aby bylo možné umístit převedený řetězec do pole, konverze uspěje a volání skončí s CCOK a kód příčiny RCNONE (nepředpokládá se žádné další chyby).
     - Pokud se řetězec rozvine, ale převedený řetězec vyžaduje, aby byly do pole vhozeny velké znaky, aby se do pole vešly, byla vrácena nekonvertovaný zpráva a volání je dokončeno s CCWARN a kódem příčiny RC2190.

       Poznámka: Kód příčiny RC2190 má za následek určení, zda byla zadána volba GMATM.
     - Pokud jsou řetězce smlouvy, správce front vycpávky z řetězce s mezerami do délky pole.
6. Pro zprávy sestávající z jedné nebo více struktur záhlaví IBM MQ následovaných uživatelskými daty je možné jeden nebo více struktur záhlaví, které mají být převedeny, zatímco zbytek zprávy nikoli. Nicméně se dvěma výjimkami vždy pole MDCSI a MDENC v každé struktuře záhlaví vždy správně označují znakovou sadu a kódování dat, která se řídí strukturou záhlaví.

   Tyto 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 informacích o řídicích informacích načítané zprávy nebo v argumentu **MSGDSC** určují hodnoty, které nejsou definovány nebo nejsou podporovány, může správce front chybu ignorovat, pokud nedefinovaná nebo nepodporovaná hodnota nemusí být při převodu zprávy použita.

   Pokud například pole MDENC ve zprávě určuje nepodporované kódování s pohyblivou řádovou čárkou, ale zpráva obsahuje pouze celočíselné údaje nebo obsahuje data s pohyblivou řádovou čárkou, která nevyžadují převod (protože zdrojová a cílová kódování s pohyblivou řádovou čárkou jsou identické), může dojít k chybě nebo nemusí být diagnostikována chyba.

   Je-li chyba diagnostikována, zpráva se vrátí nekonvertovaný, kód dokončení CCWARN a jeden z RC2111, RC2112, RC2113, RC2114 nebo RC2115, RC2116, RC2117, RC2118 kódů příčiny (podle potřeby); pole MDCSI a MDENC v parametru **MSGDSC** jsou nastaveny na hodnoty v řídicích informacích ve zprávě.

Není-li chyba diagnostikována a konverze se úspěšně dokončí, hodnoty vrácené v polích MDCSI a MDENC v argumentu **MSGDSC** jsou hodnoty zadané aplikací zadávající volání MQGET.

8. Ve všech případech je zpráva vrácena do aplikace bez převedení kódu dokončení je nastavena na hodnotu CCWARN a pole MDCSI a MDENC v parametru **MSGDSC** jsou nastaveny na hodnoty odpovídající nekonverzeným datům. To je také provedeno pro FMNONE také.

   Argument **REASON** je nastaven na kód, který udává, proč se konverze nepodařilo provést, pokud zpráva také nebyla oříznuta; kódy příčiny související s oseknutím mají přednost před kódy příčiny souvisejícími s převodem. (Chcete-li určit, zda byla zkrácená zpráva převedena, zkontrolujte hodnoty vrácené v polích MDCSI a MDENC v parametru **MSGDSC** .)

   Je-li diagnostikována chyba, je vrácen specifický kód příčiny nebo obecný kód příčiny RC2119. Vrácený kód příčiny závisí na schopnostech diagnostiky základní služby pro převod dat.

9. Je-li vrácen kód dokončení CCWARN a je relevantní více než jeden kód příčiny, bude mít pořadí přednosti následující pořadí:

   a. Následující příčina má přednost před všemi ostatními:

      - RC2079

   b. Další z priority je následující příčina:

      - RC2110

   c. Pořadí priorit v rámci zbývajících kódů příčiny není definováno.

10. Po dokončení volání MQGET:

    - Následující kód příčiny indikuje, že zpráva byla úspěšně převedena:
      - RCNONE
    - Následující kód příčiny informuje o tom, že zpráva *může* byla úspěšně převedena (zkontrolujte pole MDCSI a MDENC v parametru **MSGDSC** , abyste zjistili aktuální informace):
      - RC2079
    - Všechny ostatní kódy příčiny indikují, že zpráva nebyla převedena.

Následující zpracování je specifické pro vestavěné formáty; nevztahuje se na uživatelem definované formáty:

1. S výjimkou následujících formátů:

   - FMADMN
   - FMEVNT
   - FMIMVS
   - FMPCF
   - FMSTR

   žádný z vestavěných formátů nelze převést ze znakových sad nebo do znakových sad, které nemají znaky SBCS pro znaky, které jsou platné ve názvech front. Je-li proveden pokus o provedení takové konverze, je vrácena nekonvertovaný zpráva s kódem dokončení CCWARN a kódem příčiny RC2111 nebo RC2115, jak je to vhodné.

   Znaková sada Unicode UTF-16 je příklad znakové sady, která nemá znaky SBCS pro znaky, které jsou platné ve jménech front.

2. Jsou-li data zprávy pro vestavěný formát zkrácena, pole ve zprávě obsahující délky řetězců nebo počty prvků nebo struktur nejsou upraveny tak, aby odrážela délku dat vrácených do aplikace; hodnoty vrácené pro taková pole v datech zprávy jsou hodnoty použitelné pro zprávu před oseknutím.

   Při zpracování zpráv, jako je 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 touto hodnotou může následovat nula nebo více bajtů dat zprávy aplikace. Formát, znaková sada a kódování dat zprávy

aplikace jsou definovány v polích DLFMT, DLCSIa DLENC ve struktuře MQDLH na začátku zprávy. Vzhledem k tomu, že struktura MQDLH a data zprávy aplikace mohou mít různé znakové sady a kódování, je možné, že pro strukturu MQDLH a pro data zprávy aplikace je třeba provést převod.

Správce front převede nejprve strukturu MQDLH podle potřeby. Pokud je převod úspěšný, nebo struktura MQDLH nevyžaduje převod, správce front zkontroluje pole DLCSI a DLENC ve struktuře MQDLH, aby zjistil, zda je vyžadována konverze dat zprávy aplikace. Je-li požadována konverze, vyvolá správce front uživatelskou proceduru s názvem zadaným polem DLFMT ve struktuře MQDLH nebo provede vlastní převod (pokud DLFMT je název vestavěného formátu).

Pokud volání MQGET vrátí kód dokončení CCWARN a kód příčiny je jeden z těch, které označují, že konverze nebyla úspěšná, použije se jedna z následujících možností:

- Strukturu MQDLH nelze převést. V tomto případě data zprávy aplikace nebudou konvertována.
- 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 bylo možné určit, která z předchozích použití se použije.

4. Je-li název formátu FMXQH, data zprávy začínají strukturou MQXQH a za ním může následovat nula nebo více bajtů dalších dat. Tato přídavná data jsou obvykle data zprávy aplikace (která mohou mít nulovou délku), ale na začátku dalších dat může být přítomna také jedna nebo více struktur záhlaví IBM MQ .

   Struktura MQXQH musí být ve znakové sadě a kódování správce front. Formát, znaková sada a kódování dat následující strukturu MQXQH jsou dány poli MDFMT, MDCSIa MDENC ve struktuře MQMD obsažené v MQXQH. Pro každou následující strukturu záhlaví produktu IBM MQ se pole MDFMT, MDCSIa MDENC ve struktuře popisují data, která následují za touto strukturou; tato data jsou buď jiná struktura záhlaví IBM MQ , nebo data zprávy aplikace.

   Je-li pro zprávu FMXQH uvedena volba GMCONV, budou převedena data zprávy aplikace a některé ze struktur záhlaví MQ , ale data ve struktuře MQXQH se nekonvertují. Při návratu z volání MQGET proto postupujte takto:

   - Hodnoty polí MDFMT, MDCSIa MDENC v parametru **MSGDSC** popisují data ve struktuře MQXQH a nikoli data zprávy aplikace; hodnoty proto nebudou stejné jako hodnoty určené aplikací, která vydala volání MQGET.

     Výsledkem je to, že aplikace, která opakovaně získává zprávy z přenosové fronty s uvedenou 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. Nejsou-li k dispozici žádné jiné struktury záhlaví produktu IBM MQ , jsou data zpráv aplikace popisována těmito poli ve struktuře MQMD v rámci struktury MQXQH. Je-li konverze úspěšná, budou hodnoty stejné jako hodnoty zadané v parametru **MSGDSC** aplikací, která vydala volání MQGET.

   Pokud se jedná o zprávu distribučního seznamu, je struktura MQXQH následována strukturou MQDH (spolu s jejími poli záznamů MQOR a MQPMR), které mohou být následně následovány nulou nebo více dalšími strukturami záhlaví IBM MQ a s nulovým počtem bajtů dat zprávy aplikace. Stejně jako struktura MQXQH se struktura MQDH musí nacházet ve znakové sadě a kódování správce front a nebude převedena na volání MQGET, i když je zadána volba GMCONV.

   Zpracování dříve popisovaných struktur MQXQH a MQDH jsou primárně určeny pro použití agenty kanálů zpráv při získávání zpráv z přenosových front.

IBM i Převod zpráv sestav v systému IBM i

Zpráva sestavy může obsahovat různé množství dat aplikační zprávy, v závislosti na volbách sestavy uvedených odesílatelem původní zprávy.

Zpráva sestavy může obsahovat zejména:

1. Žádná data zprávy aplikace

2. Některá data zprávy aplikace z původní zprávy

   K tomu dojde, když odesilatel původní zprávy uvádí RO* D a zpráva je delší než 100 bajtů.

3. Všechna data zprávy aplikace z původní zprávy

   K tomu dojde, když odesilatel 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 s hlášením, zkopíruje název formátu z původní zprávy do pole *MDFMT* v řídicí informaci ve zprávě sestavy. Název formátu ve zprávě sestavy může proto znamenat délku dat, která se liší od délky uvedené ve zprávě hlášení (případy 1 a 2 popsané dříve).

Je-li při načítání zprávy sestavy uvedena volba GMCONV, postupujte takto:

- Pro případ, který byl popsán dříve, nebude volání konverze dat vyvoláno (protože zpráva hlášení nebude mít žádná data).
- Pro případ 3 popsané dříve název formátu správně implikuje délku dat zprávy.
- Ale pro případ 2 popsané dříve bude uživatelská procedura převodu dat vyvolána pro převod zprávy, která je *kratší* než délka, která je implikovaná názvem formátu.

  Kód příčiny předaný uživatelské proceduře bude obvykle nastaven na hodnotu RCNONE (to znamená, že kód příčiny nebude označovat, že zpráva byla zkrácena). Důvodem je skutečnost, že data zprávy byla zkrácena *odesilatelem* zprávy sestavy a nikoli správcem front příjemce v odezvě na volání MQGET.

Vzhledem k těmto možnostem by uživatelská procedura pro převod dat neměla používat název formátu k vyvození délky dat, která mu byla předána; místo procedury ukončení by měla být zkontrolovaná délka poskytnutých dat a aby byla připravena převést menší množství dat, než je délka, která je odvozena z názvu formátu. Pokud lze data úspěšně převést, kód dokončení CCOK a kód příčiny RCNONE by měly být vráceny uživatelskou procedurou. Délka dat zprávy, která má být konvertována, je předána ukončení jako parametr **INLEN** .

Rozhraní pro programování závislé na produktu

Pokud zpráva sestavy obsahuje informace o aktivitě, která byla provedena, je známá jako zpráva o aktivitě. Mezi příklady činností patří:

- MCA při odesílání zprávy z fronty mimo kanál,
- MCA při příjmu zprávy z kanálu a jeho vložení do fronty,
- nedoručených zpráv MCA-zařazení nedoručitelné zprávy do fronty
- MCA při získávání zprávy z fronty a její vyřazení
- obslužná rutina dead-letter, která umístí zprávu zpět do fronty
- příkazový server zpracovávající požadavek PCF-zprostředkovatel zpracovávající požadavek na publikování
- uživatelská aplikace získávajícího zprávu z fronty-uživatelská aplikace prohledá zprávu ve frontě,

Libovolná aplikace, včetně správce front, může přidat některá data zprávy do sestavy aktivity za záhlavím sestavy. Množství dat, které by mělo být dodáno, pokud některé je odesláno, není opraveno a je rozhodnuto aplikací. Vrácené informace by měly být užitečné pro zpracování aplikace v sestavě aktivity. Sestavy aktivity správce front budou s nimi vracet všechny standardní struktury záhlaví produktu IBM MQ (začátek 'MQH') obsažené v původní zprávě. To zahrnuje například všechna záhlaví MQRFH2 , která byla zahrnuta do původní zprávy. Také správce front vrátí nalezenou hlavičku MQCFH, ale ne přidružené parametry PCF, které jsou k němu přidruženy. To poskytuje aplikacím monitorování představu o tom, o čem se zpráva týká.

IBM i MQDXP (parametr uživatelské procedury konverze dat) v systému IBM i

Blok parametru ukončení konverze dat.

Přehled

Účel: Struktura MQDXP je parametr, který správce front předá výstupnímu programu pro převod dat při vyvolání uživatelské procedury pro převod dat zprávy v rámci zpracování volání MQGET. Podrobnosti o ukončení konverze dat najdete v popisu volání MQCONVX.

Znaková sada a kódování: Znaková data v MQDXP jsou ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front **CodedCharSetId** . Numerická data v MQDXP jsou v nativním kódování počítače; to je dáno ENNAT.

Použití: uživatelská procedura může změnit pouze pole *DXLEN*, *DXCC*, *DXREA* a *DXRES* v MQDXP; změny v jiných polích budou ignorovány. Pole *DXLEN* však nelze změnit, jestliže převáděná zpráva je segment, který obsahuje pouze část logické zprávy.

Když se řízení vrátí ke správci front z uživatelské procedury, správce front zkontroluje hodnoty vrácené MQDXP. Pokud vrácené hodnoty nejsou platné, bude správce front pokračovat ve zpracování, jako kdyby byla v produktu *DXRES* vrácena uživatelská procedura XRFAIL. Správce front však ignoruje hodnoty polí *DXCC* a *DXREA* vrácených uživatelskou procedurou v tomto případě a použije místo toho hodnoty, které měla tato pole na hodnotě *vstup* pro ukončení. Následující hodnoty v MQDXP způsobí, že se toto zpracování bude provádět:

- Pole *DXRES* není XROK a ne XRFAIL
- Pole *DXCC* není CCOK a ne CCWARN
- Pole *DXLEN* menší než nula nebo *DXLEN* se změnilo, když převáděná zpráva je segment, který obsahuje pouze část logické zprávy.
- “Pole” na stránce 1428
- “Deklarace RPG (kopie souboru CMQDXPH)” na stránce 1432

Pole

Struktura MQDXP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

DXAOP (10ciferné celé číslo se znaménkem)

Volby aplikace.

Jedná se o kopii pole *GMOPT* struktury MQGMO určeného aplikací, která vydala volání MQGET. Ukončení může být nutné prozkoumat, aby se zjistilo, zda byla uvedena volba GMATM.

Toto je vstupní pole pro ukončení.

DXCC (10ciferné celé číslo se znaménkem)

Kód dokončení.

Když je vyvoláno ukončení, obsahuje kód dokončení, který bude vrácen do aplikace, která vydala volání MQGET, pokud se ukončení rozhodne nedělat nic. Vždy je to CCWARN, protože buď byla zpráva zkrácena, nebo zpráva požaduje konverzi, a to ještě nebylo provedeno.

Na výstupu z uživatelské procedury obsahuje toto pole kód dokončení, který má být vrácen do aplikace v parametru **CMPCOD** volání MQGET; jsou platné pouze parametry CCOK a CCWARN. Podívejte se na popis pole *DXREA* , kde najdete návrhy, jak by procedura měla nastavit toto pole na výstupu.

Jedná se o vstupní/výstupní pole pro ukončení.

DXCSI (10ciferné celé číslo se znaménkem)

Znaková sada vyžadovaná aplikací.

Jedná se o identifikátor kódované znakové sady znakové sady vyžadované aplikací, která vydala volání MQGET; viz pole *MDCSI* ve struktuře MQMD pro více podrobností. Pokud aplikace určuje speciální hodnotu CSQM u volání MQGET, změní ji správce front na skutečný identifikátor znakové sady znakové sady použité správcem front před vyvoláním uživatelské procedury.

Je-li konverze úspěšná, uživatelská procedura by měla kopírovat toto pole do pole *MDCSI* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

DXENC (10ciferné celé číslo se znaménkem)

Numerické kódování požadované aplikací.

Jedná se o číselné kódování požadované aplikací, která vydala volání MQGET. Další podrobnosti naleznete v poli *MDENC* ve struktuře MQMD.

Je-li konverze úspěšná, uživatelská procedura by měla kopírovat toto pole do pole *MDENC* v deskriptoru zprávy.

Toto je vstupní pole pro ukončení.

DXHCN (10číslicové celé číslo se znaménkem)

Manipulátor připojení.

Jedná se o manipulátor připojení, který lze použít při volání MQXCNVC. Tento manipulátor nemusí být nutně stejný jako popisovač určený aplikací, která vydala volání MQGET.

DXLEN (10ciferné celé číslo se znaménkem)

Délka dat zprávy v bajtech.

Když je vyvoláno ukončení, toto pole obsahuje původní délku dat zprávy aplikace. Pokud byla zpráva zkrácena, aby se vešla do vyrovnávací paměti poskytnuté aplikací, velikost zprávy poskytnuté při ukončení bude *menší* než hodnota parametru *DXLEN*. Velikost zprávy poskytované při ukončení je vždy dána parametrem **INLEN** ukončení, bez ohledu na případné oříznutí, které se mohlo vyskytnout.

Oříznutí je indikováno polem *DXREA* , které má hodnotu RC2079 na vstupu do uživatelské procedury.

Většina konverzí nebude muset tuto délku změnit, ale v případě potřeby to může provést ukončení; hodnota nastavená uživatelskou procedurou se vrátí do aplikace v parametru **DATLEN** volání MQGET. Tato délka však nemůže být změněna, pokud převáděná zpráva je segment, který obsahuje pouze část logické zprávy. Důvodem je to, že změna délky by způsobila, že odchylky dalších segmentů v logické zprávě budou nesprávné.

Všimněte si, že pokud chce uživatelská procedura změnit délku dat, uvědomte si, 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 se přemístil kurzor procházení pro požadavek na procházení) a není ovlivněn žádnou změnou délky dat způsobené převodem. Z tohoto důvodu se doporučuje, aby převodní procedura nezpůsobila změnu v délce dat zprávy aplikace.

Pokud převod znaků implikuje změnu délky, lze řetězec převést na jiný řetězec se stejnou délkou v bajtech, zkracovat koncové mezery nebo vyplňovat mezerami podle potřeby.

Uživatelská procedura se nevyvolá, pokud zpráva neobsahuje žádná data zprávy aplikace; proto je *DXLEN* vždy větší než nula.

Jedná se o vstupní/výstupní pole pro ukončení.

DXREA (10ciferné celé číslo se znaménkem)

Kód příčiny kvalifikující *DXCC*.

Když je vyvolána uživatelská procedura, obsahuje kód příčiny, který bude vrácen do aplikace, která vydala volání MQGET, pokud se uživatelská procedura rozhodne pro nic neprovést. Mezi možné hodnoty patří RC2079, což znamená, že zpráva byla zkrácena, aby se vešla do vyrovnávací paměti poskytnuté aplikací, a RC2119, což znamená, že zpráva vyžaduje konverzi, ale že tato zpráva ještě nebyla hotova.

Na výstupu z uživatelské procedury je toto pole obsahovat důvod vrátit aplikaci do parametru **REASON** volání MQGET; doporučuje se následující:

- Pokud měl parametr *DXREA* hodnotu RC2079 na vstupu do uživatelské procedury, neměla by být pole *DXREA* a *DXCC* změněna bez ohledu na to, zda je převod úspěšný nebo neúspěšný.

  (Pokud pole *DXCC* není CCOK, aplikace, která načte zprávu, může identifikovat selhání převodu porovnáním vrácených hodnot *MDENC* a *MDCSI* v deskriptoru zpráv s požadovanými hodnotami; naopak aplikace nemůže rozlišit oříznutou zprávu od zprávy, která právě namontuje vyrovnávací paměť. Z tohoto důvodu by měl být návratový kód RC2079 vrácen přednostně s libovolným z důvodů, které indikují selhání převodu.)
- Pokud má *DXREA* jakoukoli jinou hodnotu na vstupu do výstupu:
  - Pokud je konverze úspěšná, *DXCC* by mělo být nastaveno na CCOK a *DXREA* nastaveno na RCNONE.
  - Pokud převod selže nebo se zpráva rozbalí a musí být oříznuta tak, aby se vešla do vyrovnávací paměti, *DXCC* by měla být nastavena na CCWARN (nebo ponechána nezměněná) a *DXREA* nastavena na jednu z hodnot i na následujícím seznamu, aby označovala povahu selhání.

    Všimněte si, že pokud je zpráva po převodu příliš velká pro vyrovnávací paměť, měla by být zkrácena pouze v případě, že aplikace, která vydala volání MQGET, byla zadána pomocí volby GMATM:

    - Pokud jste tuto volbu uvedli, měla by být vrácena příčina RC2079 .
    - Pokud jste tuto volbu neuvedli, měla by se zpráva vrátit bez převodu s kódem příčiny RC2120.

Kódy příčiny v následujícím seznamu jsou doporučeny pro použití uživatelskou procedurou k určení příčiny selhání převodu, ale tento výstup může vracet jiné hodnoty ze sady kódů RC*, je-li to považováno za vhodné. Kromě toho je rozsah hodnot RC0900 až RC0999 alokován pro použití uživatelskou procedurou za účelem označení podmínek, které chce uživatelská procedura komunikovat s aplikací, která vydala volání MQGET.

Poznámka: Pokud zprávu nelze úspěšně převést, musí v poli *DXRES* návratový kód vrátit hodnotu XRFAIL, aby mohl správce front vrátit nepřevedenou zprávu. To je pravda 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 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

RC2119
(2119, X'847 ') Data zprávy nejsou 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 čísel s pohyblivou řádovou čárkou ve zprávě nebylo rozpoznáno.

RC2112
(2112, X'840 ') Kódování celého čísla zdroje nebylo rozpoznáno.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2117
(2117, X'845 ') Packed-decimal encoding specified by receiver not recognized.

RC2118
(2118, X'846 ') Kódování čísel s pohyblivou řádovou čárkou určené příjemcem není rozpoznáno.

RC2116
(2116, X'844 ') Cílové celé číslo kódování nebylo rozpoznáno.

RC2079
(2079, X'81F') Byla vrácena oříznutá zpráva (zpracování dokončeno).

Jedná se o vstupní/výstupní pole pro ukončení.

DXRES (10ciferné celé číslo se znaménkem)

Odezva z ukončení.

Toto nastavení je nastaveno na základě ukončení, aby se označilo úspěch nebo jinak konverze. Musí se jednat o jeden z následujících:

XROK
Převod byl úspěšný.

Pokud tato hodnota určuje tuto hodnotu, vrátí správce front následující informace o aplikaci, která vydala volání MQGET:

- Hodnota pole *DXCC* na výstupu z uživatelské procedury
- Hodnota pole *DXREA* na výstupu z uživatelské procedury
- Hodnota pole *DXLEN* na výstupu z uživatelské procedury
- Obsah výstupní vyrovnávací paměti výstupu *OUTBUF*. Počet vrácených bajtů je menší z hodnot 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 *both* nezměněná, vrátí správce front následující zprávy:

- Hodnota polí *MDENC* a *MDCSI* ve struktuře MQDXP na *vstupu* do uživatelské procedury

Pokud byla změněna jedna nebo obě pole *MDENC* a *MDCSI* v parametru deskriptoru zpráv uživatelské procedury, vrátí správce front následující zprávy:

- Hodnota polí *MDENC* a *MDCSI* v parametru deskriptoru zprávy uživatelské procedury na výstupu z uživatelské procedury.
-

XRFAIL
Převod byl neúspěšný.

Pokud tato hodnota určuje tuto hodnotu, vrátí správce front následující informace o aplikaci, která vydala volání MQGET:

- Hodnota pole *DXCC* na výstupu z uživatelské procedury
- Hodnota pole *DXREA* na výstupu z uživatelské procedury
- Hodnota pole *DXLEN* na *vstupu* pro ukončení
- Obsah vstupní vyrovnávací paměti uživatelské procedury *INBUF*. Počet vrácených bajtů je zadán parametrem **INLEN** .

Pokud byla ukončena uživatelská procedura *INBUF*, výsledky nejsou definovány.

DXRES je výstupní pole z uživatelské procedury.

DXSID (čtyřbajtový znakový řetězec)

Identifikátor struktury.

Hodnota musí být:

DXSIDV
Identifikátor pro strukturu výstupního parametru konverze dat.

Toto je vstupní pole pro ukončení.

DXVER (10ciferné celé číslo se znaménkem)

Číslo verze struktury.

Hodnota musí být:

DXVER1
Číslo verze pro strukturu parametru výstupního bodu převodu dat.

Následující konstanta uvádí číslo verze aktuální verze:

DXVERC
Aktuální verze struktury parametru ukončení konverze dat.

Poznámka: Když je představena nová verze této struktury, rozvržení existující součásti se nezmění. Uživatelská procedura by proto měla zkontrolovat, zda je pole *DXVER* rovno nebo větší než nejnižší verze, která obsahuje pole, která má uživatelská procedura použít.

Toto je vstupní pole pro ukončení.

DXXOP (10ciferné celé číslo se znaménkem)

Vyhrazeno.

Jedná se o vyhrazené pole; jeho hodnota je 0.

Deklarace RPG (kopie souboru 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 (Konverze znaků) na IBM i

Volání MQXCNVC převádí znaky z jedné znakové sady do jiné.

Toto volání je součástí produktu IBM MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce produktu IBM MQ . Poznámka: Toto volání lze použít pouze z uživatelské procedury pro převod dat.

- “Syntaxe” na stránce 1433
- “Parametry” na stránce 1433
- “Vyvolání RPG (ILE)” na stránce 1437

Syntaxe

```
MQXCNVC HCONN, OPTS, SRCCSI, SRCLEN, SRCBUF, TGTCSI, TGTLEN,
   TGTBUF, DATLEN, CMPCOD, REASON)
```

Parametry

Volání MQXCNVC má následující parametry:

HCONN (desetimístné podepsané celé číslo)-vstup

Manipulátor připojení.

Tento manipulátor představuje připojení ke správci front. Normálně by měl být manipulační prostředek předávaný proceduře pro převod dat v poli DXHCN struktury MQDXP. Tento manipulátor nemusí být nutně stejný jako popisovač určený aplikací, která vydala volání MQGET.

V systému IBM ilze pro produkt HCONNzadat následující speciální hodnotu:

HCDEFH
Výchozí popisovač připojení.

OPTS (10ciferné celé číslo se znaménkem)-vstup

Volby, které řídí akci MQXCNVC.

Může být uvedena nula nebo více voleb popsaných dále v této sekci. Je-li vyžadováno více než jedno, lze tyto hodnoty přidat (nepřidávat stejnou konstantu víckrát než jednou).

Volba Výchozí převod: Tato volba určuje použití výchozí konverze znaků:

DEFINICE DCCDEF
Výchozí převod.

Tato volba uvádí, že lze použít výchozí převod znaků, pokud jedna nebo obě znakové sady určené ve volání nejsou podporovány. To umožňuje správci front použít výchozí znakovou sadu určenou pro instalaci, která se bude při převodu řetězce přibližovat zadané znakové sadě.

Poznámka: Výsledkem použití přibližné znakové sady pro převod řetězce je to, že některé znaky mohou být nesprávně převedeny. Tomu lze zabránit tak, že použijete v řetězci pouze znaky, které jsou společné jak pro uvedenou znakovou sadu, tak pro výchozí znakovou sadu.

Výchozí znakové sady jsou definovány volbou konfigurace, když je správce front instalován nebo restartován.

Není-li hodnota DCCDEF zadána, správce front použije k převodu řetězce pouze zadané znakové sady a volání selže, pokud jedna nebo obě znakové sady nejsou podporovány.

Volba vyplnění: Následující volba umožňuje správci front vyplnění převedeného řetězce s mezerami nebo zahodit nevýznamné koncové znaky za účelem převedení převedeného řetězce na cílovou vyrovnávací paměť:

DCFIL
Vyplnit cílovou vyrovnávací paměť.

Tato volba vyžaduje provedení převodu takovým způsobem, aby byla cílová vyrovnávací paměť zcela vyplněna:

- Jsou-li při převodu zadány koncové mezery, jsou za účelem vyplnění cílové vyrovnávací paměti přidány koncové mezery.
- Pokud se řetězec při převodu rozvine, koncové znaky, které nejsou významné, budou vyřazeny, aby převedený řetězec vešel do cílové vyrovnávací paměti. Je-li to možné provést úspěšně, volání skončí s CCOK a kódem příčiny RCNONE.

  Pokud existuje příliš málo nevýznamných koncových znaků, tak velká část řetězce, jak se vejde, se umístí do cílové vyrovnávací paměti a volání skončí s CCWARN a kódem příčiny RC2120.

  Nevýznamné znaky jsou:

- Koncové mezery
- Znaky následující za prvním znakem null v řetězci (ale kromě prvního znaku null samotného)

- Pokud je řetězec, TGTCSI a TGTLEN takový, že cílová vyrovnávací paměť nemůže být nastavena úplně s platnými znaky, volání selže s CCFAIL a s kódem příčiny RC2144. K tomu může dojít, když TGTCSI je čistá DBCS znaková sada (jako je UTF-16), ale TGTLEN uvádí délku, která je lichým počtem bajtů.
- TGTLEN může být menší než nebo větší než SRCLEN. Při návratu z MQXCNVC má DATLEN stejnou hodnotu jako TGTLEN.

Není-li tato volba zadána, postupujte takto:

- Řetězec se může podle potřeby ve vyrovnávací paměti podle potřeby uzavírat nebo rozšiřovat v rámci cílové vyrovnávací paměti. Nevýznamné koncové znaky nejsou přidány nebo zrušeny.

  Pokud se převedený řetězec vejde do cílové vyrovnávací paměti, je volání dokončeno s CCOK a kódem příčiny RCNONE.

  Je-li převedený řetězec příliš velký pro cílovou vyrovnávací paměť, tolik znaků, kolik se vejde do cílové vyrovnávací paměti, a volání bude dokončeno s CCWARN a kódem příčiny RC2120. Všimněte si, že v tomto případě může být vráceno méně než *TGTLEN* bajtů.
- TGTLEN může být menší než nebo větší než SRCLEN. Při návratu z MQXCNVC je DATLEN menší než nebo rovno TGTLEN.

Volby kódování: Tyto volby lze použít k uvedení celočíselných kódování zdrojového a cílového řetězce. Relevantní kódování je použito pouze v případě, že odpovídající identifikátor znakové sady označuje, že znázornění znakové sady v hlavní paměti je závislé na kódování použité pro binární celá čísla. Toto se týká pouze určitých vícebajtových znakových sad (například znakových sad UTF-16).

Kódování je ignorováno, pokud znaková sada je jednobajtová znaková sada (SBCS), nebo vícebajtová znaková sada s reprezentací v hlavní paměti, která není závislá na celočíselném kódování.

Měla by být uvedena pouze jedna z hodnot DCCS*, kombinovaná 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 . Je-li však zdrojové nebo cílové kódování získáno z pole MDENC v produktu MQMD nebo v jiné struktuře, je třeba provést následující zpracování:

1. Celočíselné kódování musí být extrahováno z pole MDENC odstraněním plovoucího a packed-decimálního kódování; podrobnosti o tom, jak to provést, viz "Analýza kódování v systému IBM i" na stránce 1417 .
2. Celočíselné kódování, které je výsledkem kroku 1, musí být vynásobeno příslušným faktorem, než bude přidáno do pole OPTS . Jedná se o následující faktory:

DCCSFACITY
Faktor kódování zdroje

DCCTFACITY
Faktor cílového kódování

Není-li tento parametr zadán, bude výchozí hodnota kódování nastavena na nedefinované (DCC* UN). Ve většině případů to nemá vliv na úspěšné dokončení volání MQXCNVC. Je-li však odpovídající znaková sada vícebajtová znaková sada se znázorněním, která je závislá na kódování (například znaková sada UTF-16), volání selže s kódem příčiny RC2112 nebo RC2116 , jak je to vhodné.

Výchozí volba: Pokud žádná z výše popsaných voleb není uvedena, lze použít následující volbu:

DCCNON
Nejsou uvedeny žádné volby.

DCCNON je definován v dokumentaci programu pomoci. Není určeno, že by tato volba byla použita s jinou, ale její hodnotou je nula, takové použití nelze detekovat.

SRCCSI (10ciferné celé číslo se znaménkem)-vstup

Identifikátor kódované znakové sady řetězce před převodem.

Jedná se o identifikátor kódované znakové sady vstupního řetězce v SRCBUF.

SRCLEN (10číslicové podepsané celé číslo)-vstup

Délka řetězce před převodem.

Délka vstupního řetězce v SRCBUF je délka (v bajtech); musí být nula nebo větší.

SRCBUF (jednobajtový 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 (10ciferné celé číslo se znaménkem)-vstup

Identifikátor kódované znakové sady řetězce po převodu.

Jedná se o identifikátor kódované znakové sady znakové sady, do níž má být produkt SRCBUF převeden.

TGTLEN (10ciferné celé číslo se znaménkem)-vstup

Délka výstupní vyrovnávací paměti.

Toto je délka výstupní vyrovnávací paměti TGTBUF , v bajtech; musí být nula nebo větší. Může být menší než nebo větší než SRCLEN.

TGTBUF (jednobajtový znakový řetězec x TGTLEN)-výstup

Řetězec po převodu.

To je řetězec poté, co byl převeden na znakovou sadu definovanou TGTCSI. Konvertovaný řetězec může být kratší nebo delší než nekonvertovaný řetězec. Argument **DATLEN** udává počet platných bajtů, které byly vráceny.

DATLEN (10ciferné celé číslo se znaménkem)-výstup

Délka výstupního řetězce.

Jedná se o délku řetězce vráceného ve výstupní vyrovnávací paměti TGTBUF. Konvertovaný řetězec může být kratší nebo delší než nekonvertovaný řetězec.

CMPCOD (10ciferné celé číslo se znaménkem)-výstup

Kód dokončení.

Jedná se o jednu z následujících položek:

KEK
Úspěšné dokončení.

CCWARN
Varování (částečné dokončení).

CCFIL
Volání se nezdařilo.

REASON (10ciferné celé číslo)-výstup

Kód příčiny kvalifikující CMPCOD.

Pokud má parametr CMPCOD hodnotu CCOK:

RCNONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li CMPCOD CCWARN:

RC2120
(2120, X'848 ') Konvertovaná data jsou příliš velká pro vyrovnávací paměť.

Je-li CMPCOD CCFAIL:

RC2010
(2010, X'7DA') Parametr délky dat není platný.

RC2150
(2150, X'866 ') DBCS řetězec není platný.

RC2018
(2018, X'7E2') Popisovač připojení není platný.

RC2046
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

RC2102
(2102, X'836 ') Není k dispozici dostatek 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 celého čísla zdroje nebylo rozpoznáno.

RC2143
(2143, X'85F') Parametr délky zdroje není platný.

RC2071
(2071, X'817 ') Není k dispozici dostatek paměti.

RC2146
(2146, X'862 ') Cílový parametr vyrovnávací paměti není platný.

RC2115
(2115, X'843 ') Identifikátor cílové kódované znakové sady není platný.

RC2116
(2116, X'844 ') Cílové celé číslo kódování 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 najdete v tématu “Návratové kódy pro IBM i (ILE RPG)” na stránce 1411.

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 (Ukončení převodu dat) v systému IBM i

Tato definice volání popisuje parametry, které jsou předány uživatelské proceduře pro převod dat.

Žádný vstupní bod s názvem MQCONVX není poskytován správcem front (viz poznámka o použití “11” na stránce 1439).

Tato definice je součástí produktu IBM MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce produktu 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, kterou provádí uživatelská procedura pro převod dat, je definována poskytovatelem uživatelské procedury, avšak tato procedura musí odpovídat pravidlům, která jsou zde popsána, a v přidružené struktuře parametrů MQDXP.

   Programovací jazyky, které lze použít pro ukončení převodu dat, jsou určovány prostředím.
2. Ukončení je vyvoláno pouze v případě, že *all* z následujících tvrzení je pravdivé:
   - Volba GMCONV je zadána na volání MQGET

- Pole *MDFMT* v deskriptoru zprávy nemá hodnotu FMNONE.
- Zpráva ještě není v požadované reprezentaci. To znamená, že jedna nebo obě zprávy *MDCSI* a *MDENC* se liší od hodnoty zadané aplikací v deskriptoru zpráv dodaném při volání MQGET.
- Správce front dosud neprovedl převod úspěšně.
- Délka vyrovnávací paměti aplikace je větší než nula.
- Délka dat zprávy je větší než nula.
- Kód příčiny tak daleko během operace MQGET je RCNONE nebo RC2079 .

3. Při zápisu uživatelské procedury by mělo být zváženo kódování uživatelské procedury způsobem, který umožní převod zpráv, které byly oříznuty. Zkrácené zprávy mohou nastat následujícími způsoby:

   - Přijímající aplikace poskytuje vyrovnávací paměť, která je menší než zpráva, ale určuje volbu GMATM na volání MQGET.

     V tomto případě bude mít pole *DXREA* v parametru **MQDXP** na vstupu do výstupu hodnotu RC2079.
   - Odesílatel zprávy jej zkrátil, než jej odešle. Tato situace může nastat například u zpráv sestavy (další podrobnosti viz "Převod zpráv sestav v systému IBM i" na stránce 1426).

     V tomto případě bude mít pole *DXREA* v parametru **MQDXP** na vstupu do výstupu hodnotu RCNONE (pokud přijímající aplikace poskytla vyrovnávací paměť, která byla dostatečně velká pro tuto zprávu).

   Tudíž hodnota pole *DXREA* na vstupu do ukončení nemůže být vždy použita k rozhodnutí, zda byla zpráva zkrácena.

   Charakteristickým znakem oříznuté zprávy je, že délka poskytnutá uživatelské proceduře v argumentu **INLEN** bude *menší než* délka zahrnutá v názvu formátu, který je obsažen v poli *MDFMT* v deskriptoru zpráv. Ukončení by proto mělo zkontrolovat hodnotu *INLEN* před tím, než se pokusíte převést jakákoli data; uživatelská procedura *by neměla* předpokládat, že bylo poskytnuto úplné množství dat, které je odvozeno od názvu formátu.

   Pokud uživatelská procedura nebyla zapsána pro převod oříznutých zpráv a **INLEN** je menší než očekávaná hodnota, výstup by měl v poli *DXRES* parametru **MQDXP** vrátit hodnotu XRFAIL, s polem *DXCC* nastaveným na hodnotu CCWARN a polem *DXREA* nastaveným na hodnotu RC2110.

   Pokud byla uživatelská procedura *zapsána* pro převod zkrácených zpráv, měla by se uživatelská procedura převést co nejvíce dat (viz další poznámka o použití), přičemž se nezajímají o pokus o prozkoumání nebo konverzi dat za koncem *INBUF*. Je-li konverze úspěšně dokončena, výstupní ukončení by mělo ponechat pole *DXREA* v parametru **MQDXP** nezměněno. Tento příkaz vrátí hodnotu RC2079 , pokud byla zpráva oseknuta správcem front příjemce, a hodnota RCNONE, pokud byla zpráva zkrácena odesílatelem zprávy.

   Je také možné, aby zpráva rozbalila *během* převodu, na místo, kde je větší než *OUTBUF*. V tomto případě se musí výstup rozhodnout, zda má být zpráva zkráceny; pole *DXAOP* v parametru **MQDXP** bude indikovat, zda přijímající aplikace specifikovala volbu GMATM.

4. Obecně se doporučuje, aby byla převedena všechna data ve zprávě poskytnuté k ukončení v produktu *INBUF* , nebo že žádná z nich není. Výjimka však nastane, pokud je zpráva zkrácena, buď před převodem, nebo během převodu; v tomto případě může být na konci vyrovnávací paměti nekompletní položka (například: jeden bajt dvoubajtového znaku, nebo 3 bajty 4bajtové celé číslo). V této situaci se doporučuje vynechávat neúplnou položku a nepoužité bajty v sadě *OUTBUF* nastavené na hodnotu null. Avšak úplné prvky nebo znaky v poli nebo řetězci *by měly* být převedeny.
5. Když je poprvé ukončena uživatelská procedura, správce front se pokusí 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 obsahuje uživatelskou proceduru, měly být identické, ačkoli ne všechna prostředí vyžadují toto.
6. A new copy of the exit is loaded when an application attempts to retrieve the first message that uses that *MDFMT* since the application connected to the queue manager. Nová kopie může být také načtena někdy, pokud správce front zahodil dříve načtenou kopii. Z tohoto důvodu by se procedura

neměla pokoušet o použití statického úložiště pro sdělování informací z jednoho vyvolání procedury do dalšího-může být uvolněno mezi oběma vyvoláními.

7. Existuje-li uživatelská procedura se stejným názvem jako jeden z vestavěných formátů podporovaných správcem front, uživatelská procedura nemůže nahradit vestavěnou rutinu převodu. Pouze okolnosti, za kterých je taková východa, jsou:
   - If the built-in conversion routine cannot handle conversions to or from either the *MDCSI* or *MDENC* involved, or
   - Pokud vestavěná rutina převodu selhala při převodu dat (například proto, že existuje pole nebo znak, který nelze převést).
8. Rozsah ukončení je závislý na prostředí. Názvy *MDFMT* by měly být vybrány tak, aby se minimalizovalo riziko konfliktů s jinými formáty. Doporučuje se, aby začínali znaky, které identifikují aplikaci definující název formátu.
9. Ukončení převodu dat se spouští v prostředí, jako je tomu u programu, který vydal volání MQGET; prostředí zahrnuje adresní prostor a profil uživatele (kde je to vhodné). Program může být agent kanálu zpráv odesílající zprávy do cílového správce front, který nepodporuje převod zpráv. Uživatelská procedura nemůže ohrozit integritu správce front, protože není spuštěna v prostředí správce front.
10. Jediné volání MQI, které lze použít při ukončení, je MQXCNVC; pokus o použití jiných volání MQI selže s kódem příčiny RC2219nebo s jinými nepředvídatelnými chybami.
11. Správcem front není poskytnut žádný vstupní bod s názvem MQCONVX. Název uživatelské procedury by měl být stejný jako název formátu (název obsažený v poli *MDFMT* v produktu MQMD), ačkoli tento název není povinný ve všech prostředích.

Parametry

Volání MQCONVX má následující parametry:

MQDXP (MQDXP)-vstup/výstup

Blok parametru ukončení konverze dat.

Tato struktura obsahuje informace vztahující se k vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby označovaly výsledek převodu. Podrobnosti o polích v této struktuře viz “MQDXP (parametr uživatelské procedury konverze dat) v systému IBM i” na stránce 1427 .

MQMD (MQMD)-vstup/výstup

Deskriptor zpráv.

Při vstupu do uživatelské procedury se jedná o deskriptor zprávy, který by byl vrácen aplikaci, pokud nebyla provedena žádná konverze. Obsahuje tedy *MDFMT*, *MDENC*a *MDCSI* nepřevedené zprávy obsažené v *INBUF*.

Poznámka: Parametr **MQMD** předaný do uživatelské procedury je vždy nejnovější verzí MQMD, kterou podporuje správce front, který vyvolá ukončení. Pokud má být uživatelská procedura přenositelná mezi různými prostředími, měla by uživatelská procedura zkontrolovat pole *MDVER* v produktu *MQMD* a ověřit, zda jsou pole, která uživatelská procedura potřebuje k přístupu, přítomna ve struktuře.

V systému IBM ije výstup předáván version-2 MQMD.

Výstup na výstupu by měl změnit pole *MDENC* a *MDCSI* na hodnoty požadované aplikací, pokud byl převod úspěšný; tyto změny se odrazí zpět do aplikace. Všechny ostatní změny, které má uživatelská procedura ke struktuře, se budou ignorovat. Neodrážejí se to zpět do aplikace.

Pokud uživatelská procedura vrátí parametr XROK v poli *DXRES* struktury MQDXP, ale nezmění pole *MDENC* nebo *MDCSI* v deskriptoru zpráv, správce front vrátí pro tato pole hodnoty, které měly odpovídající pole ve struktuře MQDXP na vstupu do uživatelské procedury.

INLEN (10ciferné celé číslo se znaménkem)-vstup

Délka v bajtech *INBUF*.

Toto je délka vstupní vyrovnávací paměti *INBUF*a uvádí počet bajtů, které mají být zpracovány uživatelskou procedurou. *INLEN* je menší z hodnot délky dat zprávy před převodem a délka vyrovnávací paměti poskytnutá aplikací na volání MQGET.

Hodnota je vždy větší než nula.

INBUF (jednobajtový bitový řetězec x INLEN)-vstup

Vyrovnávací paměť obsahující nepřevedené zprávy.

Obsahuje data zprávy před převodem. Pokud uživatelská procedura nemůže převést data, vrátí správce front obsah této vyrovnávací paměti do aplikace po dokončení uživatelské procedury.

Poznámka: Uživatelská procedura by neměla měnit *INBUF* ; je-li tento parametr změněn, nejsou výsledky definovány.

OUTLEN (10ciferné celé číslo se znaménkem)-vstup

Délka v bajtech *OUTBUF*.

Jedná se o délku výstupní vyrovnávací paměti *OUTBUF*a je stejná jako délka vyrovnávací paměti poskytnutá aplikací na volání MQGET.

Hodnota je vždy větší než nula.

OUTBUF (1-bajtový bitový řetězec x OUTLEN)-výstup

Vyrovnávací paměť obsahující převedenou zprávu.

Na výstupu z uživatelské procedury, pokud byl převod úspěšný (jak uvádí hodnota XROK v poli *DXRES* parametru **MQDXP**), obsahuje **OUTBUF** data zprávy, která mají být doručena aplikaci, v požadovaném znázornění. Pokud byl převod neúspěšný, budou všechny změny provedené v této vyrovnávací paměti 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
```

Konec programovacího rozhraní s citlivým produktem

Uživatelské procedury, uživatelské procedury rozhraní API a odkazy na instalovatelné služby

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

- “Struktura MQIEP” na stránce 1441

- “Odkaz na výstupní bod pro převod dat” na stránce 1444
- “MQ_PUBLISH_EXIT-Uživatelská procedura publikování” na stránce 1448
- “Volání uživatelských procedur kanálů a datové struktury” na stránce 1456
- “Popis uživatelské procedury rozhraní” na stránce 1545
- “Referenční informace o rozhraní instalovatelných služeb” na stránce 1606

Související pojmy
Uživatelské procedury, uživatelské procedury rozhraní API a instalovatelné služby produktu IBM MQ
Související úlohy
Rozšíření zařízení správce front

Struktura MQIEP

Struktura MQIEP obsahuje vstupní bod pro každé volání funkce, které je povoleno provést.

Pole

StrucId
Typ: MQCHAR4 -Vstup

Identifikátor struktury. Hodnota je následující:

ID_KONSTRUKCE_MQIEP_

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQIEP_VERSION_1
Číslo verze struktury verze 1.

AKTUÁLNÍ_VERZE MQIEP_
Aktuální verze struktury.

StrucLength
Typ: MQLONG

Velikost struktury MQIEP v bajtech. Hodnota je následující:

MQIEP_LENGTH_1

Příznaky
Typ: MQLONG

Poskytuje informace o adresách funkcí. Příznak, který označuje, zda je knihovna vláknem, může být použit s parametrem, aby indikoval, zda je knihovna klientem nebo serverovou knihovnou.

Následující hodnota se používá k uvedení žádných informací o knihovně:

MQIEPF_NONE

Jedna z následujících hodnot se používá k určení, zda je sdílená knihovna vláknová nebo nevláknová:

KNIHOVNU MQIEPF_NON_THREADED_LIBRARY
sdílená knihovna bez podprocesů

MQIEPF_THREADED_LIBRARY, KNIHOVNA
Sdílená knihovna s podporou podprocesů

Jedna z následujících hodnot se používá k určení, zda je sdílená knihovna klientem nebo sdílenou knihovnou serveru:

KNIHOVNA MQIEPF_CLIENT_LIBRARY
Klientská sdílená knihovna

KNIHOVNA MQIEPF_LOCAL_LIBRARY
Sdílená knihovna serveru

Vyhrazené
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_
Typ: PMQ_CLOSE_CALL

Adresa volání MQCLOSE.

Volání MQCMIT_Call
Typ: PMQ_CMIT_CALL

Adresa volání MQCMIT.

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.

MQDISC_Volat
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.

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.

MQMHBUF_Call
Typ: PMQ_MHBUF_CALL

Adresa volání MQMHBUF.

Funkce MQOPEN_Call
Typ: PMQ_OPEN_CALL

Adresa volání MQOPEN.

MQPUT_Call
Typ: PMQ_PUT_CALL

Adresa volání MQPUT.

MQPUT1_Call
Typ: PMQ_PUT1_CALL

Adresa volání MQPUT1 .

MQSET_Volání
Typ: PMQ_SET_CALL

Adresa volání MQSET.

Volání MQSETMP_Call
Typ: PMQ_SETMP_CALL

Adresa volání MQSETMP.

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: PMQ_ZEP_CALL

Adresa volání MQZEP.

Deklarace C

```
struct tagMQIEP {
   MQCHAR4          StrucId;          /* Structure identifier */
   MQLONG           Version;          /* Structure version number */
   MQLONG           StrucLength;      /* Structure length */
   MQLONG           Flags;            /* Flags */
   MQPTR            Reserved;         /* Reserved */
   PMQ_BACK_CALL    MQBACK_Call;      /* Address of MQBACK */
   PMQ_BEGIN_CALL   MQBEGIN_Call;     /* Address of MQBEGIN */
   PMQ_BUFMH_CALL   MQBUFMH_Call;     /* Address of MQBUFMH */
   PMQ_CB_CALL      MQCB_Call;        /* Address of MQCB */
   PMQ_CLOSE_CALL   MQCLOSE_Call;     /* Address of MQCLOSE */
   PMQ_CMIT_CALL    MQCMIT_Call;      /* Address of MQCMIT */
   PMQ_CONN_CALL    MQCONN_Call;      /* Address of MQCONN */
   PMQ_CONNX_CALL   MQCONNX_Call;     /* Address of MQCONNX */
   PMQ_CRTMH_CALL   MQCRTMH_Call;     /* Address of MQCRTMH */
   PMQ_CTL_CALL     MQCTL_Call;       /* Address of MQCTL */
   PMQ_DISC_CALL    MQDISC_Call;      /* Address of MQDISC */
   PMQ_DLTMH_CALL   MQDLTMH_Call;     /* Address of MQDLTMH */
   PMQ_DLTMP_CALL   MQDLTMP_Call;     /* Address of MQDLTMP */
   PMQ_GET_CALL     MQGET_Call;       /* Address of MQGET */
   PMQ_INQ_CALL     MQINQ_Call;       /* Address of MQINQ */
   PMQ_INQMP_CALL   MQINQMP_Call;     /* Address of MQINQMP */
   PMQ_MHBUF_CALL   MQMHBUF_Call;     /* Address of MQMHBUF */
   PMQ_OPEN_CALL    MQOPEN_Call;      /* Address of MQOPEN */
   PMQ_PUT_CALL     MQPUT_Call;       /* Address of MQPUT */
   PMQ_PUT1_CALL    MQPUT1_Call;      /* Address of MQPUT1 */
   PMQ_SET_CALL     MQSET_Call;       /* Address of MQSET */
   PMQ_SETMP_CALL   MQSETMP_Call;     /* Address of MQSETMP */
   PMQ_STAT_CALL    MQSTAT_Call;      /* Address of MQSTAT */
   PMQ_SUB_CALL     MQSUB_Call;       /* Address of MQSUB */
   PMQ_SUBRQ_CALL   MQSUBRQ_Call;     /* Address of MQSUBRQ */
   PMQ_XCLWLN_CALL  MQXCLWLN_Call;    /* Address of MQXCLWLN */
   PMQ_XCNVC_CALL   MQXCNVC_Call;     /* Address of MQXCNVC */
   PMQ_XDX_CALL     MQXDX_Call;       /* Address of MQXDX */
   PMQ_XEP_CALL     MQXEP_Call;       /* Address of MQXEP */
   PMQ_ZEP_CALL     MQZEP_Call;       /* Address of MQZEP */
};
```

Odkaz na výstupní bod pro převod dat

Pro z/OSmusíte zapsat ukončení převodu dat v jazyce assembler. U jiných platforem se doporučuje používat programovací jazyk C.

Chcete-li vám pomoci vytvořit výstupní program pro převod dat, jsou dodány následující prostředky:

- Zdrojový soubor kostry
- Volání konverze znaků
- Obslužný program, který vytváří fragment kódu, který provádí převod dat na strukturách datových typů. Tento obslužný program bere pouze vstupní data jazyka C. V systému z/OSvytváří kód assembler.

Postup při psaní programů je uveden v následujících tématech:

- IBM i Psaní ukončovacího programu pro převod dat pro produkt IBM MQ for IBM i
- z/OS Psaní ukončovacího programu pro převod dat pro produkt IBM MQ for z/OS

- Psaní uživatelské procedury pro převod dat pro systémy IBM MQ for AIX or Linux
- Psaní uživatelské procedury pro převod dat pro produkt IBM MQ for Windows

Zdrojový soubor skeletonu

Ty mohou být použity jako výchozí bod při zápisu ukončovacího programu pro převod dat.

Dodané soubory jsou vypsány v Tabulka 817 na stránce 1445.

Tabulka 817. Zdrojové soubory skeletonu

Platforma	**Soubor**
AIX	amqsvfc0.c
IBM i	QMQMSAMP/QCSRC (AMQSVFC4)
Linux	amqsvfc0.c
Systémy Windows	amqsvfc0.c
z/OS	CSQ4BAX8 (“1” na stránce 1445) CSQ4BAX9 (“2” na stránce 1445) CSQ4CAX9 (“3” na stránce 1445)

Notes:

1. Znázorňuje volání MQXCVNC.
2. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití ve všech prostředích kromě CICS.
3. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití v prostředí produktu CICS .

Konvertovat volání znaků

Chcete-li převést data znakových zpráv z jednoho znaku na jiný, použijte volání MQXCNVC (konvertovat znaky) z ukončovacího programu pro převod dat. Pro určité vícebajtové znakové sady (například ve znakových sadách UTF-16) se musí použít příslušné volby.

V rámci uživatelské procedury nelze provést žádná jiná volání MQI. Pokus o provedení takové volání selže s kódem příčiny MQRC_CALL_IN_PROGRESS.

Další informace o volání MQXCNVC a příslušných volbách viz “MQXCNVC-Převod znaků” na stránce 909 .

Obslužný program pro vytvoření kódu ukončení převodu

V této části se dozvíte více o vytváření kódu pro ukončení převodu.

Příkazy pro vytvoření kódu ukončení konverze jsou:

IBM i **IBM i**
CVTMQMDTA (Konverze datového typu IBM MQ)

ALW **Systémy AIX, Linux, and Windows**
`crtmqcvx` (Vytvoření převodu IBM MQ -ukončení)

z/OS **z/OS**
CSQUCVX

Příkaz pro vaši platformu vytvoří fragment kódu, který provádí konverzi dat na strukturách datových typů, pro použití ve vašem ukončovacím programu pro převod dat. Příkaz vezme soubor obsahující jednu nebo více definic struktury jazyka C. z/OS V z/OSpak vygeneruje datovou sadu obsahující fragmenty kódu assembler a konverzní funkce. Na jiných platformách vygeneruje soubor s funkcí jazyka C pro převod každé definice struktury. V systému z/OSvyžaduje obslužný program přístup ke knihovně běhového prostředí LE/370 SCEERUN.

Vyvolání obslužného programu CSQUCVX na systému z/OS

z/OS

Obrázek Obrázek 10 na stránce 1446 znázorňuje příklad kódu 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 kód JCL použitý k vyvolání obslužného programu CSQUCVX

Příkazy definice dat produktu z/OS

z/OS

Obslužný program CSQUCVX vyžaduje příkazy definice dat s následujícími názvy definic dat, které jsou uvedeny v souboru Tabulka 818 na stránce 1446:

Tabulka 818. Názvy a popisy příkazů definic dat

Příkaz DD	**Popis**
SYSPRINT	Uvádí datovou sadu nebo tisk třídy souběžného tisku pro zprávy a chybové zprávy.
CSQUINP	Uvádí rozdělenou datovou sadu obsahující definice datových struktur, které mají být převedeny.
CSQUOUT	Určuje rozdělenou datovou sadu, do níž mají být zapsány fragmenty kódu konverze. Délka logického záznamu (LRECL) musí být 80 a formát záznamu (RECFM) musí být FB.

Chybové zprávy v systémech AIX, Linux, and Windows

Příkaz `crtmqcvx` vrací zprávy v rozsahu AMQ7953 až AMQ7970.

Tyto zprávy jsou uvedeny v seznamu Zprávy a kódy příčiny *IBM MQ Zprávy*.

Existují dva hlavní typy chyb:

- Závažné chyby, jako jsou syntaktické chyby, při 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 analýza struktury může pokračovat.

  Výstupní soubor byl vytvořen a obsahuje informace o chybě týkající se problémů, které se vyskytly. Tyto informace o chybě jsou uvedeny předponou `#error` , takže vytvořený kód není žádným kompilátorem akceptován bez nutnosti zásahu k nápravě problémů.

Platná syntaxe

Váš vstupní soubor pro obslužný program musí odpovídat syntaxi jazyka C.

Pokud nejste obeznámeni s C, prostudujte si téma Příklad kódu C v tomto tématu.

Kromě toho si buďte vědomi následujících pravidel:

- typedef je rozpoznáván pouze před klíčovým slovem struct.
- Ve vašich deklaracích struktury je vyžadována značka struktury.
- Prázdné hranaté závorky [] můžete použít k označení pole nebo řetězce proměnné délky na konci zprávy.
- Vícerozměrná pole a pole řetězců nejsou podporována.
- Jsou rozeznány následující další datové typy:
  - MQBOOL
  - MQBYTE
  - MQCHAR
  - MQFLOAT32
  - MQFLOAT64
  - MQSHORT
  - MQLONG
  - MQINT8
  - MQUINT8
  - MQINT16
  - MQUINT16
  - MQINT32
  - MQUINT32
  - MQINT64
  - MQUINT64

  Pole MQCHAR jsou převedena na kódovou stránku, ale MQBYTE, MQINT8 a MQUINT8 zůstanou beze změny. Je-li kódování odlišné, MQSHORT, MQLONG, MQINT16, MQUINT16, MQINT32, MQUINT32, MQINT64, MQUINT64, MQFLOAT32, MQFLOAT64 a MQBOOL se odpovídajícím způsobem konvertují.
- Nepoužívejte následující typy dat:
  - dvojitý
  - ukazatele
  - bitová pole

  Důvodem je to, že obslužný program pro vytvoření kódu ukončení převodu neposkytuje prostředek pro převod těchto datových typů. Chcete-li to překonat, můžete napsat své vlastní rutiny a volat je z uživatelské procedury.

Další poznámky:

- Nepoužívejte pořadová čísla ve vstupní datové sadě.
- Pokud existují pole, pro která chcete poskytnout vlastní převodní rutiny, deklarujte je jako MQBYTE a pak nahraďte vygenerovaná makra CMQXCFBA vlastním konverzním kódem.

Příklad příkazu 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 pro 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 pozměňovat zprávy doručené odběratelům.

Účel

Pomocí uživatelské procedury pro publikování zkontrolujte a pozměňte zprávy doručené odběratelům:

- Prozkoumat obsah zprávy publikované pro každého odběratele
- Upravit obsah zprávy publikované pro každého odběratele
- Změnit frontu, do níž je zpráva vložena
- Zastavit doručení zprávy odběrateli

Tato uživatelská procedura není k dispozici na systému IBM MQ for z/OS.

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, který je příjemcem publikace.

MQPSXP-Struktura dat uživatelské procedury publikování

Struktura MQPSXP popisuje informace, které jsou předány a vráceny z uživatelské procedury publikování.

Tabulka 819 na stránce 1449 shrnuje pole ve struktuře:

Tabulka 819. Pole v MQPSXP

Pole	Popis
StrucID	Identifikátor struktury
Version	Číslo verze struktury
ExitId	Typ uživatelské procedury, která se volá
ExitReason	Důvod volání uživatelské procedury
ExitResponse	Odezva z uživatelské procedury
ExitResponse2	Sekundární odezva od ukončení
Feedback	Kód zpětné vazby
ExitUserArea	Uživatelská oblast pro ukončení
ExitData	Data uživatelské procedury
QMgrName	Název lokálního správce front
Hconn	Manipulátor připojení
MsgDescPtr	Adresa deskriptoru zpráv (MQMD)
MsgHandle	Popisovač pro vlastnosti 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 struktury 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 uživatelské procedury, která se volá. Hodnota je následující:

MQXT_PUBLISH_EXIT
Uživatelská procedura publikování.

ExitId je vstupní pole pro ukončení.

***ExitReason* (MQLONG)**
ExitReason je důvod volání ukončení. Možné hodnoty jsou:

MQXR_INIT
Ukončení pro toto připojení je voláno pro inicializaci. Ukončení může získat a inicializovat prostředky, které potřebuje; například hlavní paměť.

MQXR_TERM
Ukončení pro toto připojení je voláno, protože ukončení se chystá ukončit. Ukončení musí uvolnit všechny prostředky, které získal od doby, kdy byla inicializována; například hlavní paměť.

MQXR_PUBLICATION
Uživatelská procedura je volána správcem front předtím, než je publikace umístěna do fronty zpráv odběratele. Uživatelská procedura může změnit zprávu, nevložit zprávu do fronty nebo zastavit publikování.

ExitReason je vstupní pole pro ukončení.

***ExitResponse* (MQLONG)**
Chcete-li uvést, jak musí zpracování pokračovat, nastavte *ExitResponse* na výstupu. *ExitResponse* je jedna z následujících hodnot:

MQXCC_OK
Nastavte MQXCC_OK pro pokračování zpracování normálně. Nastavte MQXCC_OK jako odpověď na libovolné hodnoty ExitReason.

Má-li ExitReason hodnotu MQXR_PUBLICATION, pole *DestinationQName* a *DestinationQMgrName* struktury MQSBC identifikují místo určení, kam se zpráva odešle.

MQXCC_FAILED
Nastavte MQXCC_FAILED , abyste zastavili 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
Chcete-li zastavit normální zpracování zprávy, nastavte hodnotu MQXCC_SUPPRESS_FUNCTION . Only set MQXCC_SUPPRESS_FUNCTION if *ExitReason* has the value MQXR_PUBLICATION.

Zpráva bude dále zpracovávána správcem front podle volby MQRO_DISCARD_MSG v poli *Report* v deskriptoru zprávy příslušné zprávy.

- Je-li zadána volba MQRO_DISCARD_MSG , zpráva se nedoručí odběrateli.
- Není-li zadána volba MQRO_DISCARD_MSG , bude zpráva vložena do fronty nedoručených zpráv. Pokud neexistuje žádná fronta nedoručených zpráv nebo zprávu nelze úspěšně umístit do fronty nedoručených zpráv, nebude tato publikace doručena odběrateli. Doručení publikování ostatním odběratelům závisí na hodnotách atributů objektu tématu PMSGDLV a NPMSGDLV . Informace o těchto atributech najdete v popisech parametrů příkazu DEFINE TOPIC .

ExitResponse je výstupní pole z uživatelské procedury.

***ExitResponse2* (MQLONG)**
ExitResponse2 je vyhrazen pro budoucí použití.

***Feedback* (MQLONG)**
Feedback je kód zpětné vazby, který má být použit, pokud uživatelská procedura vrací MQXCC_SUPPRESS_FUNCTION v *ExitResponse*.

Na vstupu do uživatelské procedury má *Feedback* vždy hodnotu MQFB_NONE. Pokud uživatelská procedura vrátí hodnotu MQXCC_SUPPRESS_FUNCTION, nastavte hodnotu *Feedback* na hodnotu, která má být použita pro zprávu, když správce front umístí tuto hodnotu do fronty nedoručených zpráv. Pokud má *Feedback* původní hodnotu MQFB_NONE, nastaví správce front *Feedback* na MQFB_STOPPED_BY_PUBSUB_EXIT.

Feedback je vstupní/výstupní pole pro ukončení.

***ExitUserArea* (MQBYTE16)**
ExitUserArea je pole, které je k dispozici pro ukončení použití. Každé připojení má samostatný *ExitUserArea*. Délka *ExitUserArea* je dána MQ_EXIT_USER_AREA_LENGTH.

Pole *ExitReason* má hodnotu MQXR_INIT na prvním vyvolání uživatelské procedury. *ExitUserArea* se inicializuje na MQXUA_NONE při prvním vyvolání uživatelské procedury pro připojení. Následné změny v produktu *ExitUserArea* budou zachovány v rámci vyvolání uživatelské procedury.

ExitUserArea je vstupní/výstupní pole pro ukončení.

***ExitData* (MQCHAR32)**
ExitData je pevná výstupní data definovaná parametrem **PublishExitData** stanzy v inicializačním souboru správce front. Data jsou vyplněna mezerami na plnou délku pole. Pokud v inicializačním souboru nejsou definována žádná pevná výstupní data, *ExitData* je prázdný. Délka *ExitData* je dána MQ_EXIT_DATA_LENGTH.

ExitData je vstupní pole pro ukončení.

***QMgrName* (MQCHAR48)**
QMgrName je název lokálního správce front. Název je doplněn mezerami do plné délky pole. Délka tohoto pole je dána MQ_Q_MGR_NAME_LENGTH.

QMgrName je vstupní pole pro ukončení.

***Hconn* (MQHCONN)**
Hconn je manipulátor představující připojení ke správci front. Funkci *Hconn* lze použít pouze jako parametr pro volání funkcí vlastností zpráv MQSETMP, MQINQMMPnebo MQDLTMP .

Hconn je vstupní pole pro ukončení.

***MsgDescPtr* (PMQMD)**
MsgDescPtr je adresa deskriptoru zpráv (MQMD) zpracovávané zprávy a je kopií deskriptoru MQMD vráceného z volání MQPUT. Uživatelská procedura může změnit obsah deskriptoru zpráv. Jakákoli změna obsahu deskriptoru zpráv musí být provedena s opatrností. Zejména v případě, že pole *SubType* struktury MQSBC má hodnotu MQSUBTYPE_PROXY, nesmí být pole *CorrelId* v deskriptoru zpráv změněno.

Do uživatelské procedury není předán žádný deskriptor zprávy, pokud *ExitReason* je MQXR_INIT nebo MQXR_TERM ; v těchto případech je *MsgDescPtr* ukazatelem null.

MsgDescPtr je vstupní pole pro ukončení.

***MsgHandle* (MQHMSG)**
MsgHandle je popisovač vlastností zprávy. K práci s vlastnostmi zprávy použijte pouze *MsgHandle* s vlastnostmi funkcí vlastností zprávy MQSETMP, MQINQMMPnebo MQDLTMP .

MsgHandle je vstupní pole pro ukončení.

***MsgInPtr* (PMQVOID)**
MsgInPtr je adresa vstupních dat zprávy. Obsah vyrovnávací paměti adresovaný produktem *MsgInPtr* může být upraven uživatelskou procedurou; viz *MsgOutPtr* .

MsgInPtr je vstupní pole pro ukončení.

***MsgInLength* (MQLONG)**
MsgInLength je délka dat zprávy předaných do ukončení v bajtech. Adresa dat je dána *MsgInPtr*.

MsgInLength je vstupní pole pro ukončení.

***MsgOutPtr* (PMQVOID)**
MsgOutPtr je adresa vyrovnávací paměti obsahující data zprávy, která se vrací z ukončení. Při vstupu do uživatelské procedury má *MsgOutPtr* hodnotu null. Při návratu z uživatelské procedury v případě, že je hodnota stále null, správce front odešle zprávu zadanou parametrem *MsgInPtr*s délkou zadanou argumentem *MsgInLength*.

Pokud uživatelská procedura modifikuje data zprávy, použijte jednu z následujících procedur:

- Pokud se délka dat nezmění, mohou být data upravena ve vyrovnávací paměti adresované pomocí *MsgInPtr*. V takovém případě neměňte *MsgOutPtr* a *MsgOutLength*.
- Jsou-li upravená data kratší než původní data, lze data upravit ve vyrovnávací paměti adresované pomocí *MsgInPtr*. V tomto případě musí být proměnná *MsgOutPtr* nastavena na adresu vstupní vyrovnávací paměti zpráv a *MsgOutLength* nastavena na novou délku dat zprávy.
- Pokud upravená data jsou nebo mohou být delší než původní data, musí uživatelská procedura získat novou vyrovnávací paměť zpráv. Zkopírujte do ní upravená data. Nastavte *MsgOutPtr* na adresu nové vyrovnávací paměti a nastavte *MsgOutLength* na délku nových dat zprávy. Ukončení je zodpovědné za uvolnění vyrovnávací paměti adresované serverem *MsgOutPtr* při příštím volání uživatelské procedury.

Poznámka: *MsgOutPtr* je vždy ukazatel null na vstupu do ukončení, a ne adresa dříve získané vyrovnávací paměti zpráv. Chcete-li uvolnit dříve získanou vyrovnávací paměť, musí uživatelská procedura uložit svou adresu a délku. Uložte informace buď v produktu *ExitUserArea*, nebo v řídicím bloku, který má svou adresu uloženou v produktu *ExitUserArea*.

MsgOutPtr je vstupní/výstupní pole pro ukončení.

***MsgOutLength* (MQLONG)**
MsgOutLength je délka dat zpráv vrácených uživatelskou procedurou v bajtech. Pro vstup na ukončení je toto pole vždy nula. Při návratu z uživatelské procedury je toto pole ignorováno, pokud má parametr *MsgOutPtr* hodnotu null. Informace o úpravách dat zprávy viz *MsgOutPtr* .

MsgOutLength je vstupní/výstupní pole pro ukončení.

***pEntryPoints* (PMQIEP)**
pEntryPoints je adresa struktury MQIEP, pomocí které 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-Struktura dat kontextu publikování

Struktura MQPBC obsahuje kontextové informace týkající se vydavatele publikování, který je předán uživatelské proceduře pro publikování.

Tabulka 820 na stránce 1453 shrnuje pole ve struktuře:

Tabulka 820. Pole v MQPBC	
Pole	**Popis**
StrucID	Identifikátor struktury
Version	Číslo verze struktury
PubTopicString	Publikační řetězec tématu
MsgDescPtr	Adresa deskriptoru zpráv (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 parametru 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, který má být publikován.

PubTopicString je vstupní pole pro ukončení.

***MsgDescPtr* (PMQMD)**
MsgDescPtr je adresa kopie deskriptoru zpráv (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-Struktura dat kontextu odběru

Struktura MQSBC obsahuje kontextové informace týkající se odběratele, který přijímá publikování, který je předán uživatelské proceduře publikování.

Tabulka 821 na stránce 1454 shrnuje pole ve struktuře:

Tabulka 821. Pole v MQSBC	
Pole	**Popis**
StrucID	Identifikátor struktury
Version	Číslo verze struktury
DestinationQMgrName	Název správce cílové fronty
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 a odběru

Pole

***StrucID* (MQCHAR4)**
Identifikátor struktury. Hodnota je následující:

MQSBC_STRUCID
MQSBC_STRUCID je identifikátor struktury parametru uživatelské procedury publikování. Pro programovací jazyk C je také definována 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 výstupního parametru 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 doplněn mezerami do plné délky pole. Název může být změněn ukončením. Délka tohoto pole je dána 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 doplněn mezerami do plné délky pole. Název může být změněn ukončením. Délka tohoto pole je dána 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 Dotaz na stav odběru (odezva).

SubType je vstupní pole pro ukončení.

***SubOptions* (MQLONG)**
SubOptions jsou volby odběru; viz “Volby (MQLONG)” na stránce 567 pro popis hodnot, které toto pole může provést.

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 MQ_TOPIC_NAME_LENGTH. Název objektu je název objektu administrativního tématu, který správce front přidružili k řetězci tématu. Even if the subscriber provided a topic object as part of the subscription, the *ObjectName* might be a different topic object. Přidružení objektu tématu s odběrem závisí na úplném vyřešení produktu *SubTopicString*.

ObjectName je vstupní pole pro ukončení.

***ObjectString* (MQCHARV)**
ObjectString je úplný řetězec tématu publikace, k jejímuž odběru jste přihlášeni. Všechny zástupné znaky v řetězci původního odběru jsou vyřešeny. Je to odlišné od pole *ObjectString* odběru MQSD popsaného v tématu “ObjectString (MQCHARV)” na stránce 576, které může obsahovat zástupné znaky a je výhradním předmětem názvu objektu poskytovaného odběratelem.

ObjectString je vstupní pole pro ukončení.

***SubTopicString* (MQCHARV)**
SubTopicString je úplný řetězec tématu, jak jej dodal odběratel. *SubTopicString* je kombinace řetězce tématu definovaného v objektu tématu a řetězce tématu. Odběratel musí poskytovat buď objekt tématu, řetězec tématu, nebo obojí. Poskytuje-li odběratel řetězec tématu, může obsahovat zástupné znaky.

SubTopicString je vstupní pole pro ukončení.

***SubName* (MQCHARV)**
SubName je název odběru, který je poskytnut buď odběratelem, nebo se jedná o vygenerovaný 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 jsou kritéria výběru použitá při přihlášení 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)” na stránce 579 .

SubLevel je vstupní pole pro ukončení.

***PSProperties* (MQLONG)**
PSProperties jsou vlastnosti publikování/odběru. Uurčují způsob, jakým jsou do zpráv odesílaných do tohoto odběru přidávány vlastnosti související s publikováním a odběry. Možné hodnoty jsou MQPSPROP_NONE, MQPSPROP_COMPAT, MQPSPROP_RFH2, MQPSPROP_MSGPROP. Popis těchto hodnot najdete v tématu Volitelné parametry (Změnit, Kopírovat a Vytvořit odběr) .

PSProperties je vstupní pole pro ukončení.

Poznámka: Kontroly autorizace jsou provedeny pouze na původních hodnotách *DestinationQMgrName* a *DestinationQName* před jejich předáním do uživatelské procedury pro publikování. Žádné nové

kontroly autorizace se neprovedou, když uživatelská procedura změní cílovou frontu, 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 uživatelských procedur kanálů a datové struktury

Tato kolekce témat obsahuje referenční informace o speciálních voláních IBM MQ a datových strukturách, které můžete použít při psaní programů uživatelské procedury kanálu.

Tyto informace jsou informace o programovacím rozhraní, které jsou citlivé na produkt. Uživatelské procedury produktu IBM MQ lze zapsat v následujících programovacích jazycích:

Tabulka 822. Uživatelské procedury produktu IBM MQ : platformy a programovací jazyky

Platforma	**Programovací jazyky**
IBM MQ for z/OS	Asembler a C (které se musí podřídit programovacím prostředím systému C pro ukončení systému, 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

Uživatelské procedury v produktu Java můžete také psát pouze pro použití s aplikacemi Java a JMS . Další informace o vytváření a používání kanálů pomocí konzoly IBM MQ classes for Javanajdete v tématu Použití uživatelských procedur kanálů v produktu IBM MQ classes for Java a pro produkt IBM MQ classes for JMSnaleznete v tématu Použití kanálů kanálů s produktem IBM MQ classes for JMS.

Uživatelské procedury IBM MQ nelze zapsat do TAL nebo Visual Basic. Deklarace struktury MQCD je však k dispozici ve Visual Basic pro použití v volání MQCONNX z programu IBM MQ MQI client .

V řadě případů v popisech, které následují, jsou parametry polí nebo znakových řetězců s velikostí, která není pevná. Pro tyto parametry se používá malá písmena "n" ke znázornění číselné konstanty. Je-li deklarace pro daný parametr kódována, musí být "n" nahrazena číselnou hodnotou, která je povinná. Další informace o konvencích použitých v těchto popisech naleznete v příručce “Elementární datové typy” na stránce 235.

Definiční soubory dat

Definiční soubory dat jsou dodávány s IBM MQ pro každý podporovaný programovací jazyk. Podrobné informace o těchto souborech najdete v tématu Kopírování, záhlaví, zahrnutí a soubory modulu.

MQ_CHANNEL_EXIT-Ukončení kanálu

Volání MQ_CHANNEL_EXIT popisuje parametry, které jsou předávány každému z uživatelských procedur kanálu volaných agentem Message Channel Agent.

Správcem front není poskytnut žádný vstupní bod s názvem MQ_CHANNEL_EXIT; název MQ_CHANNEL_EXIT nemá žádný speciální význam, protože názvy uživatelských procedur kanálu jsou uvedeny v definici kanálu MQCD.

Existuje pět typů uživatelské procedury kanálu:

- Ukončení zabezpečení kanálu
- Ukončení zprávy kanálu
- Uživatelská procedura odeslání kanálu
- Ukončení příjmu kanálu
- Zpráva kanálu-ukončení opakování

Parametry jsou podobné pro každý typ výstupu a popis uvedený zde se vztahuje na všechny tyto parametry, kromě 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.

Parametry ChannelExitParms (MQCXP)-vstupní/výstupní

Blok výstupních parametrů kanálu.

Tato struktura obsahuje další informace vztahující se k vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře tak, aby ukazovalo, jak bude agent MCA pokračovat.

ChannelDefinition (MQCD)-input/output

Definice kanálu.

Tato struktura obsahuje parametry nastavené administrátorem k řízení chování kanálu.

DataLength (MQLONG)-vstupní/výstupní

Délka dat.

Data závisí na typu uživatelské procedury:

- V případě uživatelské procedury zabezpečení kanálu je při vyvolání uživatelské procedury v poli *AgentBuffer* uvedena délka jakékoli zprávy zabezpečení, pokud *ExitReason* je MQXR_SEC_MSG. Je-li zde žádná zpráva, je nula. Výsadu musí toto pole nastavit na délku jakékoli zprávy zabezpečení, která má být odeslána partnerovi, pokud nastaví *ExitResponse* na MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_REQUEST_SEC_MSG. Data zprávy se nacházejí buď v produktu *AgentBuffer* , nebo v produktu *ExitBufferAddr*.

  Obsah bezpečnostních zpráv je výhradní odpovědností bezpečnostních východů.
- Pro uživatelskou proceduru zprávy kanálu je při vyvolání uživatelské procedury tento parametr obsahovat délku zprávy (včetně záhlaví přenosové fronty). Uživatelská procedura musí nastavit toto pole na délku zprávy buď v *AgentBuffer* , nebo v *ExitBufferAddr* , které má pokračovat. Tato hodnota musí být větší nebo rovna délce záhlaví přenosové fronty (MQXQH).
- Pro kanál odeslání nebo přijetí kanálu je při vyvolání uživatelské procedury tento parametr obsažen v tomto parametru, který obsahuje délku přenosu. Výstupem musí být toto pole nastaveno na délku přenosu buď v *AgentBuffer* , nebo v *ExitBufferAddr* , které má pokračovat.

Pokud uživatelská procedura zabezpečení odešle zprávu a neexistuje žádná uživatelská procedura zabezpečení na druhém konci kanálu, nebo druhý konec nastaví *ExitResponse* MQXCC_OK, bude zahajující uživatelská procedura znovu vyvolána s hodnotou MQXR_SEC_MSG a odezvou s hodnotou Null (*DataLength* = 0).

Délka AgentBufferLength (MQLONG)-input

Délka vyrovnávací paměti agenta.

Tento parametr může být větší než *DataLength* při vyvolání.

Pro zprávy kanálu, odeslání a přijetí ukončení může být nevyužitý prostor na vyvolání použit k rozbalení dat na místě. Je-li tomu tak, musí být parametr **DataLength** nastaven odpovídajícím způsobem uživatelskou procedurou.

V programovacím jazyku C se tento parametr předává prostřednictvím adresy.

AgentBuffer (MQBYTE x AgentBufferLength)-vstup/výstup

Vyrovnávací paměť agenta.

Obsah tohoto parametru závisí na typu ukončení:

- V případě uživatelské procedury zabezpečení kanálu je při vyvolání uživatelské procedury obsažena zpráva zabezpečení, je-li *ExitReason* MQXR_SEC_MSG. Chcete-li odeslat zprávu o zabezpečení zpět, může uživatelská procedura buď použít tuto vyrovnávací paměť, nebo její vlastní vyrovnávací paměť (*ExitBufferAddr*).
- Pro uživatelskou proceduru zprávy kanálu při vyvolání tohoto parametru tento parametr obsahuje:
  - Hlavička přenosové fronty (MQXQH), která obsahuje deskriptor zprávy (který sám obsahuje informace o kontextu pro zprávu), bezprostředně za následovaným
  - Data zprávy

  Pokud má zpráva pokračovat, může uživatelská procedura provést jednu z následujících možností:

  - Ponechat obsah vyrovnávací paměti beze změny
  - Upravte obsah na místě (vrací novou délku dat v produktu *DataLength* ; nesmí být větší než *AgentBufferLength*)
  - Zkopírujte obsah na server *ExitBufferAddr*a proveďte požadované změny.

  Žádné změny, které uživatelská procedura provede v záhlaví přenosové fronty, nebudou kontrolovány; nesprávné úpravy však mohou znamenat, že zprávu nelze umístit do cíle.
- U kanálu odesílání nebo příjmu kanálu je při vyvolání uživatelské procedury tato data obsažena v datech přenosu. Ukončení může provést jednu z následujících možností:
  - Ponechat obsah vyrovnávací paměti beze změny
  - Upravte obsah na místě (vrací novou délku dat v produktu *DataLength* ; nesmí být větší než *AgentBufferLength*)
  - Zkopírujte obsah na server *ExitBufferAddr*a proveďte požadované změny.

  První 8 bajtů dat nesmí být změněno uživatelskou procedurou.

ExitBufferLength (MQLONG)-vstupní/výstupní

Délka výstupní vyrovnávací paměti.

Při prvním vyvolání procedury je tento parametr nastaven na nulu. Poté, co bude při každém vyvolání předávána jakákoli hodnota zpět, je při každém vyvolání předána k ukončení další. Hodnota není použita agentem MCA.

Poznámka: Tento parametr nesmí být použit uživatelskými procedurami psanými v programovacích jazycích, které nepodporují datový typ ukazatele.

ExitBufferAdr (MQPTR)-vstupní/výstupní

Adresa vyrovnávací paměti uživatelské procedury.

Tento parametr je ukazatel na adresu vyrovnávací paměti úložiště spravované uživatelskou procedurou, kde může zvolit vrácení zprávy nebo přenosu dat (v závislosti na typu ukončení) agentovi, pokud je vyrovnávací paměť agenta, nebo nemusí být dostatečně velká, nebo pokud je vhodnější pro ukončení, aby to bylo možné provést.

Při prvním vyvolání uživatelské procedury je adresa předaná do uživatelské procedury null. Poté je jakákoli adresa předávána zpět při každém vyvolání, při každém vyvolání, která se při příštím vyvolání zobrazí.

Má-li parametr ExitBufferAdr hodnotu null, data použitá jsou převzata z parametru AgentBuffer .

Pokud parametr ExitBufferAddr nemá hodnotu null, data použitá jsou převzata z vyrovnávací paměti, na kterou ukazuje parametr ExitBufferAddr.

Poznámka: Tento parametr nesmí být použit uživatelskými procedurami psanými 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                              *
```

Vyvolání asembleru System/390

```
         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, která je prováděna uživatelskou procedurou kanálu, je definována poskytovatelem uživatelské procedury. Ukončení se však musí řídit pravidly definovanými zde a v přidruženém řídicím bloku, MQCXP.
2. Parametr **ChannelDefinition** předaný do uživatelské procedury kanálu může být jeden z několika verzí. Další informace naleznete v poli *Version* ve struktuře MQCD.
3. Pokud uživatelská procedura kanálu přijme strukturu MQCD s polem *Version* nastaveným na hodnotu větší než MQCD_VERSION_1, musí uživatelská procedura používat pole *ConnectionName* na MQCD, a to v předvolbách do pole *ShortConnectionName* .
4. Obecně platí, že ukončení kanálu je povoleno měnit délku dat zprávy. To může nastat jako důsledek ukončení přidání dat do zprávy nebo odebrání dat ze zprávy, nebo komprese nebo šifrování zprávy. Speciální omezení však platí, je-li zpráva segmentem, který obsahuje pouze část logické zprávy. Zejména nesmí existovat žádná čistá změna v délce zprávy jako výsledek akcí doplňujících se vysílacích a přijímacích východů.

   Například je přípustné pro ukončení odeslání ke zkrácení zprávy tím, že ji komprimuje, ale doplňková přijímací uživatelská procedura musí obnovit původní délku zprávy tím, že ji dekomprimuje, takže nedojde k žádné změně sítě v délce zprávy.

   Toto omezení vzniká, protože změna délky segmentu by způsobila, že by odchylky dalších segmentů ve zprávě byly nesprávné, a tím by se zabránilo tomu, že by správce front rozpoznal, že segmenty tvoří úplnou logickou zprávu.

MQ_CHANNEL_AUTO_DEF_EXIT-Uživatelská procedura automatické definice kanálu

Volání MQ_CHANNEL_AUTO_DEF_EXIT popisuje parametry, které jsou předávány do uživatelské procedury automatické definice kanálu volané agentem MCA (Message Channel Agent).

Žádný vstupní bod s názvem MQ_CHANNEL_AUTO_DEF_EXIT je poskytován správcem front; název MQ_CHANNEL_AUTO_DEFEXIT nemá žádný speciální význam, protože názvy uživatelských procedur automatické definice jsou ve správci front zadány.

Syntaxe

MQ_CHANNEL_AUTO_DEF_EXIT *(ChannelExitParms, ChannelDefinition)*

Parametry

Volání MQ_CHANNEL_AUTO_DEF_EXIT má následující parametry.

Parametry ChannelExitParms (MQCXP)-vstupní/výstupní

Blok výstupních parametrů kanálu.

Tato struktura obsahuje další informace vztahující se k vyvolání uživatelské procedury.Uživatelská procedura nastaví informace v této struktuře tak, aby ukazovalo, jak bude agent MCA pokračovat.

ChannelDefinition (MQCD)-input/output

Definice kanálu.

Tato struktura obsahuje parametry nastavené administrátorem k řízení chování kanálů, které jsou vytvářeny automaticky.Uživatelská procedura nastaví informace v této struktuře, aby bylo možné upravit výchozí chování nastavené administrátorem.

Pole MQCD uvedená v seznamu nesmí být změněna uživatelskou procedurou:

- *ChannelName*
- *ChannelType*
- *StrucLength*
- *Version*

Změní-li se jiná pole, hodnota nastavená uživatelskou procedurou musí být platná. Pokud hodnota není platná, chybová zpráva se zapíše do souboru protokolu chyb nebo se zobrazí na konzole (jak je to vhodné pro prostředí).

Upozornění: Automaticky definované kanály vytvořené uživatelskou procedurou automatické definice kanálu (CHAD) nemohou nastavit jmenovku certifikátu, protože k navázání komunikace TLS došlo v době vytvoření kanálu. Nastavení štítku certifikátu v uživatelské proceduře CHAD pro příchozí kanály nemá žádný účinek.

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
```

Vyvolání asembleru System/390

```
         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, která je prováděna uživatelskou procedurou kanálu, je definována poskytovatelem uživatelské procedury. Ukončení se však musí řídit pravidly definovanými zde a v přidruženém řídicím bloku, MQCXP.
2. Parametr **ChannelExitParms** předaný do uživatelské procedury automatické definice kanálu je struktura MQCXP. Předaná verze MQCXP závisí na prostředí, ve kterém je spuštěna uživatelská procedura; podrobnosti naleznete v popisu pole *Version* v příručce “MQCXP-Výstupní parametr kanálu” na stránce 1504 .
3. Parametr **ChannelDefinition** předaný do uživatelské procedury automatické definice kanálu je struktura MQCD. Předaná verze produktu MQCD závisí na prostředí, v němž je spuštěna uživatelská procedura; podrobnosti naleznete v popisu pole *Version* v příručce “MQCD-Definice kanálu” na stránce 1464 .

MQXWAIT-Čekání na ukončení

Volání MQXWAIT čeká na výskyt události. Lze ji použít pouze z uživatelské procedury kanálu v produktu z/OS.

Použití funkce MQXWAIT pomáhá vyhnout se problémům s výkonem, které by jinak mohly nastat, pokud procedura ukončení kanálu způsobí čekání. Událost MQXWAIT čeká na signál z objektu MVS MVS (řídicí blok událostí). ECB je popsána v popisu řídicího bloku MQXWD.

z/OS Další informace o použití programu MQXWAIT a zápisu ukončovacích programů kanálů naleznete v tématu Psaní výstupních programů kanálů 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-Ukončení deskriptoru čekání” na stránce 1518 .

CompCode (MQLONG)-výstup

Kód dokončení.

Je to jeden z následujících kódů:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup

Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

MQRC_ADAPTER_NOT_AVAILABLE
(2204, X'89C') Adaptér není k dispozici.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'7FE') Volby nejsou platné nebo nejsou konzistentní.

ZRUŠENÉ MQRC_XWAIT_CANCELED
(2107, X'83B') Volání MQXWAIT bylo zrušeno.

CHYBA MQRC_XWAIT_ERROR
(2108, X'83C') Vyvolání volání MQXWAIT není platné.

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 */
```

Vyvolání asembleru System/390

```
         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 provedení kanálu. Předá se každému ukončovacímu programu kanálu, který je volán z agenta MCA (Message Channel Agent).

Další informace o uživatelských procedurách kanálů naleznete v tématu “MQ_CHANNEL_EXIT-Ukončení kanálu” na stránce 1457. Popis v tomto tématu se týká jak kanálů zpráv, tak kanálů MQI.

Pole jména ukončení

Když je zavolána uživatelská procedura, obsahuje příslušné pole z polí *SecurityExit*, *MsgExit*, *SendExit*, *ReceiveExit*a *MsgRetryExit* název uživatelské procedury, která se právě volá. Význam názvu v těchto polích závisí na prostředí, ve kterém je agent MCA spuštěn. Není-li uvedeno jinak, jméno je zarovnáno vlevo uvnitř pole bez vložených mezer; jméno je doplněno mezerami do délky pole. V popisech, které následují, hranaté závorky ([]) označují nepovinné informace:

AIX and Linux
Název uživatelské procedury je název dynamicky zaváděného modulu nebo knihovny s příponou s příponou s názvem funkce umístěné v dané knihovně. Název funkce musí být uzavřen do závorek. Název knihovny může být volitelně s předponou cesty k adresáři:

```
[ path ] library ( function )
```

Název je omezen na maximálně 128 znaků.

z/OS
Název uživatelské procedury je název načítaného modulu, který je platný pro specifikaci v parametru EP makra LINK nebo LOAD. Název je omezen na maximálně osm znaků.

Windows
Název uživatelské procedury je název knihovny s dynamicky propojovacím odkazem s příponou s názvem funkce umístěné v dané knihovně. Název funkce musí být uzavřen do závorek. Název knihovny může být volitelně uvozeno cestou k adresáři a jednotkou:

```
[d:][ path ] library ( function )
```

Název je omezen na maximálně 128 znaků.

IBM i
Název uživatelské procedury je desetibajtový název programu následovaný desetibajtovým názvem knihovny. Jsou-li názvy kratší než 10 bajtů, každý název je doplněn mezerami, aby se zarovnali 10 bajtů. Název knihovny může být `*LIBL` , ale při volání uživatelské procedury automatické definice kanálu. V takovém případě je třeba zadat úplný název.

Změna polí MQCD v uživatelské proceduře kanálu

Uživatelská procedura kanálu může měnit pole na disku MQCD. Změněná hodnota zůstane na aplikaci MQCD a bude předána zbývajícím uživatelským procedurám v řetězu ukončení a v libovolné konverzaci sdílející instanci kanálu. Změněný objekt MQCD je také používán agentem MCA pro normální zpracování během pokračující životnosti kanálu.

Uživatelská procedura nesmí být změněna následujícími poli MQCD:

- ChannelName
- ChannelType
- StrucLength
- Verze

Související odkazy

"Pole" na stránce 1465
Toto téma uvádí všechna pole ve struktuře MQCD a popisuje každé pole.

"Deklarace C" na stránce 1491
Toto deklarace je deklarací C pro strukturu MQCD.

"Deklarace COBOL" na stránce 1493
Toto deklarace je deklarací COBOL pro strukturu MQCD.

"Deklarace RPG (ILE)" na stránce 1495
Toto prohlášení je deklarací RPG pro strukturu MQCD.

"Deklarace asembleru System/390" na stránce 1498
Toto prohlášení je deklarací assembleru System/390 pro strukturu MQCD.

"Deklarace jazyka Visual Basic" na stránce 1499
Toto prohlášení je prohlášení o Visual Basicu struktury MQCD.

"Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1501
Uživatelská procedura kanálu může měnit pole na disku MQCD. Tyto změny se však obvykle nepodniká, s výjimkou uvedených okolností.

Pole

Toto téma uvádí všechna pole ve struktuře MQCD a popisuje každé pole.

Limit BatchDataLimit (MQLONG)
Toto pole určuje omezení množství dat, které lze prostřednictvím kanálu odeslat před přijetím bodu synchronizace, v kilobajtech.

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.
- **`BatchDataLimit`** bajtů bylo odesláno.
- Přenosová fronta je prázdná a **`BatchInterval`** je překročena.

Hodnota musí být v rozsahu 0 až 999999. Výchozí hodnota je 5000.

Hodnota nula v tomto atributu znamená, že pro dávky přes tento kanál se nepoužije žádné omezení dat.

Tento parametr se vztahuje pouze na kanály s parametrem *`ChannelType`* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSRCVR nebo MQCHT_CLUSSDR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *`Version`* je menší než MQCD_VERSION_11.

BatchHeartbeat (MQLONG
Toto pole uvádí časový interval, který se používá ke spuštění prezenčního signálu dávky pro kanál.

Dávkové prezenční signál umožňuje odesílacím kanálům určit, zda je vzdálená instance kanálu stále aktivní, než bude nejistá. Prezenční signál dávky se vyskytne, pokud odesílací kanál nekomunikoval s instancí vzdáleného kanálu v uvedeném časovém intervalu.

Hodnota je v rozsahu od 0 do 999 999; jednotky jsou milisekundy. Hodnota nula označuje, že dávkové pulzování dávky není povoleno.

Toto pole je relevantní pouze pro kanály, které mají *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_7.

BatchInterval (MQLONG)
Toto pole uvádí přibližný čas v milisekundách, po který kanál udržuje dávku otevřenou, je-li v aktuální dávce méně než *BatchSize* zpráv.

Je-li *BatchInterval* větší než nula, dávka se ukončí podle toho, která z následujících událostí se vyskytne jako první:

- *BatchSize* zprávy byly odeslány, nebo
- *BatchInterval* milisekund uplynulo od začátku dávky.

Je-li *BatchInterval* nula, dávka se ukončí podle toho, která z následujících událostí se vyskytne jako první:

- *BatchSize* zprávy byly odeslány, nebo
- přenosová fronta bude prázdná.

BatchInterval musí být v rozsahu nula až 999 999 999.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, když *Version* je 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 *ChannelType* MQCHT_SVRCONN nebo MQCHT_CLNTCONN.

CertificateLabel (MQCHAR64)
Toto pole uvádí podrobnosti o použité návěští certifikátu.

IBM MQ inicializuje výchozí hodnotu pro pole *CertificateLabel* jako prázdné místo.

To je interpretováno za běhu jako výchozí hodnota a je zpětně kompatibilní.

Například určení verze produktu MQCD nižší než 11 nebo použití výchozí hodnoty mezer pro pole *CertificateLabel* znamená, že toto pole bude ignorováno.

Délka tohoto pole je dána hodnotou MQ_CERT_LABEL_LENGTH.

ChannelMonitoring (MQLONG)
Toto pole uvádí aktuální úroveň shromažďování monitorovacích dat pro kanál.

Toto pole není relevantní pro kanály s typem ChannelType MQCHT_CLNTCONN.

Je to jedna z následujících hodnot:

- MQMON_OFF
- MQMON_LOW
- MQMON_MEDIUM
- MQMON_HIGH

Toto je vstupní pole pro ukončení. Není přítomna, pokud *Version* je menší než MQCD_VERSION_8.

ChannelName (MQCHAR20)
Toto pole uvádí název definice kanálu.

Ve vzdáleném počítači musí existovat definice kanálu se stejným názvem, aby bylo možné komunikovat.

Název musí používat pouze znaky:

- Velká písmena A-Z
- Malá písmena a-z
- Číslice 0-9
- Tečka (.)
- Lomítko (/)
- Podtržítko (_)
- Procento (%)

a být polstrované vpravo s mezerami. Vložené mezery ani mezery na začátku nejsou povoleny.

Délka tohoto pole je dána hodnotou MQ_CHANNEL_NAME_LENGTH.

ChannelStatistics (MQLONG)
Toto pole uvádí aktuální úroveň shromažďování statistických dat pro kanál.

Toto pole není relevantní pro kanály s typem ChannelType MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Je to jedna z následujících hodnot:

- MQMON_OFF
- MQMON_LOW
- MQMON_MEDIUM
- MQMON_HIGH

Toto je vstupní pole pro ukončení. Není přítomna, pokud *Version* je menší než MQCD_VERSION_8.

ChannelType (MQLONG)
Toto pole určuje typ kanálu.

Je to jedna z následujících hodnot:

MQCHT_SENDER
Odesílatel.

SERVER MQCHT_SERVER
.

PŘÍJEMCE MQCHT_RECEIVER
Přijímač.

MQCHT_REQUESTER
Žadatel.

MQCHT_CLNTCONN
Připojení klienta.

FUNKCE MQCHT_SVRCONN
Server-připojení (pro použití klienty).

MQCHT_CLUSDR
Odesílatel klastru.

SOUBOR MQCHT_CLURCVR
Příjemce klastru.

Váha ClientChannel(MQLONG)
Toto pole určuje váhu ovlivňující definici kanálu připojení klienta, která má být použita.

Používá se atribut váhy klienta ClientChannel, aby bylo možné náhodně vybrat definice kanálů klienta na základě jejich váhy, je-li k dispozici více než jedna vhodná definice. Když klient vydá požadavek MQCONN, který požaduje připojení ke skupině správců front, zadáním názvu správce front začínajícího hvězdičkou a více než jedné vhodné definice kanálu je k dispozici v tabulce CCDT (Client Channel Definition CCDT), bude definice použití náhodně vybrána na základě váhy s libovolnějšími definicemi ClientChannels váhou (0), které byly vybrány 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. Distribuce zpráv mezi dvěma nebo více kanály s nenulovým váhami je úměrná poměru těchto vah. Například, tři kanály s hodnotami 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 *Verze* je menší než MQCD_VERSION_9.

ClusterPtr (MQPTR)
Toto pole uvádí adresu seznamu názvů klastru.

Je-li *ClustersDefined* větší než nula, je tato adresa adresou seznamu názvů klastru. Kanál patří ke každému uvedenému klastru.

Toto pole je relevantní pouze pro kanály s rozhraním *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_5.

ClustersDefined (MQLONG)
Toto pole určuje počet klastrů, ke kterým kanál patří.

Toto pole je počet názvů klastrů, na které ukazuje *ClusterPtr*. Je nula nebo větší.

Toto pole je relevantní pouze pro kanály s rozhraním *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_5.

CLWLChannelPriority (MQLONG)
Toto pole určuje prioritu kanálu pracovní zátěže klastru.

Správce pracovní zátěže zvolí místo určení s nejvyšší prioritou ze sady cílů vybraných na základě ohodnocení důležitosti. Pokud existují dva možné správce cílových front, lze tento atribut použít k převedení jednoho správce front na druhého správce front. Všechny zprávy jdou do správce front s nejvyšší prioritou do té doby, než budou ukončeny všechny zprávy, které jsou odesílány do správce front s nejvyšší prioritou.

Hodnota je v rozsahu od 0 do 9. Výchozí hodnota je 0.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_8.

Další informace naleznete v tématu Konfigurace klastru správců front.

CLWLChannelRank (MQLONG)
Toto pole uvádí ohodnocení důležitosti kanálu pracovní zátěže klastru.

Zvolený algoritmus správce pracovní zátěže vybere místo určení s nejvyšší úrovní hodnocení. Je-li konečným cílem správce front v jiném klastru, můžete nastavit pořadí středních správců front brány (v průniku sousedních klastrů), aby si příslušný algoritmus vybral správně cílového správce front s blížícím se cílovým místem určení.

Hodnota je v rozsahu od 0 do 9. Výchozí hodnota je 0.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než 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.

Správce pracovní zátěže pro výběr algoritmu používá atribut "weight" kanálu k posunu cíle tak, aby bylo možné odeslat více zpráv určitému počítači. Například můžete dát kanál na velkém serveru UNIX větší "váhu" než jiný kanál na malém PC počítače, a zvolit algoritmus zvolí UNIX server častěji než PC.

Hodnota je v rozsahu od 1 do 99. Výchozí hodnota je 50.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_8.

Další informace naleznete v tématu Konfigurace klastru správců front.

ConnectionAffinity (MQLONG)
Toto pole určuje, zda klientské aplikace, které se připojují vícekrát s použitím 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 možností:

PREFEROVANÉ MQCAFTY_
První připojení v procesu čtení tabulky CCDT (Client Channel Definition table) vytváří seznam použitelných definic založených na vážení s příslušnými definicemi CLNTWGHT (0) jako první a v abecedním pořadí. Každé připojení v procesu se pokusí připojit pomocí první definice v seznamu. Pokud se navázání připojení nezdaří, je použita další definice. Neúspěšná definice s hodnotami CLNTWGHT jiných 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.

U klientských aplikací napsaných v jazycích C, C++ nebo v programovacím rámci .NET (včetně plně spravovaných .NET) se seznam aktualizuje, pokud byla tabulka CCDT upravena od vytvoření seznamu.

Tato hodnota je výchozí hodnotou.

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.

U klientských aplikací napsaných v jazycích C, C++ nebo v programovacím rámci .NET (včetně plně spravovaných .NET) se seznam aktualizuje, pokud byla tabulka CCDT upravena od vytvoření seznamu.

Tento atribut je platný pouze pro typ kanálu připojení klienta.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Verze* je menší než MQCD_VERSION_9.

ConnectionName (MQCHAR264)
Toto pole uvádí název připojení pro kanál.

Pro kanály příjemce klastru (je-li zadán) CONNAME se vztahuje k lokálnímu správci front, a pro další kanály, které souvisí s cílovým správcem front. Hodnota, kterou zadáte, závisí na přenosovém protokolu (*TransportType*), který má být použit:

- Pro MQXPT_LU62je to plně kvalifikovaný název partnerské logické jednotky.
- Pro MQXPT_NETBIOS se jedná o název NetBIOS definovaný na vzdáleném počítači.
- Pro MQXPT_TCP je to buď název hostitele, síťová adresa vzdáleného počítače uvedená ve tečkovém desítkovém zápisu IPv4 nebo hexadecimální formát IPv6 , nebo lokální počítač pro kanály příjemce klastru.
- Pro MQXPT_SPX se jedná o adresu ve stylu SPX obsahující 4bajtovou síťovou adresu, 6bajtovou adresu uzlu a 2bajtovou adresu soketu.

Při definování kanálu není toto pole relevantní pro kanály s *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 přítomno, pokud *Version* je menší než MQCD_VERSION_2.

DataConversion (MQLONG)
Toto pole uvádí, zda se odesílající agent kanálu zpráv pokusí o konverzi dat zprávy aplikace, pokud přijímající agent kanálu zpráv nemůže provést tento převod.

Toto pole se vztahuje pouze na zprávy, které nejsou segmenty logických zpráv; agent MCA se nikdy nepokusí o převod zpráv, které jsou segmenty.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR. Jedná se o jednu z následujících položek:

KONVERZE MQCDC_SENDER_CONVERSION
Převod odesílatelem.

KONVERZE MQCDC_NO_SENDER_CONVERSION
Odesilatel nekonvertují.

DefReconnect (MQLONG)
Atribut kanálu produktu `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.

Reconnction je volba `MQCONNX` . Pomocí atributu kanálu produktu `DefReconnect` můžete přidat chování opětovného připojení k existujícím aplikacím, které používají produkt `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`
Připojení je zakázáno, a to i v případě, že o to klientský program požádá prostřednictvím volání **MQCONNX** MQI.

Automatické opětovné připojení klienta není podporováno produktem IBM MQ classes for Java.

Tabulka 823. 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

Připojení kanálu a klienta znovu

Související odkazy

stanza CHANNELS konfiguračního souboru klienta

Volby připojení

Volby, které řídí akci MQCONNX.

Popis (MQCHAR64)
Toto pole lze použít pro popisný komentář.

Obsah pole nemá význam pro agenty kanálu zpráv. Musí však obsahovat pouze znaky, které lze zobrazit. Nesmí obsahovat žádné prázdné znaky; je-li to nutné, je zprava vyplněno mezerami. V případě instalace 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, je-li toto pole odesláno jinému správci front.

Délka tohoto pole je dána hodnotou MQ_CHANNEL_DESC_LENGTH.

DiscInterval (MQLONG)
Toto pole uvádí maximální dobu (v sekundách), po kterou kanál čeká na příchod zprávy do přenosové fronty, před ukončením kanálu.

Jinými slovy, určuje interval odpojení.

Nulová hodnota způsobí, že agent MCA bude čekat po neomezenou dobu.

Pro kanály připojení serveru používající protokol TCP interval představuje hodnotu odpojení neaktivního klienta, která je uvedena v sekundách. Pokud připojení k serveru neobdrželo od svého partnerského klienta po tuto dobu žádnou komunikaci, ukončí spojení. Interval nečinnosti připojení serveru se používá pouze mezi voláními rozhraní API produktu IBM MQ od klienta, takže během dlouhodobé operace MQGET bez čekání na připojení není odpojen žádný klient.

Tento atribut nelze použít pro kanály připojení serveru pomocí protokolů jiných než TCP.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR, MQCHT_CLUSRCVR nebo MQCHT_SVRCONN.

Délka ExitData(MQLONG)
Toto pole uvádí délku každého z datových položek uživatele v seznamech uživatelských datových položek, které jsou adresovány 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 *Version* je menší než MQCD_VERSION_4.

ExitNameDélka (MQLONG)
Toto pole určuje délku každého z názvů v bajtech, která jsou uvedena v seznamech výstupních názvů, 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 *Version* je menší než MQCD_VERSION_4.

Seznam HdrComp[2] (MQLONG)
Toto pole uvádí seznam technik komprese dat záhlaví, které jsou podporovány kanálem.

Seznam obsahuje jednu nebo více z následujících hodnot:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat hlavičky.

SYSTÉM MQCOMPRESS_SYSTEM
Provádí se komprese dat hlavičky.

Nepoužívané hodnoty v poli jsou nastaveny na hodnotu MQCOMPRESS_NOT_AVAILABLE.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_8.

HeartbeatInterval (MQLONG)
Toto pole uvádí dobu (v sekundách) mezi toky synchronizačních signálů.

Interpretace tohoto pole závisí na typu kanálu následujícím způsobem:

- Pro typ kanálu MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR je toto pole čas v sekundách mezi toky synchronizačních signálů předávanými z odesílající sběrnice MCA, když nejsou v přenosové frontě žádné zprávy. To dává přijímajícímu agentovi MCA možnost uvést kanál do klidového stavu. Chcete-li být užitečný, *HeartbeatInterval* musí být menší než *DiscInterval*.
- Pro typ kanálu MQCHT_CLNTCONN nebo MQCHT_SVRCONN s polem Konverzace sdílení MQCD nastaveným na hodnotu 0 je toto pole čas v sekundách mezi toky synchronizačních signálů předávanými z agenta MCA serveru, když tato MCA vydala volání MQGET s volbou MQGMO_WAIT v zastoupení aplikace klienta. To umožňuje serveru MCA obsluhovat situace, kdy se připojení klienta nezdaří během MQGET s MQGMO_WAIT.
- Pro typ kanálu MQCHT_CLNTCONN nebo MQCHT_SVRCONN s polem Konverzace sdílení MQCD nastaveným na neprázdnou hodnotu toto pole odpovídá času v sekundách mezi tokem prezenčního signálu, když nejsou odeslány nebo přijaty žádné toky dat. To umožňuje efektivní uvedení kanálu do klidového stavu.

Hodnota je v rozsahu od 0 do 999 999. Hodnota, která se používá, je větší z hodnot zadaných na odesílající straně a přijímající straně, pokud není hodnota 0 uvedena na obou stranách, v takovém případě nedochází k žádné výměně synchronizačních signálů.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Interval KeepAliveinterval (MQLONG)
Toto pole uvádí hodnotu předanou do komunikačního zásobníku pro časování uchování 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. Nulová hodnota určuje, že udržování aktivity kanálu není povoleno, ačkoli je možné zachovat udržení aktivity protokolu TCP/IP, pokud je povolena funkce udržení aktivity TCP/IP (místo 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 následujícím způsobem:

  - Pokud je vyjednaný interval prezenčního signálu větší než nula, interval udržení aktivity, který se používá, je interval prezenčního signálu plus 60 sekund.
  - Je-li vyjednaný interval prezenčního signálu nula, je použitý interval udržení aktivity nastaven na nulu.
- V systému z/OSdochází k udržení aktivity TCP/IP, je-li v objektu správce front zadán parametr TCPKEEP (YES).
- V jiných prostředích probíhá udržení aktivity TCP/IP, pokud je parametr **KEEPALIVE=YES** zadán ve stanze TCP v konfiguračním souboru s distribuovanými frontami.

Toto pole je relevantní pouze pro kanály, které mají *TransportType* MQXPT_TCP nebo MQXPT_SPX.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_7.

LocalAddress (MQCHAR48)
Toto pole uvádí lokální adresu TCP/IP definovanou pro kanál pro odchozí komunikaci.

Toto pole je prázdné, pokud není definována žádná specifická adresa pro odchozí komunikaci. Adresa může volitelně obsahovat číslo portu nebo rozsah čísel portů. Formát této adresy je:

```
[ip-addr][(low-port[,high-port])]
```

kde hranaté závorky ([]) označují nepovinné informace, `ip-addr` je zadán v desítkové tečkové notaci IPv4 , IPv6 hexadecimální nebo alfanumerický formát a `low-port` a `high-port` jsou čísla portů uzavřená v závorkách. Vše je nepovinné.

Specifická adresa IP, port nebo rozsah portů pro odchozí komunikaci jsou užitečné ve scénářích zotavení, kde je kanál restartován na jiném zásobníku TCP/IP.

LocalAddress je podobný ve tvaru *ConnectionName*, ale nesmí být zaměňován s ním. Parametr *LocalAddress* určuje charakteristiky lokální komunikace, zatímco *ConnectionName* určuje, jak se má dostat ke vzdálenému správci front.

V 9.2.0.2 V 9.2.2 Od IBM MQ 9.2.0 Fix Pack 2 for Long Term Support and IBM MQ 9.2.2 for Continuous Deliverybylo aktualizováno Java Message Queueing Interface (JMQI), aby se zajistilo, že pole lokální adresy bude nastaveno na objekt MQCD poté, co byla vytvořena instance kanálu a je připojena ke správci front. To znamená, že při ukončení kanálu napsaného v Java volá metoda `MQCD.getLocalAddress()` metoda, která vrací lokální adresu, kterou instance kanálu používá. Před IBM MQ 9.2.0 Fix Pack 2 a IBM MQ 9.2.2nemohla uživatelská procedura zabezpečení kanálu přistupovat k lokální adrese používané instancí kanálu a metoda `MQCD.getLocalAddress()` vrátila hodnotu null.

Toto pole je relevantní pouze pro kanály s *TransportType* MQXPT_TCP a *ChannelType* z 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 *Version* je menší než MQCD_VERSION_7.

LongMCAUserIdLength (MQLONG)
Toto pole uvádí délku úplného identifikátoru uživatele MCA, na který ukazuje *LongMCAUserIdPtr*, v bajtech.

Toto pole není relevantní pro kanály s *ChannelType* MQCHT_CLNTCONN.

Jedná se o vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

LongMCAUserIdPtr (MQPTR)
Toto pole uvádí adresu dlouhého identifikátoru uživatele MCA.

Je-li *LongMCAUserIdLength* větší než nula, je toto pole adresou celého jména uživatele MCA. Délka celého identifikátoru je dána *LongMCAUserIdLength*. Prvních 12 bajtů identifikátoru uživatele MCA se také nachází v poli *MCAUserIdentifier*.

Podrobnosti o identifikátoru uživatele MCA najdete v popisu pole *MCAUserIdentifier* .

Toto pole není relevantní pro kanály s *ChannelType* MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN nebo MQCHT_CLUSSDR.

Jedná se o vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

LongRemoteUserIdDélka (MQLONG)
Toto pole uvádí délku úplného vzdáleného identifikátoru uživatele, na který ukazuje *LongRemoteUserIdPtr*, v bajtech.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

LongRemoteUserIdPtr (MQPTR)
Toto pole uvádí adresu dlouhého vzdáleného identifikátoru uživatele.

Je-li *LongRemoteUserIdLength* větší než nula, je tento parametr adresa úplného identifikátoru vzdáleného uživatele. Délka celého identifikátoru je dána *LongRemoteUserIdLength*. Prvních 12 bajtů identifikátoru vzdáleného uživatele je také obsaženo v poli *RemoteUserIdentifier*.

Podrobnosti o identifikátoru vzdáleného uživatele najdete v popisu pole *RemoteUserIdentifier* .

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

Počet LongRetryCount (MQLONG)
Toto pole uvádí počet použitý po vyčerpání počtu, který byl zadán *ShortRetryCount* .

Určuje maximální počet dalších pokusů o připojení ke vzdálenému počítači, v intervalech určených parametrem *LongRetryInterval*, před tím, než se do operátoru protokolování chyb přihlásí.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

LongRetryInterval (MQLONG)
Toto pole uvádí maximální počet sekund, po které se má čekat, než se znovu pokusí o připojení ke vzdálenému počítači.

Interval mezi novými pokusy lze rozšířit, pokud má kanál čekat, než se stane aktivním.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

MaxInstances (MQLONG)
Toto pole určuje maximální počet současně existujících instancí individuálního kanálu připojení k serveru, které lze spustit.

Toto pole je použito pouze v kanálech připojení serveru.

Pole může mít hodnotu v rozsahu 0-999 999 999. Hodnota nula předchází všechny přístupy klienta.

Výchozí hodnota tohoto pole je 999 999 999.

Je-li hodnota tohoto pole zmenšena na číslo, které je nižší než počet instancí kanálu připojení serveru, které jsou aktuálně spuštěny, pak tyto spuštěné instance nebudou ovlivněny. Nové instance se však nemohou spustit, dokud nebudou spuštěny dostatečné existující instance, takže počet momentálně spuštěných instancí je menší než hodnota pole.

MaxInstancesPerClient (MQLONG)
Toto pole určuje maximální počet současně existujících instancí jednotlivých kanálů připojení serveru, které lze spustit z jednoho klienta.

V tomto kontextu jsou připojení, která pocházejí ze stejné vzdálené síťové adresy, považována za přicházející od stejného klienta.

Toto pole je použito pouze v kanálech připojení serveru.

Pole může mít hodnotu v rozsahu 0-999 999 999. Hodnota nula předchází všechny přístupy klienta.

Výchozí hodnota tohoto pole je 999 999 999.

Je-li hodnota tohoto pole zmenšena na číslo, které je nižší než počet instancí kanálu připojení serveru, které jsou aktuálně spuštěny z jednotlivých klientů, nebudou tyto spuštěné instance ovlivněny. Nové instance z některého z těchto klientů však nemohou začít, dokud nebudou spuštěny dostatečné existující instance, takže počet aktuálně spuštěných instancí, pocházejících z klienta, který se pokouší o spuštění nové instance, je menší než hodnota pole.

MaxMsgDélka (MQLONG)
Toto pole uvádí maximální délku zprávy, kterou lze přenést na kanál.

Ta je porovnána s hodnotou pro vzdálený kanál a skutečné maximum je nižší z těchto dvou hodnot.

MCanime (MQCHAR20)
Toto pole je rezervované pole.

Hodnota tohoto pole je prázdná.

Délka tohoto pole je dána hodnotou MQ_MCA_NAME_LENGTH.

MCASecurityId (MQBYTE40)
Toto pole uvádí identifikátor zabezpečení pro agenta MCA.

Toto pole není relevantní pro kanály s *ChannelType* MQCHT_CLNTCONN.

Následující speciální hodnota označuje, že neexistuje žádný identifikátor zabezpečení:

MQSID_NONE
Není uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQSID_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQSID_NONE, ale je to pole znaků místo řetězce.

Jedná se o vstupní/výstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_SECURITY_ID_LENGTH. Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

MCATType (MQLONG)
Toto pole uvádí typ programu agenta kanálu zpráv.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Hodnota je jedna z následujících možností:

PROCES 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 k dispozici, je-li hodnota *Verze* menší než hodnota MQCD_VERSION_2.

MCAUserIdentifier (MQCHAR12)
Toto pole uvádí identifikátor uživatele pro agenta kanálu zpráv (MCA).

Toto pole používá prvních 12 bajtů identifikátoru uživatele MCA a lze jej nastavit pomocí agenta zabezpečení.

Jsou dvě pole, která obsahují identifikátor uživatele MCA:

- *MCAUserIdentifier* obsahuje prvních 12 bajtů identifikátoru uživatele MCA a je doplněn mezerami, je-li 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 ukončený znakem null. Je-li identifikátor prázdný, *LongMCAUserIdLength* je nula a hodnota *LongMCAUserIdPtr* není definovaná.

  Poznámka: *LongMCAUserIdPtr* není přítomen, pokud *Version* je menší než MQCD_VERSION_6.

Je-li identifikátor uživatele MCA neprázdný, uvádí identifikátor uživatele, který má být použit agentem kanálu zpráv pro autorizaci pro přístup k prostředkům produktu IBM MQ . Pro typy kanálů MQCHT_REQUESTER, MQCHT_RECEIVER a MQCHT_CLUSRCVR, je-li hodnota PutAuthority MQPA_DEFAULT, je tento identifikátor uživatele použit pro kontrolu autorizace pro operaci vložení do cílových front.

Je-li identifikátor uživatele MCA prázdný, použije agent kanálu zpráv výchozí identifikátor uživatele.

Identifikátor uživatele MCA může být nastaven pomocí uživatelské procedury zabezpečení, který označuje identifikátor uživatele, který musí agent kanálu zpráv použít. Uživatelská procedura může změnit buď *MCAUserIdentifier*, nebo řetězec, na který ukazuje *LongMCAUserIdPtr*. Pokud se obě hodnoty změnily, ale liší se od sebe navzájem, program MCA používá *LongMCAUserIdPtr* jako předvolbu pro *MCAUserIdentifier*. Pokud uživatelská procedura změní délku řetězce adresovaného *LongMCAUserIdPtr*, *LongMCAUserIdLength* musí být nastaven odpovídajícím způsobem. Pokud uživatelská procedura zvýší délku identifikátoru, musí uživatelská procedura alokovat paměť požadované délky, nastavit toto úložiště na požadovaný identifikátor a umístit adresu tohoto úložiště do *LongMCAUserIdPtr*. Uživatelská procedura je odpovědná za uvolnění úložného prostoru, je-li uživatelská procedura později vyvolána s příčinou MQXR_TERM.

Pro kanály s *ChannelType* MQCHT_SVRCONN, je-li *MCAUserIdentifier* v definici kanálu prázdné, bude do něj zkopírován jakýkoli identifikátor uživatele přenesený z klienta. Tento identifikátor uživatele (po jakékoli modifikaci bezpečnostní procedurou na serveru) je ten, pod kterým se předpokládá, že klientská aplikace běží.

Identifikátor uživatele MCA není relevantní pro kanály s *ChannelType* MQCHT_SDR, MQCHT_SVR, MQCHT_CLNTCONN, MQCHT_CLUSSDR.

Jedná se o vstupní/výstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_2.

ModeName (MQCHAR8)
Toto pole uvádí název režimu LU 6.2 .

Toto pole je relevantní pouze v případě, že přenosový protokol (*TransportType*) je MQXPT_LU62a produkt *ChannelType* není MQCHT_SVRCONN nebo MQCHT_RECEIVER.

Toto pole je vždy prázdné. Informace jsou místo toho obsaženy v objektu na straně komunikace.

Délka tohoto pole je dána proměnnou MQ_MODE_NAME_LENGTH.

Seznam MsgComp[16] (MQLONG)
Toto pole uvádí seznam technik komprese dat zpráv, které jsou podporovány kanálem.

Seznam obsahuje jednu nebo více z 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.

Nepoužívané hodnoty v poli jsou nastaveny na hodnotu MQCOMPRESS_NOT_AVAILABLE.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_8.

MsgExit (MQCHARn)
Toto pole určuje název uživatelské procedury pro zprávy kanálu.

Je-li tento název neprázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě po načtení zprávy z přenosové fronty (odesilatel nebo server) nebo bezprostředně před tím, než je zpráva vložena do cílové fronty (příjemce nebo žadatele).

  Výstupem je dána celá hlavička aplikace a záhlaví přenosové fronty pro úpravu.
- Při inicializaci a ukončení kanálu.

Toto pole není relevantní pro kanály s *ChannelType* MQCHT_SVRCONN nebo MQCHT_CLNCONN; a pro takové kanály se uživatelská procedura pro zprávy nikdy nevyvolá.

Viz “MQCD-Definice kanálu” na stránce 1464 , kde najdete popis obsahu tohoto pole v různých prostředích.

Délka tohoto pole je dána proměnnou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro prostředí.

MsgExitPtr (MQPTR)
Toto pole uvádí adresu prvního pole *MsgExit* .

Je-li *MsgExitsDefined* větší než nula, je tato adresa adresou seznamu názvů všech uživatelských procedur kanálu zpráv v řetězci.

Každý název je v poli o délce *ExitNameLength*vyplněný zprava mezerami. Existuje *MsgExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou proceduru.

Jakékoli změny provedené v těchto názvech podle ukončení jsou zachovány, přestože uživatelská procedura kanálu zpráv neprovede žádnou explicitní akci-nezmění se, které uživatelské procedury jsou vyvolány.

Je-li *MsgExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

MsgExitsDefinovaný (MQLONG)
Toto pole určuje počet uživatelských procedur pro zprávy kanálu definovaných v řetězci.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Počet MsgRetry(MQLONG)
Toto pole uvádí počet pokusů agenta MCA o vložení zprávy po prvním pokusu, který selhal.

Toto pole udává počet případů, kdy se agent MCA pokusí o operaci otevření nebo vložení, pokud se nezdaří první volání MQOPEN nebo MQPUT s kódem dokončení MQCC_FAILED. Efekt tohoto atributu závisí na tom, zda je *MsgRetryExit* prázdný nebo neprázdný:

- Pokud je parametr *MsgRetryExit* prázdný, určuje atribut **MsgRetryCount** , zda se agent MCA pokusí o opakované pokusy. Je-li hodnota atributu nula, nepokusí se žádný nový pokus. Je-li hodnota atributu větší než nula, pokusí se o opakované pokusy v intervalech zadaných atributem **MsgRetryInterval** .

  Opakované pokusy jsou zkoušeny pouze u následujících kódů příčiny:

  – ÚPLNÁ OPERACE MQRC_PAGESET_FULL

  – MQRC_PUT_BLOKOVÁNO

  – MQRC_Q_FULL

  U jiných kódů příčiny je agent MCA okamžitě pokračovat v normálním zpracování selhání, aniž by došlo k zopakování nezdařené zprávy.
- Pokud je parametr *MsgRetryExit* prázdný, atribut **MsgRetryCount** neovlivňuje agenta MCA; místo toho se jedná o ukončení opakování zprávy, které určuje, kolikrát je pokus o zopakování proveden, a v jakých intervalech; procedura je vyvolána i v případě, že atribut **MsgRetryCount** je nulový.

  The **MsgRetryCount** attribute is made available to the exit in the MQCD structure, but the exit it not required to honor it - retries continue indefinitely until the exit returns MQXCC_SUPPRESS_FUNCTION in the *ExitResponse* field of MQCXP.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_3.

MsgRetryUkončení (MQCHARn)
Toto pole uvádí název ukončení opakování zprávy kanálu.

Uživatelská procedura opakování zpráv je ukončení, které je vyvoláno agentem MCA, když agent MCA obdrží kód dokončení MQCC_FAILED z volání MQOPEN nebo MQPUT. Účelem této procedury je určení časového intervalu, po který agent MCA čeká před dalším pokusem o zopakování operace MQOPEN nebo MQPUT. Alternativně lze proceduru nastavit, aby se operace nepokusila provést znovu.

Ukončení je vyvoláno pro všechny kódy příčiny, které mají kód dokončení MQCC_FAILED-nastavení uživatelské procedury určuje, jaké kódy příčiny chce agent MCA zkusit znovu, pro počet pokusů a v jakých časových intervalech.

Pokud se již operace neprovede, program MCA provede normální zpracování selhání; toto zpracování zahrnuje generování zprávy zprávy o výjimce (je-li určena odesilatelem) a buď umístění původní zprávy do fronty nedoručených zpráv, nebo zrušení zprávy (podle toho, zda odesílatel uvedl MQRO_DEAD_LETTER_Q nebo MQRO_DISCARD_MSG). Selhání, která zahrnuje frontu nedoručených zpráv (například plná fronta nedoručených zpráv), nezpůsobila vyvolání uživatelské procedury opakování zprávy.

Je-li název uživatelské procedury prázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě před provedením čekání, než se znovu pokusíte doručit zprávu
- Při inicializaci a ukončení kanálu

Viz “MQCD-Definice kanálu” na stránce 1464 , kde najdete popis obsahu tohoto pole v různých prostředích.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Délka tohoto pole je dána proměnnou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro prostředí.

Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_3.

Interval MsgRetryInterval (MQLONG)
Toto pole určuje minimální interval v milisekundách, po jehož uplynutí je operace otevření nebo vložení zopakována.

Efekt tohoto atributu závisí na tom, zda je *MsgRetryExit* prázdný nebo neprázdný:

- Pokud je parametr *MsgRetryExit* prázdný, určuje atribut **MsgRetryInterval** minimální dobu, po kterou agent MCA čeká před zopakováním zprávy, pokud se nezdaří první volání MQOPEN nebo MQPUT s kódem dokončení MQCC_FAILED. Hodnota nula znamená, že opakovaný pokus bude proveden co nejdříve po předchozím pokusu. Opakované pokusy jsou provedeny pouze tehdy, je-li *MsgRetryCount* větší než nula.

  Tento atribut se také používá jako čekací doba, pokud uživatelská procedura pro opakování zprávy vrátí neplatnou hodnotu v poli *MsgRetryInterval* v MQCXP.
- Není-li parametr *MsgRetryExit* prázdný, neovlivní atribut **MsgRetryInterval** funkci MCA; místo toho se jedná o uživatelskou proceduru opakování zprávy, která určuje, jak dlouho agent MCA čeká. Atribut **MsgRetryInterval** je k dispozici pro uživatelskou proceduru ve struktuře MQCD, ale při ukončení není nutné jej respektovat.

Hodnota je v rozsahu od 0 do 999 999 999.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_3.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCD_VERSION_4.

MsgRetryUserData (MQCHAR32)
Toto pole uvádí uživatelská data ukončení opakování zprávy kanálu.

Tato data jsou předána uživatelské proceduře kanálu pro opakování zpráv v poli *ExitData* v parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na začátku obsahuje data, která byla nastavena v definici kanálu. Avšak během doby životnosti této instance MCA jsou všechny změny provedené v obsahu tohoto pole při ukončení libovolného typu zachovány agentem MCA a jsou viditelné pro následná vyvolání ukončení (bez ohledu na typ) pro tuto instanci MCA. Takové změny nemají vliv na definici kanálu používanou jinými instancemi MCA. Mohou být použity jakékoliv znaky (včetně binárních dat).

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH. Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_3.

Toto pole není relevantní pro IBM MQ for IBM i.

Data MsgUserData (MQCHAR32)
Toto pole uvádí 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 začátku obsahuje data, která byla nastavena v definici kanálu. Avšak během doby životnosti této instance MCA jsou všechny změny provedené v obsahu tohoto pole při ukončení libovolného typu zachovány agentem MCA a jsou viditelné pro následná vyvolání ukončení (bez ohledu na typ) pro tuto instanci MCA. Takové změny nemají vliv na definici kanálu používanou jinými instancemi MCA. Mohou být použity jakékoliv znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není relevantní pro IBM MQ for IBM i.

MsgUserDataPtr (MQPTR)
Toto pole uvádí adresu prvního pole *MsgUserData* .

Je-li *MsgExitsDefined* větší než nula, je tato adresa adresou seznamu uživatelských datových položek pro každou uživatelskou proceduru zprávy kanálu v řetězci.

Každá uživatelská datová položka je v poli o délce *ExitDataLength*, která je směrem doprava vyplněna mezerami. Existuje *MsgExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou

proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů procedur, budou nedefinované datové položky uživatele nastaveny na mezery. Naopak, pokud je počet definovaných položek uživatelských dat větší než počet názvů procedur, přebytečné uživatelské datové položky se budou ignorovat a nebudou představeny k ukončení.

Všechny změny provedené v těchto hodnotách budou zachovány. To umožňuje jedné uživatelské proceduře předat informace dalšímu ukončení. Na žádných změnách se neprovedou žádné ověření, takže binární data lze v případě potřeby zapsat do těchto polí.

Je-li *MsgExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

NetworkPriority (MQLONG)
Toto pole uvádí prioritu připojení k síti pro kanál.

Je-li k dispozici více cest k určitému místu určení, je zvolena cesta s nejvyšší prioritou. Hodnota je v rozsahu 0 až 9; 0 je nejnižší priorita.

Toto pole je relevantní pouze pro kanály s rozhraním *ChannelType* MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_5.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCD_VERSION_6.

NonPersistentMsgSpeed (MQLONG)
Toto pole uvádí rychlost, jakou přechodné zprávy cestují přes kanál.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER, MQCHT_REQUESTER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Hodnota je jedna z následujících možností:

MQNPMS_NORMAL
Normální rychlost.

Je-li kanál definován jako hodnota MQNPMS_NORMAL, budou přechodné zprávy přenášeny prostřednictvím kanálu při normální rychlosti. To má tu výhodu, že tyto zprávy nejsou ztraceny, pokud dojde k selhání kanálu. Také trvalé a přechodné zprávy ve stejné přenosové frontě si udržují pořadí ve vztahu k sobě navzájem.

MQNPMS_FAST
Rychlá rychlost.

Je-li kanál definován jako MQNPMS_FAST, přechodné zprávy procházejí kanálem rychlou rychlostí. To zvyšuje propustnost kanálu, ale znamená, že přechodné zprávy se ztratí, dojde-li k selhání kanálu. Je také možné, že přechodné zprávy přeskakovaly před trvalými zprávami čekajícími na stejnou přenosovou frontu, tj. pořadí přechodných zpráv se neudržuje relativně k trvalým zprávám. Avšak pořadí přechodných zpráv relativně k sobě navzájem je udržováno. Podobně i pořadí trvalých zpráv ve vztahu k sobě navzájem se udržuje.

Heslo (MQCHAR12)
Toto pole uvádí heslo použité agentem oznamovacího kanálu při pokusu o inicializaci zabezpečené relace SNA se vzdáleným agentem kanálu zpráv.

Toto pole může být prázdné pouze v produktu AIX, Linux, and Windowsa je relevantní pouze pro kanály s *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER nebo MQCHT_CLNTCONN. V systému z/OStoto pole není relevantní.

Délka tohoto pole je dána hodnotou MQ_PASSWORD_LENGTH. Použije se však pouze prvních 10 znaků.

Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_2.

PropertyControl (MQLONG)
Toto pole určuje, co se stane s vlastnostmi zpráv, pokud se zpráva chystá odeslat do V6 nebo předchozího správce front (správce front, který nerozumí konceptu deskriptoru vlastností).

Hodnota může být některá z následujících:

KOMPATIBILITA MQPROP_COMPATIBILITY
Pokud zpráva obsahuje vlastnost s předponou **mcd.**, **jms.**, **usr.** nebo **mqext.**, jsou všechny vlastnosti zprávy doručovány do aplikace v záhlaví MQRFH2 . Jinak budou všechny vlastnosti zprávy, kromě vlastností obsažených v deskriptoru (či rozšíření) zprávy, zahozeny a nebudou nadále přístupné aplikaci.

Tato hodnota je výchozí hodnotou; umožňuje aplikacím, které očekávají JMSsouvisející vlastnosti, v záhlaví MQRFH2 v datech zprávy pokračovat v práci beze změn.

MQPROP_NONE
Všechny vlastnosti zprávy, kromě vlastností v deskriptoru (či rozšíření) zprávy, budou odebrány ze zprávy před odesláním zprávy vzdálenému správci front.

MQPROP_ALL
Všechny vlastnosti zprávy jsou zahrnuty ve zprávě, když jsou odeslány 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 je použitelný pro kanály odesílatele, Server, odesilatele klastru a příjemce klastru.

"MQIA_ * (Selektory celočíselných atributů)" na stránce 128
"MQPROP_ * (kontrolní hodnoty vlastností fronty a kanálu a maximální délka vlastností)" na stránce 168

PutAuthority (MQLONG)
Toto pole uvádí, zda se identifikátor uživatele v kontextových informacích přidružených ke zprávě používá k zavedení oprávnění k vložení zprávy do cílové fronty.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_REQUESTER, MQCHT_RECEIVER nebo MQCHT_CLUSRCVR. Jedná se o jednu z následujících položek:

VÝCHOZÍ HODNOTA MQPA_DEFAULT
Je použit výchozí identifikátor uživatele.

KONTEXT MQPA_CONTEXT
Identifikátor uživatele kontextu je použit.

MQPA_ALTERNATE_NEBO_MCA
Je použito ID uživatele z pole UserIdentifier deskriptoru zpráv. Jakékoli ID uživatele přijaté ze sítě se nepoužije. Tato hodnota je podporována pouze v systému z/OS.

POUZE MQPA_ONLY_MCA
Je použito výchozí ID uživatele. Jakékoli ID uživatele přijaté ze sítě se nepoužije. Tato hodnota je podporována pouze v systému z/OS.

QMgrName (MQCHAR48)
Toto pole určuje název správce front, ke kterému se může ukončit připojení.

Pro kanály s produktem *ChannelType* jiným než MQCHT_CLNTCONN je toto pole názvem správce front, ke kterému se může připojit uživatelská procedura, která je v systému AIX, Linux, and Windowsvždy neprázdná.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

ReceiveExit (MQCHARn)
Toto pole uvádí název uživatelské procedury pro přijetí kanálu.

Je-li tento název neprázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě před tím, než se zpracovaná síťová data zpracují.

Výstupem je přidělena úplná vyrovnávací paměť pro přenos jako přijatá. Obsah vyrovnávací paměti lze upravit podle potřeby.

- Při inicializaci a ukončení kanálu.

Viz “MQCD-Definice kanálu” na stránce 1464 , kde najdete popis obsahu tohoto pole v různých prostředích.

Délka tohoto pole je dána proměnnou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro prostředí.

ReceiveExitPtr (MQPTR)
Toto pole uvádí adresu prvního pole *ReceiveExit* .

Je-li *ReceiveExitsDefined* větší než nula, je tato adresa adresou seznamu názvů všech uživatelských procedur pro příjem kanálu v řetězci.

Každý název je v poli o délce *ExitNameLength*vyplněný zprava mezerami. Existuje *ReceiveExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou proceduru.

Jakékoli změny provedené v těchto názvech podle ukončení jsou zachovány, přestože uživatelská procedura kanálu zpráv neprovede žádnou explicitní akci-nezmění se, které uživatelské procedury jsou vyvolány.

Je-li *ReceiveExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

ReceiveExitsDefinované (MQLONG)
Toto pole uvádí počet uživatelských procedur příjmu kanálu definovaných v řetězci.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Data ReceiveUserData (MQCHAR32)
Tento kanál určuje uživatelská data uživatelské procedury příjmu kanálu.

Tato data jsou předána uživatelské proceduře pro přijetí zprávy kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na začátku obsahuje data, která byla nastavena v definici kanálu. Avšak během doby životnosti této instance MCA jsou všechny změny provedené v obsahu tohoto pole při ukončení libovolného typu zachovány agentem MCA a jsou viditelné pro následná vyvolání ukončení (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy z různých konverzací. Takové změny nemají vliv na definici kanálu používanou jinými instancemi MCA. Mohou být použity jakékoliv znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není relevantní pro IBM MQ for IBM i.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCD_VERSION_2.

ReceiveUserDataPtr (MQPTR)
Toto pole uvádí adresu prvního pole *ReceiveUserData* .

Je-li *ReceiveExitsDefined* větší než nula, je tato adresa adresou seznamu uživatelských datových položek pro každou uživatelskou proceduru příjmu kanálu v řetězci.

Každá uživatelská datová položka je v poli o délce *ExitDataLength*, která je směrem doprava vyplněna mezerami. Existuje *ReceiveExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů procedur, budou nedefinované datové položky uživatele nastaveny na mezery. Naopak, pokud je počet

definovaných položek uživatelských dat větší než počet názvů procedur, přebytečné uživatelské datové položky se budou ignorovat a nebudou představeny k ukončení.

Všechny změny provedené v těchto hodnotách budou zachovány. To umožňuje jedné uživatelské proceduře předat informace dalšímu ukončení. Na žádných změnách se neprovedou žádné ověření, takže binární data lze v případě potřeby zapsat do těchto polí.

Je-li *ReceiveExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCD_VERSION_5.

RemotePassword (MQCHAR12)
Toto pole uvádí heslo od partnera.

Toto pole obsahuje platné informace pouze v případě, že *ChannelType* je MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

- V případě uživatelské procedury zabezpečení v kanálu MQCHT_CLNTCONN je toto heslo heslem, které bylo získáno z prostředí. Ukončení se může rozhodnout odeslat ji na konec zabezpečení na serveru.
- Pro uživatelskou proceduru zabezpečení v kanálu MQCHT_SVRCONN může toto pole obsahovat heslo, které bylo získáno z prostředí na klientovi, pokud neexistuje žádná uživatelská procedura zabezpečení klienta. Uživatelská procedura může použít toto heslo k ověření identifikátoru uživatele v produktu *RemoteUserIdentifier*.

  Pokud na straně klienta existuje uživatelská procedura zabezpečení, lze tyto informace získat v rámci toku zabezpečení od klienta.

Délka tohoto pole je dána hodnotou MQ_PASSWORD_LENGTH. Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_2.

ID RemoteSecurity(MQBYTE40)
Toto pole uvádí identifikátor zabezpečení pro vzdáleného uživatele.

Toto pole je relevantní pouze pro kanály s *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

Následující speciální hodnota označuje, že neexistuje žádný identifikátor zabezpečení:

MQSID_NONE
Není uveden žádný identifikátor zabezpečení.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQSID_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQSID_NONE, ale je to pole znaků místo řetězce.

Toto je vstupní pole pro ukončení. Délka tohoto pole je dána hodnotou MQ_SECURITY_ID_LENGTH. Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_6.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCD_VERSION_7.

Identifikátor RemoteUser(MQCHAR12)
Toto pole uvádí prvních 12 bajtů identifikátoru uživatele z partnera.

Jsou zde dvě pole, která obsahují identifikátor vzdáleného uživatele:

- *RemoteUserIdentifier* obsahuje prvních 12 bajtů identifikátoru vzdáleného uživatele a je doplněn mezerami, je-li 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 ukončený znakem null. Je-li identifikátor prázdný, *LongRemoteUserIdLength* je nula a hodnota *LongRemoteUserIdPtr* není definovaná.

LongRemoteUserIdPtr není přítomen, pokud *Version* je menší než MQCD_VERSION_6.

Identifikátor vzdáleného uživatele je relevantní pouze pro kanály s *ChannelType* MQCHT_CLNTCONN nebo MQCHT_SVRCONN.

- Pro uživatelskou proceduru zabezpečení u kanálu MQCHT_CLNTCONN je tato hodnota identifikátor uživatele, který byl získán z prostředí. Ukončení se může rozhodnout odeslat ji na konec zabezpečení na serveru.
- Pro uživatelskou proceduru zabezpečení u kanálu MQCHT_SVRCONN může toto pole obsahovat identifikátor uživatele, který byl získán z prostředí na klientovi, pokud neexistuje žádná uživatelská procedura zabezpečení klienta. Uživatelská procedura může ověřit toto ID uživatele (pravděpodobně s heslem v produktu *RemotePassword*) a aktualizovat hodnotu v produktu *MCAUserIdentifier*.

  Pokud na straně klienta existuje uživatelská procedura zabezpečení, lze tyto informace získat v rámci toku zabezpečení od klienta.

Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_2.

SecurityExit (MQCHARn)
Toto pole určuje název uživatelské procedury zabezpečení kanálu.

Je-li tento název neprázdný, 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 odpovědi na tok zpráv zabezpečení.

  Veškeré toky zpráv zabezpečení přijaté ze vzdáleného procesoru na vzdáleném počítači jsou předány k ukončení.
- Při inicializaci a ukončení kanálu.

Viz "MQCD-Definice kanálu" na stránce 1464 , kde najdete popis obsahu tohoto pole v různých prostředích.

Délka tohoto pole je dána proměnnou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro prostředí.

Data SecurityUserData (MQCHAR32)
Tento kanál určuje uživatelská data uživatelské procedury zabezpečení kanálu.

Tato data se předají do uživatelské procedury zabezpečení kanálu v poli *ExitData* parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na začátku obsahuje data, která byla nastavena v definici kanálu. Avšak během doby životnosti této instance MCA jsou všechny změny provedené v obsahu tohoto pole při ukončení libovolného typu zachovány agentem MCA a jsou viditelné pro následná vyvolání ukončení (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy z různých konverzací. Takové změny se neprojeví na definici kanálu použité jinými instancemi MCA. Mohou být použity jakékoliv znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není relevantní pro IBM MQ for IBM i.

SendExit (MQCHARn)
Toto pole uvádí název uživatelské procedury odeslání kanálu.

Je-li tento název neprázdný, je uživatelská procedura volána v následujících časech:

- Okamžitě před odesláním dat v síti.

  Výstupem je dána úplná přenosová vyrovnávací paměť před přenosem. Obsah vyrovnávací paměti lze upravit podle potřeby.

- Při inicializaci a ukončení kanálu.

Viz “MQCD-Definice kanálu” na stránce 1464 , kde najdete popis obsahu tohoto pole v různých prostředích.

Délka tohoto pole je dána proměnnou MQ_EXIT_NAME_LENGTH.

Poznámka: Hodnota této konstanty je specifická pro prostředí.

SendExitPtr (MQPTR)
Toto pole uvádí adresu prvního pole *SendExit* .

Je-li *SendExitsDefined* větší než nula, je tato adresa adresou seznamu názvů všech uživatelských procedur odeslání kanálu v řetězci.

Každý název je v poli o délce *ExitNameLength*vyplněný zprava mezerami. Existuje *SendExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou proceduru.

Jakékoli změny provedené v těchto názvech podle ukončení se zachovají, ačkoli ukončení odeslání zprávy neprovede žádnou explicitní akci-nezmění se, které uživatelské procedury jsou vyvolány.

Je-li *SendExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

SendExitsDefinováno (MQLONG)
Toto pole uvádí počet uživatelských procedur odeslání kanálu definovaných v řetězci.

Je větší než nebo rovno nule.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Data SendUser(MQCHAR32)
Toto pole uvádí uživatelská data ukončení odeslání kanálu.

Tato data jsou předána uživatelské proceduře pro odeslání kanálu do pole *ExitData* v parametru **ChannelExitParms** (viz MQ_CHANNEL_EXIT).

Toto pole na začátku obsahuje data, která byla nastavena v definici kanálu. Avšak během doby životnosti této instance MCA jsou všechny změny provedené v obsahu tohoto pole při ukončení libovolného typu zachovány agentem MCA a jsou viditelné pro následná vyvolání ukončení (bez ohledu na typ) pro tuto instanci MCA. To platí pro východy z různých konverzací. Takové změny nemají vliv na definici kanálu používanou jinými instancemi MCA. Mohou být použity jakékoliv znaky (včetně binárních dat).

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto pole není relevantní pro IBM MQ for IBM i.

SendUserDataPtr (MQPTR)
Toto pole uvádí adresu pole *SendUserData* .

Je-li *SendExitsDefined* větší než nula, je tato adresa adresou seznamu uživatelských datových položek pro každou uživatelskou proceduru zprávy kanálu v řetězci.

Každá uživatelská datová položka je v poli o délce *ExitDataLength*, která je směrem doprava vyplněna mezerami. Existuje *MsgExitsDefined* polí sousedících s jedním dalším-jeden pro každou uživatelskou proceduru. Je-li počet definovaných položek uživatelských dat menší než počet názvů procedur, budou nedefinované datové položky uživatele nastaveny na mezery. Naopak, pokud je počet definovaných položek uživatelských dat větší než počet názvů procedur, přebytečné uživatelské datové položky se budou ignorovat a nebudou představeny k ukončení.

Všechny změny provedené v těchto hodnotách budou zachovány. To umožňuje jedné uživatelské proceduře předat informace dalšímu ukončení. Na žádných změnách se neprovedou žádné ověření, takže binární data lze v případě potřeby zapsat do těchto polí.

Je-li *SendExitsDefined* nula, toto pole je ukazatel null.

Na platformách, v nichž programovací jazyk nepodporuje datový typ ukazatele, je toto pole deklarováno jako bajtový řetězec příslušné délky.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

Zalamovat SeqNumber(MQLONG)
Toto pole uvádí nejvyšší přípustné pořadové číslo zprávy.

Když je tato hodnota dosažena, zalomení se zalomí, aby se znovu spustil na 1.

Tato hodnota je nepřevoditelná a musí odpovídat jak v definici lokálního i vzdáleného kanálu.

Toto pole není relevantní pro kanály s *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 pro kanály připojení klienta a kanály připojení serveru.

Hodnota 0 znamená, že kanál pracuje tak, jak byl 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 hodnotou pro chování produktu IBM WebSphere MQ 7.0 . Přestože je na instanci kanálu povolena pouze jedna konverzace, je k dispozici asynchronní spotřeba a IBM WebSphere MQ 7.0 chování prezenčního signálu `CLNTCONN-SVRCONN` a zastavení kanálu v klidu jsou k dispozici.

Toto je vstupní pole pro ukončení. Není přítomna, pokud *Version* je menší než MQCD_VERSION_9.

Výchozí hodnota tohoto pole je 10.

Poznámka: Limity *MaxInstances* a *MaxInstancesPerClient* použité na kanál omezují počet instancí kanálu, nikoli počet konverzací, které by mohly tyto instance sdílet.

Název ShortConnection(MQCHAR20)
Toto pole uvádí prvních 20 bajtů názvu připojení.

Je-li pole *Version* MQCD_VERSION_1, obsahuje *ShortConnectionName* úplný název připojení.

Je-li pole *Version* MQCD_VERSION_2 nebo vyšší, obsahuje *ShortConnectionName* prvních 20 znaků názvu připojení. Úplný název připojení je uveden v poli *ConnectionName* ; *ShortConnectionName* a prvních 20 znaků *ConnectionName* je identické.

Podrobné informace o obsahu tohoto pole naleznete v příručce *ConnectionName* .

Poznámka: Název tohoto pole byl změněn pro MQCD_VERSION_2 a následné verze produktu MQCD; pole bylo dříve voláno *ConnectionName*.

Délka tohoto pole je dána hodnotou MQ_SHORT_CONN_NAME_LENGTH.

Počet ShortRetryCount (MQLONG)
Toto pole uvádí maximální počet pokusů, které se provedou pro připojení ke vzdálenému počítači.

Toto pole je maximální povolený počet pokusů o připojení ke vzdálenému počítači, v intervalech určených parametrem *ShortRetryInterval*, před použitím (obvykle delších) *LongRetryCount* a *LongRetryInterval* .

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

Interval ShortRetry(MQLONG)
Toto pole uvádí maximální počet sekund, po které se má čekat, než se znovu pokusí o připojení ke vzdálenému počítači.

Interval mezi novými pokusy může být prodloužen, pokud má kanál čekat, než se stane aktivním.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_CLUSSDR nebo MQCHT_CLUSRCVR.

SPLProtection (MQLONG)
Toto pole uvádí hodnotu ochrany zásad zabezpečení produktu AMS .

Hodnota je jedna z následujících možností:

MQSPL_PASSTHRU
Průchod, nezměněný, všechny zprávy odeslané nebo přijaté agentem MCA pro tento kanál.

Tato hodnota je relevantní pouze pro kanály s *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_RECEIVER nebo MQCHT_REQUESTER, a je výchozí hodnotou.

MQSPL_REMOVE
Odeberte jakoukoli ochranu produktu AMS ze zpráv načtených z přenosové fronty agentem MCA a odešlete zprávy partnerovi.

Tato hodnota je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER nebo MQCHT_SERVER.

ZÁSADA 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 *ChannelType* MQCHT_RECEIVER nebo MQCHT_REQUESTER.

Toto je vstupní pole pro ukončení. Toto pole není přítomno, pokud *Version* je menší než MQCD_VERSION_12.

SSLCipherSpec (MQCHAR32)
Toto pole uvádí specifikaci šifry, která se používá při použití TLS.

Je-li hodnota SSLCipherSpec prázdná, kanál nepoužívá TLS. Pokud pole není prázdné, obsahuje toto pole řetězec určující použití CipherSpec .

Tento parametr je platný pro všechny typy kanálů. Tento produkt je podporován na následujících platformách:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows
- 

z/OS

Tento parametr 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 *Version* je menší než MQCD_VERSION_7.

SSLClientAuth (MQLONG)
Toto pole uvádí, zda je požadováno ověření klienta TLS.

Toto pole je relevantní pouze pro definice kanálu SVRCONN.

Je to jedna z následujících hodnot:

POŽADOVÁNO MQSCA_REQUIRED
Je vyžadováno ověření klienta.

MQSCA_OPTIONAL
Ověření klienta je nepovinné.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_7.

SSLPeerNameDélka (MQLONG)
Toto pole uvádí délku názvu partnera TLS, na který ukazuje *SSLPeerNamePtr*, v bajtech.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_7.

SSLPeerNamePtr (MQPTR)
Toto pole uvádí adresu názvu partnera TLS.

Je-li během úspěšného navázání komunikace TLS přijat certifikát, je rozlišující název předmětu certifikátu zkopírován do pole MQCD, ke kterému má přístup SSLPeerNamePtr na konci kanálu, který přijímá certifikát. Přepisuje hodnotu parametru SSLPeerName pro kanál, je-li tato hodnota přítomna v definici kanálu lokálního uživatele. Je-li na tomto konci kanálu zadána uživatelská procedura zabezpečení, získá rozlišující název z certifikátu rovnocenného partnera na serveru MQCD.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_7.

Poznámka: Aplikace uživatelské procedury zabezpečení postavené před vydáním produktu IBM WebSphere MQ 7.1 mohou vyžadovat aktualizaci. Další informace najdete v tématu Uživatelské programy zabezpečení kanálu.

StrucLength (MQLONG)
Toto pole určuje délku struktury MQCD v bajtech.

Délka nezahrnuje žádný z řetězců adresovaných poli ukazatelů obsažených ve struktuře. Hodnota je jedna z následujících možností:

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 V 9.2.0 **MQCD_LENGTH_12**
Délka struktury definice kanálu version-12 .

Následující konstanta uvádí délku aktuální verze:

AKTUÁLNÍ_DÉLKA_MQCD_
Length of current version of channel definition structure.

Poznámka: Tyto konstanty mají hodnoty, které jsou specifické pro prostředí.

Pole není přítomno, pokud *Version* je menší než MQCD_VERSION_4.

TpName (MQCHAR64)
Toto pole uvádí název transakčního programu LU 6.2 .

Toto pole je relevantní pouze v případě, že přenosový protokol (*TransportType*) je MQXPT_LU62a produkt *ChannelType* není MQCHT_SVRCONN nebo MQCHT_RECEIVER.

Toto pole je vždy prázdné na platformách, na kterých jsou místo toho informace obsaženy v objektu na straně komunikace.

Délka tohoto pole je dána hodnotou MQ_TP_NAME_LENGTH.

TransportType (MQLONG)
Toto pole uvádí přenosový protokol, který se má použít.

Hodnota se nekontroluje, pokud byl kanál iniciován z druhého konce.

Je to jedna z následujících hodnot:

MQXPT_LU62
Protokol přenosu 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: Windows, plus klienti IBM MQ , kteří jsou připojeni 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 nemohou být doručeny kanálem, jsou považovány za selhání. Kanál buď zahodí zprávu, nebo kanál skončí, v souladu s nastavením NPMSPEED.

MQUSEDLQ_YES
Když atribut správce front DEADQ poskytuje název fronty nedoručených zpráv, použije se, jinak se chování používá jako pro NO. Hodnota YES je výchozí hodnotou.

UserIdentifier (MQCHAR12)
Toto pole uvádí identifikátor uživatele používaný agentem kanálu zpráv při pokusu o inicializaci zabezpečené relace SNA se vzdáleným agentem kanálu zpráv.

Toto pole může být prázdné pouze v produktu AIX, Linux, and Windowsa je relevantní pouze pro kanály s *ChannelType* MQCHT_SENDER, MQCHT_SERVER, MQCHT_REQUESTER nebo MQCHT_CLNTCONN. V systému z/OStoto pole není relevantní.

Délka tohoto pole je dána hodnotou MQ_USER_ID_LENGTH. Použije se však pouze prvních 10 znaků.

Toto pole není přítomno, je-li *Version* menší než MQCD_VERSION_2.

Verze (MQLONG)
Pole Version určuje nejvyšší číslo verze, které lze 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.

Verze 9 je nejvyšší, když můžete nastavit pole na IBM WebSphere MQ 7.0 a IBM WebSphere MQ 7.0.1 na všech platformách.

MQCD _VERSION_10
Struktura definice kanálu verze 10.

Verze 10 je nejvyšší hodnotou, kterou lze nastavit na IBM WebSphere MQ 7.1 a IBM WebSphere MQ 7.5 na všech platformách.

MQCD _VERSION_11
Struktura definice kanálu verze 11.

Verze 11 je nejvyšší, pokud můžete nastavit pole na IBM MQ 8.0 na všech platformách.

MQCD _VERSION_12
Struktura definice kanálu verze 12.

Verze 12 je nejvyšší, abyste mohli nastavit pole na IBM MQ 9.1.3.

Pole, která existují pouze v novějších verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

MQCD_CURRENT_VERSION
Hodnota nastavená v produktu MQCD_CURRENT_VERSION je aktuální verzí použité struktury definice kanálu.

Hodnota parametru MQCD_CURRENT_VERSION závisí na daném 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 verzi.

Pro IBM WebSphere MQ 7.0 a pozdější verze jsou deklarace MQCD v záhlaví, kopírování a zahrnutí souborů inicializovány na MQCD_VERSION_6. Chcete-li použít další pole MQCD , aplikace musí nastavit číslo verze na MQCD_CURRENT_VERSION. Pokud zapisujete aplikaci, která je přenosná mezi několika prostředími, musíte zvolit verzi, která je podporována ve všech prostředích.

Tip: Když se zavádí nová verze struktury MQCD , rozvržení existující součásti se nezmění. Uživatelská procedura musí zkontrolovat číslo verze. Musí být rovno nebo větší než nejnižší verze, která obsahuje pole, která musí uživatelská procedura použít.

XmitQName (MQCHAR48)
Toto pole uvádí jméno přenosové fronty, ze které jsou zprávy načítány.

Toto pole je relevantní pouze pro kanály s parametrem *ChannelType* MQCHT_SENDER nebo MQCHT_SERVER.

Délka tohoto pole je dána hodnotou MQ_Q_NAME_LENGTH.

Deklarace C

Toto deklarace je deklarací C pro strukturu MQCD.

V 9.2.0

```
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 COBOL

Toto deklarace je deklarací COBOL pro strukturu MQCD.

V 9.2.0

```
** 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)

Toto prohlášení 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 **
```

Deklarace asembleru System/390

Toto prohlášení je deklarací assembleru System/390 pro strukturu MQCD.

V 9.2.0

```
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)
```

Deklarace jazyka Visual Basic

Toto prohlášení je prohlášení o Visual Basicu struktury MQCD.

Ve Visual Basic může být struktura MQCD použita se strukturou MQCNO v 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 měnit pole na disku MQCD. Tyto změny se však obvykle nepodniká, s výjimkou uvedených okolností.

Pokud uživatelský program kanálu změní pole ve struktuře dat MQCD, nová hodnota je obvykle ignorována procesem kanálu produktu IBM MQ . Nová hodnota však zůstane na MQCD a je předávána všem zbývajícím uživatelským procedurám v řetězu ukončení a v libovolné konverzaci sdílející instanci kanálu.

Je-li vlastnost SharingConversations nastavena na hodnotu FALSE ve struktuře MQCXP, lze v závislosti na typu uživatelského programu, typu kanálu a kódu příčiny ukončení provádět změny v určitých polích. Následující tabulka zobrazuje pole, která lze změnit a ovlivňují chování kanálu, a za jakých okolností. Pokud uživatelský program změní jedno z těchto polí za jakýchkoli jiných okolností nebo jakékoliv pole neuvedeno na seznamu, bude nová hodnota ignorována procesem kanálu. Nová hodnota zůstane na MQCD a je předávána všem zbývajícím uživatelským procedurám v řetězu ukončení a v libovolné konverzaci sdílející instanci kanálu.

Jakýkoliv typ ukončovacího programu při volání inicializace (MQXR_INIT) může změnit pole ChannelName libovolného typu kanálu, pokud je MQCXP SharingConverstions nastaveno na hodnotu FALSE. Pouze uživatelská procedura zabezpečení může změnit pole MCAUserIdentifier bez ohledu na hodnotu MQCXP SharingConverstions.

Tabulka 824. Pole, která lze měnit a ovlivňují chování kanálu.

Pole	**Výstupní kód příčiny**	**Typ ukončení**	**Typ kanálu**
ChannelName	FUNKCE MQXR_INIT	Vše	Vše
TransportType	FUNKCE MQXR_INIT	Vše	Vše
XmitQName	FUNKCE MQXR_INIT	Vše	SDR, RCVR
ModeName	FUNKCE MQXR_INIT	Vše	Vše
TpName	FUNKCE MQXR_INIT	Vše	Vše
BatchSize	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
DiscInterval	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR

Tabulka 824. Pole, která lze měnit a ovlivňují chování kanálu. (pokračování)			
Pole	**Výstupní kód příčiny**	**Typ ukončení**	**Typ kanálu**
Počet ShortRetry	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
Interval ShortRetry	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
Počet LongRetry	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
Interval LongRetry	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
SeqNumberObtékání textu	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
MaxMsgLength	FUNKCE MQXR_INIT	Vše	Vše
PutAuthority	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
DataConversion	FUNKCE MQXR_INIT	Vše	Vše
MCAUserIdentifier	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Zabezpečení	RCVR, RQSTR, SVRCONN, CLURCVR
ConnectionName	FUNKCE MQXR_INIT	Vše	SDR, SVR, RQSTR, CLNTCONN, CLUSDSR, CLURCVR
MsgRetryUserData	FUNKCE MQXR_INIT	Vše	RCVR, RQSTR, CLURCVR
Počet MsgRetry	FUNKCE MQXR_INIT	Vše	RCVR, RQSTR, CLURCVR

Tabulka 824. Pole, která lze měnit a ovlivňují chování kanálu. (pokračování)			
Pole	**Výstupní kód příčiny**	**Typ ukončení**	**Typ kanálu**
Interval MsgRetry	FUNKCE MQXR_INIT	Vše	RCVR, RQSTR, CLURCVR
HeartbeatInterval	FUNKCE MQXR_INIT	Vše	Vše
BatchInterval	FUNKCE MQXR_INIT	Vše	SDR, SVR, CLUSDSR, CLURCVR
NonPersistentMsgSpeed	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
MCASecurityId	MQXR_INIT, MQXR_INIT_SEC, MQXR_SEC_MSG, MQXR_SEC_PARMS	Zabezpečení	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSDSR, CLURCVR
SSLCipherSpec	FUNKCE MQXR_INIT	Vše	Vše
SSLPeerNamePtr	FUNKCE MQXR_INIT	Vše	Vše
SSLPeerNameDélka	FUNKCE MQXR_INIT	Vše	Vše
SSLClientAuth	FUNKCE MQXR_INIT	Vše	SVR, RCVR, RQSTR, SVRCONN, CLURCVR
KeepAliveInterval	FUNKCE MQXR_INIT	Vše	Vše
LocalAddress	FUNKCE MQXR_INIT	Vše	SDR, SVR, RQSTR, CLNTCONN, CLUSDSR, CLURCVR
BatchHeartbeat	FUNKCE MQXR_INIT	Vše	SDR, SVR, CLUSDSR, CLURCVR
Seznam HdrComp	FUNKCE MQXR_INIT	Vše	Vše
Seznam MsgComp	FUNKCE MQXR_INIT	Vše	Vše
ChannelMonitoring	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, SVRCONN, CLUSDSR, CLURCVR

Tabulka 824. Pole, která lze měnit a ovlivňují chování kanálu. (pokračování)			
Pole	**Výstupní kód příčiny**	**Typ ukončení**	**Typ kanálu**
ChannelStatistics	FUNKCE MQXR_INIT	Vše	SDR, SVR, RCVR, RQSTR, CLUSDSR, CLURCVR
SharingConversations	FUNKCE MQXR_INIT	Vše	SVRCONN, CLNTCONN
PropertyControl	FUNKCE MQXR_INIT	Vše	SDR, SVR, CLUSDSR, CLURCVR

MQCXP-Výstupní parametr kanálu

Struktura MQCXP je předávána každému typu uživatelské procedury volaného agentem MCA (Message Channel Agent), kanálem připojení klienta nebo kanálu připojení serveru.

Viz MQ_CHANNEL_EXIT.

Pole označená jako "vstup do výstupního bodu" v popisech, které následují za znakem, jsou kanálem ignorována, když uživatelská procedura vrátí řízení kanálu. Všechna vstupní pole, která se změní v bloku parametrů ukončení kanálu, nebudou pro další vyvolání zachována. Změny vstupních a výstupních polí (například pole *ExitUserArea*) jsou zachovány pro vyvolání této instance pouze pro ukončení. Tyto změny nelze použít k předávání dat mezi různými uživatelskými procedurami definovanými na stejném kanálu nebo mezi stejnou uživatelskou procedurou definovanou v různých kanálech.

Související odkazy

“Pole” na stránce 1504
Toto téma uvádí všechna pole v rámci struktury MQCXP a popisuje každé pole.

“Deklarace C” na stránce 1515
Toto prohlášení je prohlášení C pro strukturu MQCXP.

“Deklarace COBOL” na stránce 1516
Toto deklarace je deklarací COBOL pro strukturu MQCXP.

“Deklarace RPG (ILE)” na stránce 1517
Toto prohlášení je deklarací RPG pro strukturu MQCXP.

“Deklarace asembleru System/390” na stránce 1518
Toto prohlášení je deklarací assembleru System/390 pro strukturu MQCXP.

Pole

Toto téma uvádí všechna pole v rámci struktury MQCXP a popisuje každé pole.

StrucId (MQCHAR4)
Toto pole uvádí identifikátor struktury.

Hodnota musí být:

ID_STRUKTURY MQCXP_STRUCTION_ID
Identifikátor pro strukturu parametru uživatelské procedury kanálu.

Pro programovací jazyk C je také definována konstanta MQCXP_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako MQCXP_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vstupní pole pro ukončení.

Verze (MQLONG)
Toto pole uvádí číslo verze struktury.

Hodnota závisí na prostředí:

MQCXP_VERSION_1
Struktura parametru ukončení kanálu Version-1 .

MQCXP_VERSION_3
Struktura výstupního parametru kanálu Version-3 .

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 výstupního parametru kanálu Version-5 .

MQCXP_VERSION_6
Struktura parametru ukončení kanálu Version-6 .

MQCXP_VERSION_8
Struktura parametrů kanálu Version-8 .

Pole má tuto hodnotu v z/OS.

MQCXP_VERSION_9
Struktura parametrů kanálu Version-9 .

Pole má tuto hodnotu v následujících prostředích:

- 

AIX
- 

IBM i
- 

Linux
- 

Windows
- 

z/OS

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCXP_CURRENT_VERSION
Aktuální verze struktury výstupního parametru kanálu.

Hodnota závisí na prostředí.

Poznámka: Je-li zavedena nová verze struktury MQCXP, rozvržení existující součásti se nezmění. Uživatelská procedura musí proto zkontrolovat, zda je číslo verze rovné nebo větší než nejnižší verze, která obsahuje pole, která má uživatelská procedura použít.

Toto je vstupní pole pro ukončení.

ExitId (MQLONG)
Toto pole uvádí typ volané procedury a je nastaven na vstupu do uživatelské procedury.

Možné jsou následující hodnoty:

MQXT_CHANNEL_SEC_EXIT
Ukončení zabezpečení kanálu.

MQXT_CHANNEL_MSG_EXIT
Ukončení zprávy kanálu.

UŽIVATELSKÁ PROCEDURA MQXT_CHANNEL_SEND_EXIT
Ukončení odeslání kanálu.

UKONČOVACÍ PROCEDURA MQXT_CHANNEL_RCV_EXIT
Ukončení příjmu kanálu.

MQXT_CHANNEL_MSG_RETRY_EXIT
Zpráva kanálu-ukončení 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 uvádí důvod, proč se procedura volá a je nastavena na vstupu do uživatelské procedury.

Nepoužívá se pro ukončení automatické definice. Možné jsou následující hodnoty:

FUNKCE MQXR_INIT
Ukončete inicializaci.

Tato hodnota označuje, že je ukončení vyvoláno poprvé. Umožňuje ukončení získat a inicializovat všechny prostředky, které potřebuje (například: paměť).

VÝRAZ MQXR_
Ukončit ukončení.

Tato hodnota označuje, že ukončení se chystá ukončit. Ukončení by mělo uvolnit všechny prostředky, které získala od své inicializace (například: paměť).

ZPRÁVA MQXR_MSG
Zpracovat zprávu.

Tato hodnota označuje, že uživatelská procedura je vyvolávána ke zpracování zprávy. Tato hodnota se vyskytne pouze pro uživatelské procedury kanálu zprávy.

MQXR_XMIT
Zpracovat přenos.

Tato hodnota se vyskytuje pouze u kanálů odeslání a příjmu kanálu.

Z_ZPR_ZA_ZPRÃ
Byla přijata zpráva zabezpečení.

Tato hodnota se vyskytne pouze pro uživatelské procedury zabezpečení kanálu.

MQXR_INIT_SEC
Zahajte výměnu zabezpečení.

Tato hodnota se vyskytne pouze pro uživatelské procedury zabezpečení kanálu.

Ukončení zabezpečení zásobníku je vždy vyvoláno touto příčinou okamžitě po vyvolání s funkcí MQXR_INIT, aby mu bylo umožněno zahájit výměnu zabezpečení. Pokud odmítne příležitost (vrátí MQXCC_OK místo MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_REQUEST_SEC_MSG), bude uživatelská procedura zabezpečení odesílatele vyvolána s funkcí MQXR_INIT_SEC.

Pokud uživatelská procedura zabezpečení příjemce zahájí výměnu zabezpečení (vrácením MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_REQUEST_SEC_MSG), nebude uživatelská procedura zabezpečení odesílatele nikdy vyvolána s parametrem MQXR_INIT_SEC; místo toho je vyvolána s příkazem MQXR_SEC_MSG ke zpracování zprávy přijímače. (V každém případě je nejprve vyvolán s MQXR_INIT.)

Pokud některý z uživatelských procedur zabezpečení nevyžaduje ukončení kanálu (nastavením parametru *ExitResponse* na hodnotu MQXCC_SUPPRESS_FUNCTION nebo MQXCC_CLOSE_CHANNEL), musí být na straně, která iniciovala výměnu, dokončena výměna zabezpečení. Proto je-li uživatelská procedura zabezpečení vyvolána s funkcí MQXR_INIT_SEC a zahájí výměnu, bude při příštím vyvolání této uživatelské procedury použita hodnota

MQXR_SEC_MSG. Tato situace nastane, pokud dojde ke zprávě zabezpečení pro ukončení procesu nebo ne. Existuje zpráva zabezpečení, pokud partner vrátí MQXCC_SEND_SEC_MSG nebo MQXCC_SEND_AND_REQUEST_SEC_MSG, ale ne, pokud partner vrátí MQXCC_OK nebo neexistuje žádná uživatelská procedura pro zabezpečení zprávy. Pokud neexistuje žádná zpráva zabezpečení ke zpracování, uživatelská procedura zabezpečení na zahajovacím konci je znovu vyvolána s hodnotou *DataLength* nula.

MQXR_RETRY
Zopakovat zprávu.

Tato hodnota se vyskytne pouze pro ukončení opakování zprávy.

MQXR_AUTO_CLUSDR
Automatická definice kanálu odesílatele klastru.

Tato hodnota se vyskytne pouze pro uživatelské procedury automatické definice kanálu.

MQXR_AUTO_RECEIVER
Automatická definice přijímacího kanálu.

Tato hodnota se vyskytne pouze pro uživatelské procedury automatické definice kanálu.

FUNKCE MQXR_AUTO_SVRCONN
Automatická definice kanálu připojení serveru.

Tato hodnota se vyskytne pouze pro uživatelské procedury automatické definice kanálu.

SOUBOR MQXR_AUTO_CLURCVR
Automatická definice přijímacího kanálu klastru.

Tato hodnota se vyskytne pouze pro uživatelské procedury automatické definice kanálu.

MQXR_SEC_PARMS
Parametry zabezpečení

Tato hodnota se vztahuje pouze k uživatelským procedurám zabezpečení a určuje, že do uživatelské procedury je předávána struktura MQCSP. Další informace naleznete v tématu "MQCSP-parametry zabezpečení" na stránce 336

Poznámka:

1. Máte-li pro kanál definován více než jednu uživatelskou proceduru, jsou při inicializaci inicializovaného agenta MCA vyvolány příkazy MQXR_INIT při volání MQXR_INIT. Také jsou vyvolány příkazem MQXR_TERM, když je agent MCA ukončen.
2. Pro uživatelskou proceduru automatické definice kanálu není produkt *ExitReason* nastaven, pokud je *Version* menší než MQCXP_VERSION_4. V tomto případě je odvozena hodnota MQXR_AUTO_SVRCONN.

Toto je vstupní pole pro ukončení.

ExitResponse (MQLONG)
Toto pole uvádí odpověď z ukončení.

Toto pole je nastaveno uživatelskou procedurou pro komunikaci s agentem MCA. Musí se jednat o jednu z následujících hodnot:

MQXCC_OK
Ukončení bylo úspěšně dokončeno.

- Pro proceduru zabezpečení kanálu tato hodnota označuje, že přenos zpráv může nyní pokračovat normálně.
- Pro ukončení opakování zprávy kanálu tato hodnota označuje, že agent MCA musí čekat na časový interval vrácený uživatelskou procedurou v poli *MsgRetryInterval* v produktu MQCXP a poté se pokusit o zprávu znovu.

Pole *ExitResponse2* může obsahovat další informace.

FUNKCE MQXCC_SUPPRESS_FUNCTION
Potlačit funkci.

- Pro uživatelskou proceduru zabezpečení kanálu tato hodnota označuje, že kanál musí být ukončen.
- Pro uživatelskou proceduru pro zprávy kanálu tato hodnota označuje, že zpráva nemá pokračovat ve směrování na místo určení. Místo toho agent MCA vygeneruje zprávu hlášení o výjimce (pokud byla požadována odesilatelem původní zprávy) a umístí zprávu obsaženou v původní vyrovnávací paměti do fronty nedoručených zpráv (pokud odesílatel uvedl MQRO_DEAD_LETTER_Q), nebo ji zahodí (pokud odesílatel uvedl MQRO_DISCARD_MSG).

  Pro trvalé zprávy, pokud odesilatel uvedl MQRO_DEAD_LETTER_Q, ale vložení do fronty nedoručených zpráv selže nebo ve frontě není žádná fronta nedoručených zpráv, je původní zpráva ponechána v přenosové frontě a zpráva sestavy se negeneruje. Pokud zpráva sestavy nemůže být úspěšně generována, je původní zpráva v přenosové frontě ponechána také.

  Pole *Feedback* ve struktuře MQDLH na začátku zprávy ve frontě nedoručených zpráv indikuje, proč byla zpráva vložena do fronty nedoručených zpráv; tento kód zpětné vazby je také použit v deskriptoru zprávy zprávy hlášení výjimek (pokud byl vyžádán odesílatelem).
- Pro ukončení opakování zprávy kanálu tato hodnota označuje, že agent MCA nečeká a zopakujte zprávu; místo toho bude program MCA pokračovat okamžitě s normálním zpracováním selhání (zpráva se umístí do fronty nedoručených zpráv nebo je vyřazena, jak je uvedeno odesilatelem zprávy).
- Pro uživatelskou proceduru automatické definice kanálu musí být zadán buď MQXCC_OK, nebo MQXCC_SUPPRESS_FUNKCE. Není-li zadána žádná z těchto hodnot, předpokládá se výchozí hodnota MQXCC_SUPPRESS_FUNKCE a automaticky zrušena definice auto-definition.

Tato odezva není u uživatelských procedur pro odeslání a příjem kanálu podporována.

MQXCC_SEND_SEC_MSG
Odeslat bezpečnostní zprávu.

Tuto hodnotu lze nastavit pouze prostřednictvím uživatelské procedury zabezpečení kanálu. Označuje, že uživatelská procedura poskytla zprávu o zabezpečení, která musí být předána partnerovi.

MQXCC_SEND_AND_REQUEST_SEC_MSG
Odeslat bezpečnostní zprávu, která vyžaduje odpověď.

Tuto hodnotu lze nastavit pouze prostřednictvím uživatelské procedury zabezpečení kanálu. Označuje

- že vyplutí poskytla zprávu o zabezpečení, kterou lze předat partnerovi, a
- , že uživatelská procedura vyžaduje odpověď od partnera. Není-li přijata žádná odezva, kanál musí být ukončen, protože uživatelská procedura ještě nerozhodla, zda komunikace může pokračovat.

UŽIVATELSKÁ PROCEDURA MQXCC_SUPPRESS_EXIT
Potlačit ukončení.

- Tato hodnota může být nastavena všemi typy jiné uživatelské procedury kanálu, než je uživatelská procedura zabezpečení nebo uživatelská procedura automatické definice. Potlačuje jakékoliv další vyvolání této uživatelské procedury (jako by jeho název byl prázdný v definici kanálu) až do ukončení kanálu, je-li ukončení znovu vyvoláno s *ExitReason* MQXR_TERM.
- Pokud uživatelská procedura opakování zprávy vrátí tuto hodnotu, opakování zpráv pro následující zprávy jsou řízeny atributy kanálu *MsgRetryCount* a *MsgRetryInterval* jako normální. Pro aktuální zprávu agent MCA provádí počet neprovedených opakovaných pokusů, v intervalech daných atributem kanálu *MsgRetryInterval* , ale pouze v případě, že se jedná o kód příčiny, který program MCA obvykle opakuje (viz pole *MsgRetryCount* popsané v části “MQCD-Definice kanálu” na stránce 1464). Počet neprovedených opětovných pokusů je hodnotou atributu **MsgRetryCount** , minus počet případů, kdy byla ukončena operace MQXCC_OK pro aktuální zprávu; je-li toto číslo záporné, neprovede se pro aktuální zprávu žádné další pokusy o další pokusy.

MQXCC_CLOSE_CHANNEL
Zavřete kanál.

Tato hodnota může být nastavena libovolným typem uživatelské procedury kanálu s výjimkou procedury automatické definice.

Pokud sdílení konverzací není povoleno, tato hodnota zavře kanál.

Je-li povoleno sdílení konverzací, tato hodnota ukončí konverzaci. Je-li tato konverzace jedinou konverzací na kanálu, kanál se také zavře.

Toto pole je vstupní/výstupní pole z ukončení.

ExitResponse2 (MQLONG)
Toto pole uvádí sekundární odpověď z uživatelské procedury.

Toto pole je nastaveno na nulu při vstupu do uživatelské procedury. Může být nastaven pomocí uživatelské procedury pro poskytnutí dalších informací o funkcích kanálu produktu IBM MQ . Nepoužívá se pro ukončení automatické definice.

Uživatelská procedura může nastavit jednu nebo více z následujících hodnot. Je-li požadováno více než jedno, jsou přidány hodnoty. Kombinace, které nejsou platné, jsou zaznamenány; jiné kombinace jsou povoleny.

MQXR2_PUT_WITH_DEF_ACTION
Vložit s výchozí akcí.

Tato hodnota je nastavena uživatelskou procedurou zprávy kanálu pro příjemce. Označuje, že má být zpráva vložena s výchozí akcí agenta MCA, která je buď výchozí ID uživatele MCA, nebo kontext *UserIdentifier* v deskriptoru zpráv MQMD (Message Descriptor).

Hodnota je nula, což odpovídá počáteční hodnotě nastavené při vyvolání uživatelské procedury. Konstanta je k dispozici pro účely dokumentace.

MQXR2_PUT_WITH_DEF_USERID
Vložit s výchozím identifikátorem uživatele.

Tato hodnota může být nastavena pouze ukončením zprávy kanálu příjemce. Označuje, že zpráva má být vložena s výchozím identifikátorem uživatele MCA.

MQXR2_PUT_WITH_MSG_USERID
Zadejte identifikátor uživatele pro zprávu.

Tato hodnota může být nastavena pouze ukončením zprávy kanálu příjemce. Označuje, že zpráva má být vložena do kontextu *UserIdentifier* v deskriptoru zpráv MQMD (deskriptor zprávy) zprávy (byla by tato změna pravděpodobně modifikována uživatelskou procedurou).

Je třeba nastavit pouze jednu z položek 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 všechna data, která mají být předána, jsou v *AgentBuffer*, nikoli *ExitBufferAddr*.

Hodnota je nula, což odpovídá počáteční hodnotě nastavené při vyvolání uživatelské procedury. Konstanta je k dispozici pro účely dokumentace.

MQXR2_USE_EXIT_BUFFER
Použít výstupní vyrovnávací paměť.

Tato hodnota označuje, že všechna data, která mají být předána, jsou v *ExitBufferAddr*, nikoli *AgentBuffer*.

Měla by být nastavena pouze jedna z položek MQXR2_USE_AGENT_BUFFER a MQXR2_USE_EXIT_BUFFER .

MQXR2_DEFAULT_CONTINUATION
Výchozí pokračování.

Pokračování s dalším výstupem v řetězci závisí na odezvě od posledního vyvolaného ukončení:

- Je-li vrácen parametr MQXCC_SUPPRESS_FUNCTION nebo MQXCC_CLOSE_CHANNEL, nebudou volány žádné další uživatelské procedury v řetězci.
- Jinak se vyvolá další ukončení v řetězci.

MQXR2_CONTINUE_CHAIN
Pokračujte s další uživatelskou procedurou.

MQXR2_SUPPRESS_CHAIN
Přeskočení zbývajících uživatelských procedur v řetězu.

Jedná se o vstupní/výstupní pole pro ukončení.

Zpětná vazba (MQLONG)
Toto pole uvádí 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_FUNVOD, pole *Feedback* určuje kód zpětné vazby, který identifikuje, proč byla zpráva vložena do fronty nedoručených zpráv (nedoručená zpráva), a také se používá k odeslání zprávy o výjimce, pokud byla požadována. V tomto případě, je-li 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 pro zabezpečení kanálu, odeslání, přijetí a ukončení opakování zprávy není používána agentem MCA.

Hodnota vrácená v tomto poli uživatelskou procedurou automatické definice se nepoužívá, pokud *ExitResponse* je MQXCC_OK, ale jinak se použije pro parametr *AuxErrorDataInt1* ve zprávě události.

Jedná se o vstupní/výstupní pole z uživatelské procedury.

MaxSegmentDélka (MQLONG)
Toto pole uvádí maximální délku v bajtech, kterou lze odeslat v jednom přenosu.

Nepoužívá se pro ukončení automatické definice. Je předmětem zájmu o uživatelskou proceduru odeslání zprávy kanálu, protože tato uživatelská procedura musí zajistit, že velikost přenosového segmentu nebude zvětšovat na hodnotu větší než *MaxSegmentLength*. Délka zahrnuje prvních 8 bajtů, které ukončení nesmí změnit. Hodnota se vyjednává mezi funkcemi kanálu produktu IBM MQ při inicializaci kanálu. Další informace o délkách segmentů najdete v tématu Psaní ukončovacích programů kanálů .

Hodnota v tomto poli není smysluplná, pokud *ExitReason* je MQXR_INIT.

Toto je vstupní pole pro ukončení.

Oblast ExitUser(MQBYTE16)
Toto pole uvádí oblast uživatelských procedur-pole dostupné pro ukončení, které se má použít.

Inicializuje se na binární nulu před prvním vyvoláním uživatelské procedury (která má sadu *ExitReason* nastavenou na hodnotu MQXR_INIT) a poté všechny změny provedené v tomto poli při ukončení budou zachovány mezi vyvoláními ukončení.

Je definována následující hodnota:

MQXA_NONE
Žádné informace o uživateli.

Hodnota je binární nula pro délku pole.

Pro programovací jazyk C je také definována konstanta MQXUA_NONE_ARRAY; tato konstanta má stejnou hodnotu jako MQXUA_NONE, ale je to pole znaků místo řetězce.

Délka tohoto pole je dána proměnnou MQ_EXIT_USER_AREA_LENGTH. Jedná se o vstupní/výstupní pole pro ukončení.

ExitData (MQCHAR32)
Toto pole uvádí výstupní data.

Toto pole je nastaveno na vstupu do uživatelské procedury na informace, které funkce kanálu produktu IBM MQ převzaly z definice kanálu. Nejsou-li takové informace k dispozici, bude toto pole prázdné.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Toto je vstupní pole pro ukončení.

Následující pole v této struktuře nejsou k dispozici, pokud je produkt *Version* menší než hodnota MQCXP_VERSION_2.

Počet MsgRetry(MQLONG)
Toto pole uvádí, kolikrát byla zpráva zopakována.

Při prvním vyvolání uživatelské procedury pro konkrétní zprávu má toto pole hodnotu nula (zatím nebyly provedeny žádné pokusy). Při každém dalším vyvolání uživatelské procedury pro tuto zprávu se hodnota zvýší o jednu podle agenta MCA.

Toto je vstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud *ExitReason* je MQXR_INIT. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_2.

Interval MsgRetryInterval (MQLONG)
Toto pole určuje minimální interval v milisekundách, po jehož uplynutí dojde k opakovaný pokus o operaci vložení.

Při prvním vyvolání uživatelské procedury pro konkrétní zprávu bude toto pole obsahovat hodnotu atributu kanálu *MsgRetryInterval* . Uživatelská procedura může ponechat hodnotu nezměněnou nebo ji upravit tak, aby určoval jiný časový interval v milisekundách. Pokud funkce uživatelské procedury MQXCC_OK v produktu *ExitResponse*vrátí MQXCC_OK, program MCA čeká alespoň tento časový interval, než se znovu pokusí o operaci MQOPEN nebo MQPUT. Uvedený časový interval musí být nula nebo větší.

Druhé a následující časy jsou vyvolány pro tuto zprávu, toto pole obsahuje hodnotu vrácenou při předchozím vyvolání uživatelské procedury.

Je-li hodnota vrácená v poli *MsgRetryInterval* menší než nula nebo větší než 999 999 999 a *ExitResponse* je MQXCC_OK, program MCA ignoruje pole *MsgRetryInterval* v MQCXP a čeká místo na interval určený atributem kanálu produktu *MsgRetryInterval* .

Jedná se o vstupní/výstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud *ExitReason* je MQXR_INIT. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_2.

Příčina MsgRetry(MQLONG)
Toto pole uvádí 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 *ExitReason* je MQXR_INIT. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_2.

Následující pole v této struktuře nejsou k dispozici, pokud je produkt *Version* menší než hodnota MQCXP_VERSION_3.

HeaderLength (MQLONG)
Toto pole uvádí délku informací záhlaví.

Toto pole je relevantní pouze pro uživatelskou proceduru zprávy a pro ukončení opakování zprávy. Hodnota je délka struktury záhlaví směrování na začátku dat zprávy; jedná se o strukturu MQXQH, záhlaví MQMDE (záhlaví rozšíření popisu zprávy) a (pro zprávu distribuční seznam) strukturu MQDH a pole záznamů MQOR a MQPMR, které postupují podle struktury MQXQH.

Uživatelská procedura pro zprávy může zkontrolovat informace o tomto záhlaví a v případě potřeby ji upravit, ale data, která vrací ukončení, musí být stále ve správném formátu. Ukončení nesmí například

šifrovat nebo komprimovat data hlavičky na odesílajícím konci, a to i v případě, že ukončení zprávy na přijímajícím konci provádí kompenzaci změn.

Pokud uživatelská procedura pro zprávy upraví informace o záhlaví způsobem, který mění jeho délku (například přidáním dalšího místa určení do zprávy distribučního seznamu), musí před vrácením změnit hodnotu *HeaderLength* .

Jedná se o vstupní/výstupní pole pro ukončení. Hodnota v tomto poli není smysluplná, pokud *ExitReason* je MQXR_INIT. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

PartnerName (MQCHAR48)
Toto pole uvádí jméno partnera.

Název partnera, jak je uvedeno dále:

- Pro kanály SVRCONN se jedná o ID přihlášeného uživatele na straně klienta.
- U všech ostatních typů kanálů se jedná o název správce front partnera.

Je-li procedura inicializována, je toto pole prázdné, protože správce front nezná jméno partnera, dokud nedojde k počátečnímu vyjednávání.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

Úroveň FAPLevel (MQLONG)
Dohodnuté formáty a úroveň protokolů.

Toto je vstupní pole pro ukončení. Změny v tomto poli by měly být provedeny pouze ve směru služby IBM . Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

CapabilityFlags (MQLONG)
Příznak schopnosti lze nastavit na hodnotu MQCF_NONE nebo MQCF_DIST_LISTS.

Můžete nastavit jednu z následujících příznaků schopností:

MQCF_NONE
Žádné vlajky.

MQCF_DIST_LISTS
Podporované seznamy distribucí.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

ExitNumber (MQLONG)
Toto pole určuje pořadové číslo uživatelské procedury.

Pořadové číslo uživatelské procedury, v rámci typu definovaného v produktu *ExitId*. Je-li například vyvolávaná procedura třetí výstupní zprávou o ukončení zprávy, obsahuje toto pole hodnotu 3. Je-li typ ukončení jeden, pro který nelze definovat seznam uživatelských procedur (například uživatelská procedura zabezpečení), má toto pole hodnotu 1.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

Následující pole v této struktuře nejsou přítomna, pokud *Version* je menší než MQCXP_VERSION_5.

ExitSpace (MQLONG)
Toto pole uvádí počet bajtů v přenosové vyrovnávací paměti vyhrazené pro ukončení, které se má použít.

Toto pole je relevantní pouze pro uživatelskou proceduru odeslání. Určuje množství prostoru v bajtech, které funkce kanálu IBM MQ vyhrazuje v přenosové vyrovnávací paměti pro použití k ukončení. Toto pole umožňuje uživatelské proceduře přidat do vyrovnávací paměti přenosové vyrovnávací paměti malé množství dat (obvykle nepřesahujících několik set bajtů) pro použití komplementární přijímací uživatelskou procedurou na druhém konci. Data přidaná uživatelskou procedurou odeslání musí být odebrána uživatelskou procedurou pro přijetí zprávy.

Hodnota je vždy nula v z/OS.

Poznámka: Toto zařízení nesmí být používáno k odesílání velkého množství dat, protože by mohlo dojít ke snížení výkonu, nebo dokonce k zastavení činnosti kanálu.

Při nastavení *ExitSpace* je ukončení garantováno, že v přenosové vyrovnávací paměti je vždy k dispozici alespoň takový počet bajtů, které má uživatelská procedura použít. Uživatelská procedura však může používat méně než rezervovanou částku nebo více než rezervovanou částku, pokud je v přenosové vyrovnávací paměti k dispozici dostatek místa. Výstupní prostor ve vyrovnávací paměti je poskytován za použití stávajících dat.

ExitSpace lze nastavit uživatelskou procedurou pouze v případě, že *ExitReason* má hodnotu MQXR_INIT; ve všech ostatních případech je hodnota vrácená uživatelskou procedurou ignorována. On input to the exit, *ExitSpace* is zero for the MQXR_INIT call, and is the value returned by the MQXR_INIT call in other cases.

Je-li hodnota vrácená voláním MQXR_INIT záporná nebo je k dispozici méně než 1024 bajtů dostupné v přenosové vyrovnávací paměti pro data zprávy po vyhrazení požadovaného výstupního prostoru pro všechny uživatelské procedury odeslání v řetězci, odešle agent MCA chybovou zprávu a zavře kanál. Podobně, pokud během přenosu dat konce výstupního řetězce odeslání alokuje více prostoru uživatele, než je rezervováno, tak, aby v přenosové vyrovnávací paměti pro data zprávy zůstalo méně než 1024 bajtů, odešle agent MCA chybovou zprávu a zavře kanál. Limit 1024 umožňuje řídicí a administrativní toky kanálu, které mají být zpracovány řetězcem ukončení odeslání, bez nutnosti segmentování toků.

Jedná se o vstupní/výstupní pole pro uživatelskou proceduru, je-li *ExitReason* MQXR_INIT, a vstupní pole ve všech ostatních případech. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_5.

ID SSLCertUser(MQCHAR12)
Toto pole uvádí UserId přidružené ke vzdálenému certifikátu.

Je prázdný na všech platformách kromě z/OS

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

SSLRemCertIssNameDélka (MQLONG)
Toto pole uvádí délku úplného rozlišujícího názvu vydavatele vzdáleného certifikátu, na který ukazuje SSLCertRemoteIssuerNamePtr.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6. Hodnota je nula, pokud se nejedná o kanál TLS.

SSLRemCertIssNamePtr (PMQVOID)
Toto pole uvádí adresu úplného rozlišujícího názvu vydavatele vzdáleného certifikátu.

Jeho hodnotou je ukazatel Null, pokud se nejedná o kanál TLS.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

Poznámka: Chování zabezpečení kanálu se ukončí při určování rozlišujícího názvu subjektu a rozlišující název vydávajícího se změní z IBM WebSphere MQ 7.1. Další informace najdete v tématu Uživatelské programy zabezpečení kanálu.

SecurityParms (PMQCSP)
Toto pole uvádí adresu struktury MQCSP použité k uvedení ID uživatele a hesla.

Počáteční hodnota tohoto pole je ukazatel Null.

Jedná se o vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

Hodnota v tomto poli vrácená uživatelskou procedurou musí být použitelná od IBM MQ do MQXR_TERM.

Komprese CurHdr(MQLONG)
Toto pole určuje, která technika se v současné době používá ke kompresi dat záhlaví.

Je nastavena na jednu z následujících možností:

MQCOMPRESS_NONE
Neprovádí se žádná komprese dat hlavičky.

SYSTÉM MQCOMPRESS_SYSTEM
Provádí se komprese dat hlavičky.

Hodnotu lze změnit odesláním zprávy kanálu odesílání do jedné z vyjednaných podporovaných hodnot, ke kterým se přistupuje z pole Seznam HdrCompna disku MQCD. To umožňuje techniku, která se používá ke kompresi dat záhlaví, která se mají zvolit 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 je ignorována, pokud je změněna mimo uživatelskou proceduru odeslání zprávy kanálu.

Jedná se o vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

Komprese CurMsg(MQLONG)
Toto pole uvádí, která technika se v současné době používá ke kompresi dat zprávy.

Je nastavena na jednu z následujících možností:

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 odesláním uživatelské procedury odesílajícího kanálu do jedné z vyjednaných podporovaných hodnot, ke kterým se přistupuje z pole MsgCompList na serveru MQCD. To umožňuje techniku, která se používá ke kompresi dat zprávy, aby se rozhodovalo 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 je ignorována, pokud je změněna mimo uživatelskou proceduru odeslání zprávy kanálu.

Jedná se o vstupní/výstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

Připojení Hconn (MQHCONN)
Toto pole určuje manipulátor připojení, který uživatelská procedura používá v případě, že je třeba provést veškerá volání MQI v rámci uživatelské procedury.

Toto pole není důležité pro ukončení spuštěné v kanálech připojení klienta, kde obsahuje hodnotu MQHC_UNUSABLE_HCONN (-1).

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_7.

SharingConversations (MQBOOL)
Toto pole uvádí, zda je konverzace jediná, která může být momentálně spuštěna na této instanci kanálu, nebo zda momentálně může být spuštěna více než jedna konverzace na této instanci kanálu.

Také označuje, zda je uživatelský program vystaven riziku, že MQCD je měněno jiným výstupním programem spuštěným ve stejnou dobu.

Toto pole je relevantní pouze pro výstupní programy spuštěné v kanálech připojení klienta nebo serveru.

Je nastavena na jednu z následujících možností:

NEPRAVDA
Instance uživatelské procedury je jediná instance ukončení, která může být momentálně spuštěna na této instanci kanálu. To umožní ukončení bezpečně aktualizovat pole MQCD bez soupeření z jiných východů spuštěných na jiných instancích kanálu. Whether changes to the MQCD fields are acted upon

by the channel is defined by the table of MQCD fields in "Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1501.

PRAVDA
Instance uživatelské procedury není jediná instance ukončení, která může být momentálně spuštěna na této instanci kanálu. Veškeré změny provedené v produktu MQCD nejsou zpracovávány kanálem, s výjimkou změn uvedených v tabulce polí MQCD v produktu "Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1501 for Exit Resons other than MQXR_INIT. Pokud tato uživatelská procedura aktualizuje pole MQCD, ujistěte se, že nedochází k soupeření o další uživatelské procedury spuštěné v rámci jiných konverzací ve stejnou dobu tím, že poskytují serializaci mezi ukončovacími programy, které jsou spuštěny na této instanci kanálu.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_7.

MCAUserSource (MQLONG)
Toto pole uvádí zdroj poskytnutého ID uživatele MCA.

Může obsahovat jednu z následujících hodnot:

MQUSRC_MAP
ID uživatele je určeno v atributu MCAUSER.

MQUSRC_KANÁL
ID uživatele je přenášeno od příchozího partnera nebo je určeno v poli MCAUSER, které je definováno v objektu kanálu.

Toto je vstupní pole pro ukončení. Pole není přítomno, je-li verze menší než MQCXP_VERSION_8.

Body pEntry(PMQIEP)
Toto pole určuje adresu vstupního bodu rozhraní pro volání MQI nebo DCI.

Pole není přítomno, pokud *Verze* je menší než MQCXP_VERSION_8.

RemoteProduct (MQCHAR4)
Toto pole uvádí název vzdáleného produktu.

Toto pole identifikuje vzdálený produkt klienta, například C nebo Java, jak je zobrazeno v poli **RPRODUCT** příkazu DISPLAY CHSATUS.

Pole není přítomno, pokud *Verze* je menší než MQCXP_VERSION_9.

RemoteVersion (MQCHAR8)
Toto pole uvádí název vzdálené verze.

Toto pole identifikuje verzi knihoven klienta, jak je zobrazeno v poli **RVERSION** příkazu DISPLAY CHSTATUS.

Pole není přítomno, pokud *Verze* je menší než MQCXP_VERSION_9.

Deklarace C
Toto prohlášení je prohlášení 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 COBOL

Toto deklarace je deklarací 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)

Toto prohlášení 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
```

Deklarace asembleru System/390

Toto prohlášení je deklarací assembleru 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-Ukončení deskriptoru čekání

Struktura MQXWD je vstupní/výstupní parametr pro volání MQXWAIT.

Tato struktura je podporována pouze v systému z/OS.

Související odkazy

“Pole” na stránce 1519
Toto téma uvádí všechna pole v rámci struktury MQXWD a popisuje každé pole.

“Deklarace C” na stránce 1519
Toto deklarace je deklarací C pro strukturu MQXWD.

“Deklarace asembleru System/390” na stránce 1520
Toto prohlášení je deklarací assembler System/390 pro strukturu MQXWD.

Pole

Toto téma uvádí všechna pole v rámci struktury MQXWD a popisuje každé pole.

StrucId (MQCHAR4)
Toto pole uvádí identifikátor struktury.

Hodnota musí být:

ID_STRUKTURY MQXWD_STRUCTION_ID
Identifikátor pro strukturu deskriptoru čekání na ukončení.

Pro programovací jazyk C je také definována konstanta MQXWD_STRUC_ID_ARRAY; tato konstanta má stejnou hodnotu jako MQXWD_STRUC_ID, ale je to pole znaků namísto řetězce.

Počáteční hodnota tohoto pole je MQXWD_STRUC_ID.

Verze (MQLONG)
Toto pole uvádí 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 vyhrazené. Jeho hodnota musí být nula.

Toto je vstupní pole.

Reserved2 (MQLONG)
Toto pole je vyhrazené. Jeho hodnota musí být nula.

Toto je vstupní pole.

Reserved3 (MQLONG)
Toto pole je vyhrazené. Jeho hodnota musí být nula.

Toto je vstupní pole.

ECB (MQLONG)
Toto pole uvádí řídicí blok události, na kterém se má čekat.

Toto pole je řídicí blok událostí (ECB), na které se má čekat. Musí být nastaven na nulu před zadáním volání MQXWAIT; při úspěšném dokončení obsahuje poštovní směrovací číslo.

Toto pole je vstupní/výstupní pole.

Deklarace C

Toto deklarace je deklarací 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 */
};
```

Deklarace asembleru System/390

Toto prohlášení je deklarací assembler 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 uživatelské procedury pracovní zátěže klastru a datové struktury

Tento oddíl obsahuje referenční informace pro uživatelskou proceduru pracovní zátěže klastru a datové struktury. Jedná se o informace o rozhraní pro programování s generickými informacemi.

Uživatelské procedury pracovní zátěže klastru lze zapisovat v následujících programovacích jazycích:

- C
- Asembler System/390 (IBM MQ for z/OS)

Volání je popsáno v:

- “MQ_CLUSTER_WORKLOAD_EXIT -Popis volání” na stránce 1521

Datové typy struktury použité uživatelskou procedurou jsou popsány v následujících tématech:

- “MQXCLWLN -Procházet záznamy pracovní zátěže klastru” na stránce 1522
- “MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru” na stránce 1526
- “MQWDR-Struktura záznamu cíle pracovní zátěže klastru” na stránce 1534
- “MQWQR -Struktura záznamu fronty pracovní zátěže klastru” na stránce 1538
- “MQWCR -Struktura záznamu klastru pracovní zátěže klastru” na stránce 1544
- z/OS Asynchronní chování příkazů CLUSTER v systému z/OS

V celé této sekci jsou atributy správce front a atributy fronty zobrazeny v plném rozsahu. Ekvivalentní názvy, které se používají v příkazech MQSC, jsou zobrazeny níže. Podrobnosti o příkazech MQSC najdete v tématu Příkazy MQSC.

Tabulka 825. Atributy správce front

Celé jméno	**Název použitý v prostředí MQSC**
ClusterWorkloadData	CLWLDATA
ClusterWorkloadExit	CLWLEXIT
ClusterWorkloadLength	CLWLLEN

Tabulka 826. Atributy fronty	
Celé jméno	**Název použitý v prostředí 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 pro směrování zprávy do dostupného správce front.

Poznámka: Správcem front není poskytnut žádný vstupní bod s názvem MQ_CLUSTER_WORKLOAD_EXIT . Namísto toho je název uživatelské procedury pracovní zátěže klastru definován atributem správce front ClusterWorkloadExit .

Ukončení MQ_CLUSTER_WORKLOAD_EXIT je podporováno 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, MQWQRa MQWCR uložených v mezipaměti klastru.

MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru MQWXP -klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole v rámci struktury záznamu cíle pracovní zátěže klastru MQWDR -klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru MQWCR .

Parametry pro MQ_CLUSTER_WORKLOAD_EXIT

Popis parametrů ve volání MQ_CLUSTER_WORKLOAD_EXIT .

***ExitParms* (MQWXP) -vstup/výstup**
Ukončí blok parametrů.

- Uživatelská procedura nastaví informace v MQWXP , aby bylo možné určit, jak se má správa pracovní zátěže spravovat.

Související odkazy

Poznámky k použití

Funkce, kterou provádí uživatelskou proceduru pracovní zátěže klastru, je definována poskytovatelem uživatelské procedury. Ukončení se však musí řídit pravidly definovanými v přidruženém řídicím bloku MQWXP.

Vyvolání jazyka pro produkt MQ_CLUSTER_WORKLOAD_EXIT
Produkt MQ_CLUSTER_WORKLOAD_EXIT podporuje dva jazyky, C a High Level Assembler.

Poznámky k použití

Funkce, kterou provádí uživatelskou proceduru pracovní zátěže klastru, je definována poskytovatelem uživatelské procedury. Ukončení se však musí řídit pravidly definovanými v přidruženém řídicím bloku MQWXP.

Správcem front není poskytnut žádný vstupní bod s názvem MQ_CLUSTER_WORKLOAD_EXIT . Pro název MQ_CLUSTER_WORKLOAD_EXIT v programovacím jazyku C je však poskytnuta hodnota typedef . Použijte typedef k deklarování uživatelské procedury, abyste se ujistili, že jsou parametry správné.

Související odkazy

Parametry pro MQ_CLUSTER_WORKLOAD_EXIT
Popis parametrů ve volání MQ_CLUSTER_WORKLOAD_EXIT .

Vyvolání jazyka pro produkt MQ_CLUSTER_WORKLOAD_EXIT
Produkt MQ_CLUSTER_WORKLOAD_EXIT podporuje dva jazyky, C a High Level Assembler.

Vyvolání jazyka pro produkt 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);
```

Nahraďte *MQ_CLUSTER_WORKLOAD_EXIT* názvem funkce uživatelské procedury pracovní zátěže klastru.

Deklarujte parametry **MQ_CLUSTER_WORKLOAD_EXIT** následujícím způsobem:

```
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, kterou provádí uživatelskou proceduru pracovní zátěže klastru, je definována poskytovatelem uživatelské procedury. Ukončení se však musí řídit 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 hlavní paměti používaná k ukládání informací souvisejících s klastrem.

Je-li mezipaměť klastru statická, má pevnou velikost. Pokud ji nastavíte na dynamický, mezipaměť klastru se může podle potřeby rozšířit.

Nastavte typ mezipaměti klastru na hodnotu `STATIC` nebo `DYNAMIC` buď pomocí parametru systému, nebo makra.

- Multi Použijte parametr systému `ClusterCacheType` na 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 pro směrování zprávy do dostupného správce front.

MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru `MQWXP` -klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole v rámci struktury záznamu cíle pracovní zátěže klastru `MQWDR` -klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole v `MQWQR` -Struktura záznamu fronty pracovní zátěže klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru `MQWCR` .

Parametry pro MQXCLWLN -Procházet záznamy pracovní zátěže klastru

Popis parametrů ve volání `MQXCLWLN` .

ExitParms (`MQWXP`) -vstup/výstup
Ukončí blok parametrů.

Tato struktura obsahuje informace vztahující se k vyvolání uživatelské procedury. Uživatelská procedura nastaví informace v této struktuře, aby indikovala, jak spravovat pracovní zátěž.

CurrentRecord (`MQPTR`) -vstup
Adresa aktuálního záznamu.

Tato struktura obsahuje informace vztahující se k adrese záznamu, který je momentálně vyšetřován z uživatelské procedury. Záznam musí být jeden z následujících typů:

- Cílový záznam pracovní zátěže klastru (`MQWDR`)
- Záznam fronty pracovní zátěže klastru (`MQWQR`)
- Záznam klastru pracovní zátěže klastru (`MQWCR`)

NextOffset (`MQLONG`) -vstup
Offset dalšího záznamu.

Tato struktura obsahuje informace vztahující se k posunu dalšího záznamu nebo struktury. *`NextOffset`* je hodnota odpovídajícího pole offsetu v aktuálním záznamu a musí být jedno z následujících polí:

- pole `OffsetChannelDefOffset` v `MQWDR`
- Pole `ClusterRecOffset` v `MQWDR`
- Pole `ClusterRecOffset` v `MQWQR`

- Pole ClusterRecOffset v MQWCR

NextRecord (MQPTR) -výstup
Adresa dalšího záznamu nebo struktury.

Tato struktura obsahuje informace vztahující se k adrese dalšího záznamu nebo struktury. Je-li *CurrentRecord* adresa MQWDRa *NextOffset* je hodnota pole ChannelDefOffset , *NextRecord* je adresa struktury definice kanálu (MQCD).

Pokud žádný další záznam nebo struktura neexistuje, správce front nastaví ukazatel *NextRecord* na ukazatel 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 se nezdařilo.

Příčina (MQLONG) -výstup
Kód příčiny, který kvalifikují CompCode

Má-li parametr CompCode hodnotu MQCC_OK:

MQRC_NONE
(0, X'0000')
Žádný důvod k hlášení.

Pokud je *CompCode* MQCC_WARNING:

MQRC_NO_RECORD_AVAILABLE
(2359, X'0937')
K dispozici není žádný záznam. Volání MQXCLWLN bylo vydáno z uživatelské procedury pracovní zátěže klastru k získání adresy dalšího záznamu v řetězci. Aktuální záznam je poslední záznam v řetězci. Nápravná akce: Žádná.

Pokud je *CompCode* MQCC_FAILED:

MQRC_CURRENT_RECORD_ERROR
(2357, X'0935')
Argument **CurrentRecord** není platný. Volání MQXCLWLN bylo vydáno z uživatelské procedury pracovní zátěže klastru k získání adresy dalšího záznamu v řetězci. Adresa zadaná parametrem **CurrentRecord** není adresou platného záznamu.

CurrentRecord musí být adresa záznamu cíle, MQWDR, záznamu fronty (MQWQR) nebo záznamu klastru (MQWCR) umístěné v mezipaměti klastru. Nápravná akce: Ujistěte se, že uživatelská procedura pracovní zátěže klastru předá adresu platného záznamu nacházejícího se v mezipaměti klastru.

MQRC_ENVIRONMENT_ERROR
(2012, X'07DC')
Volání není platné v prostředí. Bylo vydáno volání MQXCLWLN , ale ne z uživatelské procedury pracovní zátěže klastru.

MQRC_NEXT_OFFSET_ERROR
(2358, X'0936')
Argument **NextOffset** není platný. Volání MQXCLWLN bylo vydáno z uživatelské procedury pracovní zátěže klastru k získání adresy dalšího záznamu v řetězci. Offset zadaný argumentem **NextOffset** není platný. **NextOffset** musí být hodnota jednoho z následujících polí:

- pole `OffsetChannelDefOffset` v `MQWDR`
- Pole `ClusterRecOffset` v `MQWDR`
- Pole `ClusterRecOffset` v `MQWQR`
- Pole `ClusterRecOffset` v `MQWCR`

Nápravná akce: Ujistěte se, že hodnota uvedená pro parametr **NextOffset** je hodnota jednoho z polí vypsaných dříve.

MQRC_NEXT_RECORD_ERROR
(2361, X'0939')
Argument **NextRecord** není platný.

MQRC_WXP_ERROR
(2356, X'0934')
Struktura výstupních parametrů pracovní zátěže není platná. Volání `MQXCLWLN` bylo vydáno z uživatelské procedury pracovní zátěže klastru k získání adresy dalšího záznamu v řetězci. Struktura parametru uživatelské procedury pracovní zátěže **ExitParms** není platná, a to z jednoho z následujících důvodů:

- Ukazatel parametru je neplatný. 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, která byla předána uživatelské proceduře správcem front.

Nápravná akce: Ujistěte se, že parametr zadaný pro **ExitParms** je struktura `MQWXP` , která byla předána uživatelské proceduře, když byla vyvolána uživatelská procedura.

Související odkazy
Poznámky k použití pro MQXCLWLN-Procházet záznamy pracovní zátěže klastru
Použijte `MQXCLWLN` k navigaci mezi záznamy klastru, a to i v případě, že je mezipaměť statická.

Vyvolání jazyků produktu MQXCLWLN
Produkt `MQXCLWLN` podporuje dva jazyky, C a High Level Assembler.

Poznámky k použití pro MQXCLWLN-Procházet záznamy pracovní zátěže klastru
Použijte `MQXCLWLN` k navigaci mezi záznamy klastru, a to i v případě, že je mezipaměť statická.

Je-li mezipaměť klastru dynamická, musí být volání `MQXCLWLN` použito k navigaci v záznamech. Uživatelská procedura skončí abnormálně, když se pro navigaci v záznamech používá jednoduchý aritmetický ukazatel a-ofsetový aritmetický ukazatel.

Je-li mezipaměť klastru statická, produkt `MQXCLWLN` nemusí být použit k navigaci v záznamech. Obvykle používejte `MQXCLWLN` i tehdy, je-li mezipaměť statická. Pak můžete změnit mezipaměť klastru tak, aby byla dynamická, aniž by bylo nutné změnit uživatelskou proceduru pracovní zátěže.

Související odkazy
Parametry pro MQXCLWLN -Procházet záznamy pracovní zátěže klastru
Popis parametrů ve volání `MQXCLWLN` .

Vyvolání jazyků produktu MQXCLWLN
Produkt `MQXCLWLN` podporuje dva jazyky, C a High Level Assembler.

Vyvolání jazyků produktu 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
Použijte MQXCLWLN k navigaci mezi záznamy klastru, a to i v případě, že je mezipaměť statická.

MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru

Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru MQWXP -klastru.

Tabulka 827. Pole v MQWXP

Pole	Popis	Stránka
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
ExitId	Typ výstupu	ExitId
ExitReason	Důvod vyvolání uživatelské procedury	ExitReason
ExitResponse	Odezva z uživatelské procedury	ExitResponse
ExitResponse2	Sekundární odezva od ukončení	ExitResponse2
Feedback	Kód zpětné vazby	Zpětná vazba
Flags	Hodnoty příznaků. Tyto bitové příznaky se používají k označení informací o vkládané zprávě.	Příznaky
ExitUserArea	Uživatelská oblast pro ukončení	OblastExitUser
ExitData	Data uživatelské procedury	ExitData
MsgDescPtr	Adresa deskriptoru zpráv (MQMD)	MsgDescPtr
MsgBufferPtr	Adresa vyrovnávací paměti obsahující některá nebo všechna data zprávy	MsgBufferPtr

Tabulka 827. Pole v MQWXP (pokračování)		
Pole	**Popis**	**Stránka**
MsgBufferLength	Délka vyrovnávací paměti obsahující data zprávy	DélkaMsgBufferDélka
MsgLength	Délka úplné zprávy	MsgLength
QName	Název fronty	QName
QMgrName	Název lokálního správce front	QMgrName
DestinationCount	Počet možných míst určení	DestinationCount
DestinationChosen	Vybraný cíl	DestinationChosen
DestinationArrayPtr	Adresa pole ukazatelů na cílové záznamy (MQWDR)	DestinationArrayPtr
QArrayPtr	Adresa pole ukazatelů na záznamy fronty (MQWQR)	QArrayPtr
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQWXP_VERSION_2.		
CacheContext	Informace o kontextu	CacheContext
CacheType	Typ mezipaměti klastru	CacheType
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQWXP_VERSION_3.		
CLWLMRUChannels	Maximální počet povolených aktivních odchozích kanálů klastru	CLWLMRUChannels
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQWXP_VERSION_4.		
pEntryPoints	Adresa struktury MQIEP pro povolení volání MQI a DCI, jež mají být provedeny	pEntrybodů

Struktura parametrů uživatelské procedury pracovní zátěže klastru popisuje informace, které jsou předány uživatelské proceduře pracovní zátěže klastru.

Struktura parametru uživatelské procedury pracovní zátěže klastru je podporována na všech platformách

Kromě toho jsou struktury MQWXP1, MQWXP2 a MQWXP3 k dispozici pro zpětnou kompatibilitu.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front pro 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 v rámci struktury záznamu cíle pracovní zátěže klastru MQWDR -klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru

Popis polí v MQWXP -Struktura parametru 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 také definována 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 parametru uživatelské procedury pracovní zátěže klastru Version-1 .

Produkt MQWXP_VERSION_1 je podporován ve všech prostředích.

MQWXP_VERSION_2
Struktura parametru 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 parametru 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 parametru 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 parametru ukončení pracovní zátěže klastru.

ExitId (MQLONG)-vstup.
Označuje typ uživatelské procedury, která se volá. 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 je ukončení vyvoláno poprvé.

Získáte a inicializujte všechny prostředky, které může uživatelská procedura potřebovat, jako např. hlavní paměť.

MQXR_TERM
Označuje, že ukončení se chystá ukončit.

Uvolněte všechny prostředky, které mohla uživatelská procedura získat od jeho inicializace, jako je hlavní paměť.

MQXR_CLWL_OPEN
Volán produktem MQOPEN.

MQXR_CLWL_PUT
Voláno příkazem MQPUT nebo MQPUT1.

MQXR_CLWL_MOVE
Voláno MCA, když se stav kanálu změnil.

MQXR_CLWL_REPOS
Volán příkazem MQPUT nebo MQPUT1 pro zprávu PCF správce úložiště.

MQXR_CLWL_REPOS_MOVE
Volán agentem MCA pro zprávu PCF správce úložiště, pokud se stav kanálu změnil.

ExitResponse (MQLONG)-výstup
Nastavte ExitResponse , abyste označili, zda zpracování zprávy pokračuje. Musí se jednat o jednu z následujících hodnot:

MQXCC_OK
Pokračujte ve zpracování zprávy obvyklým způsobem.

- DestinationChosen identifikuje místo určení, do kterého má být zpráva odeslána.

MQXCC_SUPPRESS_FUNCTION
Přerušte zpracování zprávy.

- Akce prováděné správcem front závisí na příčině, proč byla vyvolána uživatelská procedura:

Tabulka 828. Akce prováděné správcem front

ExitReason	**Provedná akce**
– MQXR_CLWL_OPEN – MQXR_CLWL_REPOS – MQXR_CLWL_PUT	Volání MQOPEN, MQPUT, nebo MQPUT1 se nezdařilo s kódem dokončení MQCC_FAILED a kódem příčiny MQRC_STOPPED_BY_CLUSTER_EXIT.
– MQXR_CLWL_MOVE – MQXR_CLWL_REPOS_MOVE	Zpráva se umístí do fronty nedoručených zpráv.

MQXCC_SUPPRESS_EXIT
Běžně pokračuje ve zpracování aktuální zprávy. Nevyvolávejte uživatelskou proceduru znovu, dokud se správce front neuzavře.

Správce front zpracovává následující zprávy, jako kdyby byl atribut správce front ClusterWorkloadExit prázdný. DestinationChosen identifikuje místo určení, do kterého se má odeslat aktuální zpráva.

Jakákoli jiná hodnota
Zpracujte zprávu, jako by byla zadána hodnota MQXCC_SUPPRESS_FUNCTION .

ExitResponse2 (MQLONG)-vstup/výstup
Chcete-li správci front poskytnout více informací, nastavte hodnotu ExitResponse2 .

- MQXR2_STATIC_CACHE je výchozí hodnota a je nastavena na vstup do uživatelské procedury.

- Má-li parametr ExitReason hodnotu MQXR_INIT, může uživatelská procedura nastavit jednu z následujících hodnot v ExitResponse2:

  MQXR2_STATIC_CACHE
  Ukončení vyžaduje statickou mezipaměť klastru.

  - Je-li mezipaměť klastru statická, nemusí uživatelská procedura používat volání produktu MQXCLWLN k navigaci v řetězech záznamů v mezipaměti klastru.
  - Je-li mezipaměť klastru dynamická, nemůže uživatelská procedura v mezipaměti správně procházet záznamy v mezipaměti.

    Poznámka: Správce front zpracovává návrat z volání funkce MQXR_INIT , jako by byla procedura vrácena MQXCC_SUPPRESS_EXIT v poli ExitResponse .

  MQXR2_DYNAMIC_CACHE
  Ukončení může fungovat buď se statickou nebo dynamickou mezipamětí.

  - Pokud uživatelská procedura vrátí tuto hodnotu, musí uživatelská procedura používat volání produktu MQXCLWLN k navigaci v řetězcích záznamů v mezipaměti klastru.

Zpětná vazba (MQLONG)-Vstup.
Vyhrazené pole. Hodnota je nula.

Parametry (MQLONG)-Vstup
Označuje informace o vkládané zprávě.

- Hodnota parametru Flags je MQWXP_PUT_BY_CLUSTER_CHL. Zpráva pochází z kanálu klastru, nikoli lokálně nebo z kanálu, který není kanálem klastru. Jinými slovy, zpráva pochází od jiného správce front klastru.

Vyhrazeno (MQLONG)-Vstup.
Vyhrazené pole. Hodnota je nula.

ExitUserArea (MQBYTE16)-vstup/výstup
Nastavte volbu ExitUserArea , chcete-li komunikovat mezi voláními do ukončení.

- OblastExitUser se inicializuje na binární nulu před prvním vyvoláním uživatelské procedury. Veškeré změny provedené v tomto poli provedené uživatelskou procedurou budou zachovány v rámci vyvolání procedury, která se vyskytne mezi voláním MQCONN a odpovídajícím voláním MQDISC . Když se vyskytne volání MQDISC , pole se vynuluje na binární nulu.
- První vyvolání uživatelské procedury je označeno polem ExitReason , které má hodnotu MQXR_INIT.
- Jsou definovány následující 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 ExitUserArea.

ExitData (MQCHAR32)-vstup
Hodnota atributu správce front ClusterWorkloadData . Pokud pro příslušný atribut nebyla definována žádná hodnota, bude toto pole obsahovat pouze prázdné znaky.

- Délka hodnoty ExitData je dána produktem MQ_EXIT_DATA_LENGTH.

MsgDescPtr (PMQMD)-vstup
Adresa kopie deskriptoru zpráv (MQMD) pro zpracovávanou zprávu.

- Veškeré změny deskriptoru zpráv provedené uživatelskou procedurou budou správcem front ignorovány.
- Má-li parametr ExitReason jednu z následujících hodnot, MsgDescPtr je nastaven na ukazatel Null a do uživatelské procedury se nepředává žádný deskriptor zprávy:
  - MQXR_INIT

- MQXR_TERM
- MQXR_CLWL_OPEN

MsgBufferPtr (PMQVOID)-vstup
Adresa vyrovnávací paměti obsahující kopii prvních MsgBufferLength bajtů dat zprávy.

- Veškeré změny dat zprávy provedené uživatelskou procedurou budou správcem front ignorovány.
- K ukončení nejsou předána žádná data zprávy, když:
  - MsgDescPtr je ukazatel Null.
  - Zpráva nemá žádná data.
  - Atribut správce front ClusterWorkloadLength má hodnotu nula.

  V těchto případech je ukazatel Null MsgBufferPtr .

MsgBufferLength (MQLONG)-vstup.
Délka vyrovnávací paměti obsahující data zprávy předaná k ukončení.

- Délka je řízena atributem správce front ClusterWorkloadLength .
- Délka může být menší než délka úplné zprávy, viz MsgLength.

MsgLength (MQLONG)-vstup.
Délka úplné zprávy předané do uživatelské procedury.

- Hodnota MsgBufferLength může být menší než délka úplné zprávy.
- MsgLength je nula, pokud ExitReason je MQXR_INIT, MQXR_TERM, nebo MQXR_CLWL_OPEN.

QName (MQCHAR48)-Vstup
Název cílové fronty. Fronta je fronta klastru.

- Délka parametru QName je MQ_Q_NAME_LENGTH.

QMgrName (MQCHAR48)-vstup
Název lokálního správce front, který vyvolal uživatelskou proceduru pracovní zátěže klastru.

- Délka parametru QMgrName je MQ_Q_MGR_NAME_LENGTH.

DestinationCount (MQLONG)-vstup
Počet možných cílů. Místa určení jsou instance cílové fronty a jsou popsána v cílových záznamech.

- 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
Zvolené místo určení.

- Číslo struktury MQWDR , která označuje přenosovou cestu 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 správce front vybral. Uživatelská procedura může přijmout tuto volbu nebo zvolit jinou trasu a instanci fronty.
- Hodnota nastavená ukončovacím programem musí být v rozsahu 1- DestinationCount. Je-li 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, existují `DestinationCount` z nich.
- Nejsou-li k dispozici žádné záznamy fronty, `QArrayPtr` je ukazatel Null.

  Poznámka: `QArrayPtr` může být prázdný ukazatel i v případě, že `DestinationCount` je 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 nemůže růst, protože správce front pracuje.
- Chcete-li procházet záznamy v tomto typu mezipaměti, nemusíte používat volání produktu `MQXCLWLN` .

`MQCLCT_DYNAMIC`
Mezipaměť je dynamická.

- Velikost mezipaměti se může zvýšit, aby bylo možné přizpůsobit proměnlivé informace 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 volby pracovní zátěže klastru.

- `CLWLMRUChannels` je hodnota 1-999 999 999.

`pEntrybodů` (`PMQIEP`): Verze 4
Adresa struktury MQIEP, pomocí které lze provádět volání MQI a DCI.

Související odkazy
Počáteční hodnoty a deklarace jazyka pro MQWXP
Počáteční hodnoty a C a High Level Assembler Jazykové deklarace pro `MQWXP` -Struktura parametru uživatelské procedury pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWXP

Počáteční hodnoty a C a High Level Assembler Jazykové deklarace pro `MQWXP` -Struktura parametru uživatelské procedury pracovní zátěže klastru.

Tabulka 829. Počáteční hodnoty polí v 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

Tabulka 829. Počáteční hodnoty polí v MQWXP (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
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 jazyce C obsahuje proměnná makra MQWXP_DEFAULT výchozí hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQWDR MyWXP = {MQWXP_DEFAULT};
```

Deklarace C

```
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 parametru uživatelské procedury pracovní zátěže klastru
Popis polí v MQWXP -Struktura parametru uživatelské procedury pracovní zátěže klastru

MQWDR-Struktura záznamu cíle pracovní zátěže klastru

Následující tabulka shrnuje pole v rámci struktury záznamu cíle pracovní zátěže klastru MQWDR -klastru.

Tabulka 830. Pole v 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

Tabulka 830. Pole v MQWDR (pokračování)		
Pole	**Popis**	**Stránka**
ClusterRecOffset	Logický posun prvního záznamu klastru (MQWCR)	PosunutíClusterRec
ChannelState	Stav kanálu	ChannelState
ChannelDefOffset	Logický posun struktury definice kanálu (MQCD)	ChannelDef
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQWDR_VERSION_2.		
DestSeqNumber	Pořadové číslo místa určení kanálu	DestSeqČíslo
DestSeqFactor	Faktor posloupnosti cíle kanálu pro váhu	DestSeqFaktor

Struktura záznamu cíle pracovní zátěže klastru obsahuje informace týkající se jednoho z možných míst určení pro zprávu. Pro každou instanci cílové fronty existuje jedna struktura záznamů místa určení pracovní zátěže klastru.

Struktura záznamu cíle pracovní zátěže klastru je podporována ve všech prostředích.

Kromě toho jsou struktury MQWDR1 a MQWDR2 k dispozici pro zpětnou kompatibilitu.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front pro 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 parametru uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru MQWXP -klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWDR-Struktura záznamu cíle pracovní zátěže klastru

Popis parametrů ve struktuře MQWDR -Struktura záznamu cíle pracovní zátěže klastru.

StrucId (MQCHAR4) -vstup
Identifikátor struktury pro strukturu záznamu cíle pracovní zátěže klastru.

- Hodnota StrucId je MQWDR_STRUC_ID.
- Pro programovací jazyk C je také definována konstanta MQWDR_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQWDR_STRUC_ID. Jedná se o pole znaků namísto řetězce.

Verze (MQLONG) -vstup
Číslo verze struktury. Verze má jednu z následujících hodnot:

MQWDR_VERSION_1
Cílový záznam pracovní zátěže klastru Version-1 .

MQWDR_VERSION_2
Cílový záznam pracovní zátěže klastru Version-2 .

MQWDR_CURRENT_VERSION
Aktuální verze záznamu cíle pracovní zátěže klastru.

StrucLength (MQLONG) -vstup
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) -vstup
Příznaky správce front označují vlastnosti správce front, který je hostitelem instance cílové fronty popsané strukturou MQWDR . Jsou definovány následující příznaky:

MQQMF_REPOSITORY_Q_MGR
Cíl je správce front úplného úložiště.

MQQMF_CLUSSDR_USER_DEFINED
Kanál odesílatele klastru byl definován ručně.

MQQMF_CLUSSDR_AUTO_DEFINED
Kanál odesílatele klastru byl definován automaticky.

MQQMF_AVAILABLE
Cílový správce front je k dispozici pro příjem zpráv.

Ostatní hodnoty
Jiný příznak v poli může být nastaven správcem front pro interní účely.

QMgrIdentifier (MQCHAR48) -vstup
Identifikátor správce front je jedinečným identifikátorem pro správce front, který je hostitelem instance cílové fronty popsané strukturou MQWDR .

- Identifikátor je generován správcem front.
- Délka parametru QMgrIdentifier je MQ_Q_MGR_IDENTIFIER_LENGTH.

QMgrName (MQCHAR48) -vstup
Název správce front, který je hostitelem instance cílové fronty popsané strukturou MQWDR .

- QMgrName může být název lokálního správce front, stejně jako jiného správce front v klastru.
- Délka parametru QMgrName je MQ_Q_MGR_NAME_LENGTH.

OffsetClusterRec (MQLONG) -vstup
Logický posun první struktury MQWCR , která patří ke struktuře MQWDR .

- Pro statické mezipaměti je hodnota ClusterRecOffset posunutí první struktury MQWCR , která patří do struktury MQWDR .
- Posunutí se měří v bajtech od začátku struktury MQWDR .
- Nepoužívejte logický posun pro aritmetiku ukazatele s dynamickými mezipaměťmi. Chcete-li získat adresu dalšího záznamu, musí být použito volání MQXCLWLN .

ChannelState (MQLONG) -vstup
Stav kanálu, který spojuje lokálního správce front se správcem front identifikovaným strukturou MQWDR . Možné jsou následující hodnoty:

MQCHS_BINDING
Kanál jedná s partnerem.

MQCHS_INACTIVE
Kanál není aktivní.

MQCHS_INITIALIZING
Probíhá inicializace kanálu.

MQCHS_PAUSED
Kanál byl pozastaven.

MQCHS_REQUESTING
Kanál žadatele vyžaduje připojení.

MQCHS_RETRYING
Kanál se znovu pokusí o vytvoření připojení.

MQCHS_RUNNING
Kanál se přenáší nebo čeká na zprávy.

MQCHS_STARTING
Kanál čeká na aktivaci.

MQCHS_STOPPING
Kanál se zastavuje.

MQCHS_STOPPED
Kanál byl zastaven.

ChannelDefOffset (MQLONG) -vstup
Logický posun definice kanálu (MQCD) pro kanál, který spojuje lokálního správce front se správcem front identifikovaným strukturou MQWDR .

- PosunutíChannelDef je jako ClusterRecOffset .
- Logický posun nemůže být použit v aritmetické aritmetice. Chcete-li získat adresu dalšího záznamu, musí být použit volání MQXCLWLN .

DestSeqFaktor (MQLONG) -vstup
Cílový sekvenční faktor, který umožňuje volbu kanálu na základě váhy.

- DestSeqFaktor se používá před tím, než se správce front změní.
- Správce pracovní zátěže zvětší DestSeqFactor způsobem, který zajišťuje distribuci zpráv směrem dolů v závislosti na jejich váze.

DestSeqČíslo (MQLONG) -vstup
Cílová hodnota kanálu klastru před tím, než je správce front změněn.

- Správce pracovní zátěže zvětší DestSeqNumber pokaždé, když je zpráva vložena do tohoto kanálu.
- Uživatelské procedury pracovní zátěže mohou použít DestSeqNumber k rozhodnutí, který kanál má být umístěn do nižší úrovně.

Související odkazy
Počáteční hodnoty a deklarace jazyka pro MQWDR
Počáteční hodnoty a C a High Level Assembler pro MQWDR -Záznam cíle pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWDR

Počáteční hodnoty a C a High Level Assembler pro MQWDR -Záznam cíle pracovní zátěže klastru.

Tabulka 831. Počáteční hodnoty polí v 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

Tabulka 831. Počáteční hodnoty polí v MQWDR (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
DestSeqFactor	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C obsahuje proměnná makra MQWDR_DEFAULT výchozí hodnoty. Použijte ji 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)
```

Deklarace C

```
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 MQWDR -Struktura záznamu cíle pracovní zátěže klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru

Následující tabulka shrnuje pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

Tabulka 832. Pole v MQWQR		
Pole	**Popis**	**Stránka**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQWQR	StrucLength
QFlags	Příznaky fronty	Parametry QFlags
QName	Název fronty	QName
QMgrIdentifier	Identifikátor správce front	QMgrIdentifier
ClusterRecOffset	Posunutí prvního záznamu klastru (MQWCR)	PosunutíClusterRec
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 se budou ignorovat, pokud je verze nižší než MQWQR_VERSION_2.		
CWLQueuePriority	Hodnota 0-9 reprezentující prioritu fronty	CLWLQueuePriority
CLWLQueueRank	Hodnota 0-9 reprezentující pořadí zařazení do fronty	CLWLQueueRank
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQWQR_VERSION_3.		
DefPutResponse	Odezva výchozího umístění	DefPutOdezva

Struktura záznamu fronty pracovní zátěže klastru obsahuje informace vztahující se k jednomu z možných cílů pro danou 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.

Kromě toho jsou struktury MQWQR1 a MQWQR2 k dispozici pro zpětnou kompatibilitu.

Související odkazy

MQ_CLUSTER_WORKLOAD_EXIT -Popis volání
Uživatelská procedura pracovní zátěže klastru je volána správcem front pro 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 parametru uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru MQWXP -klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole v rámci struktury záznamu cíle pracovní zátěže klastru MQWDR -klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru
Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru MQWCR .

Pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru

Popis polí ve struktuře MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

StrucId (MQCHAR4) -vstup
Identifikátor struktury pro strukturu záznamu fronty pracovní zátěže klastru.

- Hodnota StrucId je MQWQR_STRUC_ID.
- Pro programovací jazyk C je také definována konstanta MQWQR_STRUC_ID_ARRAY . Má stejnou hodnotu jako MQWQR_STRUC_ID. Jedná se o pole znaků namísto řetězce.

Verze (MQLONG) -vstup
Číslo verze struktury. Verze má jednu z následujících hodnot:

MQWQR_VERSION_1
Záznam fronty pracovní zátěže klastru Version-1 .

MQWQR_VERSION_2
Záznam fronty pracovní zátěže klastru Version-2 .

MQWQR_VERSION_3
Záznam fronty pracovní zátěže klastru Version-3 .

MQWQR_CURRENT_VERSION
Aktuální verze záznamu fronty pracovní zátěže klastru.

StrucLength (MQLONG) -vstup
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) -vstup
Parametry 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 v puts.

MQQF_CLWL_USEQ_LOCAL
Povolit vložení pouze lokální fronty.

Ostatní hodnoty
Jiný příznak v poli může být nastaven správcem front pro interní účely.

QName (MQCHAR48) -vstup
Název fronty, která je jedním z možných cílů zprávy.

- Délka parametru QName je MQ_Q_NAME_LENGTH.

QMgrIdentifier (MQCHAR48) -vstup
Identifikátor správce front je jedinečný identifikátor pro správce front, který je hostitelem instance fronty popsané strukturou MQWQR .

- Identifikátor je generován správcem front.
- Délka parametru QMgrIdentifier je MQ_Q_MGR_IDENTIFIER_LENGTH.

OffsetClusterRec (MQLONG) -vstup
Logický posun první struktury MQWCR , která patří do struktury MQWQR .

- Pro statické mezipaměti je hodnota `OffsetClusterRecOffset` posun první struktury `MQWCR` , která patří do struktury `MQWQR` .
- Posunutí se měří v bajtech od začátku struktury `MQWQR` .
- Nepoužívejte logický posun pro aritmetiku ukazatele s dynamickými mezipaměťmi. Chcete-li získat adresu dalšího záznamu, musí být použito volání `MQXCLWLN` .

QType (MQLONG) -vstup
Typ fronty místa určení. 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) -vstup
Atribut fronty popisu fronty definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou `MQWQR` .

- Délka `QDesc` je `MQ_Q_DESC_LENGTH`.

DefBind (MQLONG) -vstup
Výchozí atribut fronty vazeb 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 opravena voláním funkce `MQOPEN` .

MQBND_BIND_NOT_FIXED
Vazba nebyla opravena.

MQBND_BIND_ON_GROUP
Umožňuje aplikaci požadovat, aby skupina zpráv byla alokována do stejné cílové instance.

DefPersistence (MQLONG). -vstup
Výchozí atribut fronty perzistence zpráv definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře `MQWQR` . Možné jsou následující hodnoty:

MQPER_PERSISTENT
Zpráva je trvalá.

MQPER_NOT_PERSISTENT
Zpráva není trvalá.

DefPriority (MQLONG) -vstup

Výchozí atribut fronty priority zpráv definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře `MQWQR` . Rozsah priority je 0- `MaxPriority`.

- 0 je nejnižší priorita.
- `MaxPriority` je atribut správce front správce front, který je hostitelem této instance cílové fronty.

InhibitPut (MQLONG) -vstup
Atribut fronty s blokovaným vsem definovaný ve správci front, který je hostitelem instance cílové fronty popsané strukturou `MQWQR` . Možné jsou následující hodnoty:

MQQA_PUT_INHIBITED
Operace vložení jsou blokovány.

MQQA_PUT_ALLOWED
Operace vložení jsou povoleny.

CLWLQueuePriority (MQLONG) -vstup
Atribut priority fronty pracovní zátěže klastru definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře MQWQR .

CLWLQueueRank (MQLONG) -vstup
Očíslování pořadí fronty pracovní zátěže klastru definované ve správci front, který je hostitelem instance cílové fronty popsané strukturou MQWQR .

DefPutResponse (MQLONG) -vstup
Výchozí atribut fronty odpovědí vložení definovaný ve správci front, který je hostitelem instance cílové fronty popsané ve struktuře 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 C a High Level Assembler Jazykové deklarace pro MQWQR -Záznam fronty pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWQR

Počáteční hodnoty a C a High Level Assembler Jazykové deklarace pro MQWQR -Záznam fronty pracovní zátěže klastru.

Tabulka 833. Počáteční hodnoty polí v 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
DefPersistence	Není	0
DefPriority	Není	0
InhibitPut	Není	0
CLWLQueuePriority	Není	0
CLWLQueueRank	Není	0
DefPutResponse	Není	1

Tabulka 833. Počáteční hodnoty polí v MQWQR (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C obsahuje proměnná makra MQWQR_DEFAULT výchozí hodnoty. Použijte ji 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.

Deklarace C

```
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 MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

MQWCR -Struktura záznamu klastru pracovní zátěže klastru

Následující tabulka uvádí souhrn polí ve struktuře záznamu pracovní zátěže klastru MQWCR .

Tabulka 834. Pole v MQWCR		
Pole	**Popis**	**Stránka**
ClusterName	Název klastru	ClusterName
ClusterRecOffset	Offset dalšího záznamu klastru (MQWCR)	PosunutíClusterRec
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 pro 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 parametru uživatelské procedury pracovní zátěže klastru
Následující tabulka shrnuje pole ve struktuře parametrů uživatelské procedury pracovní zátěže klastru MQWXP -klastru.

MQWDR-Struktura záznamu cíle pracovní zátěže klastru
Následující tabulka shrnuje pole v rámci struktury záznamu cíle pracovní zátěže klastru MQWDR -klastru.

MQWQR -Struktura záznamu fronty pracovní zátěže klastru
Následující tabulka shrnuje pole v MQWQR -Struktura záznamu fronty pracovní zátěže klastru.

Pole v MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Popis polí ve struktuře MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

ClusterName (MQCHAR48) -vstup
Název klastru, do kterého patří instance cílové fronty, která vlastní strukturu MQWCR . Instance cílové fronty je popsána strukturou MQWDR .

- Délka parametru ClusterName je MQ_CLUSTER_NAME_LENGTH.

OffsetClusterRec (MQLONG) -vstup
Logický posun následující struktury MQWCR .

- Pokud již neexistuje více struktur MQWCR , bude hodnota ClusterRecOffset nulová.
- Posunutí se měří v bajtech od začátku struktury MQWCR .

ClusterFlags (MQLONG) -vstup
Parametry klastru označují vlastnosti správce front určeného strukturou MQWCR . Jsou definovány následující příznaky:

MQQMF_REPOSITORY_Q_MGR
Cíl je správce front úplného úložiště.

MQQMF_CLUSSDR_USER_DEFINED
Kanál odesílatele klastru byl definován ručně.

MQQMF_CLUSSDR_AUTO_DEFINED
Kanál odesílatele klastru byl definován automaticky.

MQQMF_AVAILABLE
Cílový správce front je k dispozici pro příjem zpráv.

Ostatní hodnoty
Jiný příznak v poli může být nastaven správcem front pro interní účely.

Související odkazy

Počáteční hodnoty a deklarace jazyka pro MQWCR
Počáteční hodnoty a C a High Level Assembler Jazykové deklarace pro MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Počáteční hodnoty a deklarace jazyka pro MQWCR

Počáteční hodnoty a C a High Level Assembler Jazykové deklarace pro MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Tabulka 835. Počáteční hodnoty polí v MQWCR

Název pole	Název konstanty	Hodnota konstanty
ClusterName	Není	""
ClusterRecOffset	Není	0
ClusterFlags	Není	0

Deklarace C

```
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 ve struktuře MQWCR -Struktura záznamu klastru pracovní zátěže klastru.

Popis uživatelské procedury rozhraní

Tato část obsahuje informace o odkazech, které se týkají hlavně zájmu programátora odepisující uživatelské procedury rozhraní API.

Obecné poznámky k použití

poznámky:

1. Všechny výstupní funkce mohou vydávat volání MQXEP; toto volání je určeno speciálně pro použití z uživatelských funkcí rozhraní API.
2. Funkce MQ_INIT_EXIT nemůže vydat žádné volání MQ jiné než MQXEP.
3. Nemůžete vydat volání MQDISC pro aktuální připojení.
4. Pokud funkce uživatelské procedury vydá volání MQCONN nebo volání MQCONNX s volbou MQCNO_HANDLE_SHARE_NONE, volání bude dokončeno s kódem příčiny

MQRC_ALREADY_CONNECTED a vrácený popisovač je stejný jako ten, který byl předán uživatelské proceduře jako parametr.

5. Obecně platí, že když funkce uživatelské procedury rozhraní API odešle volání MQI, uživatelské procedury rozhraní API nejsou volány rekurzivně. Pokud však funkce ukončení vyvolá volání MQCONNX s volbami MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, pak volání vrátí nový sdílený popisovač. Poskytuje výstupní sadu s vlastním připojením, a tudíž jednotkou práce, která je nezávislá na pracovní jednotce aplikace. Výstupní sada může tento popisovač použít k vložení a získání zpráv v rámci své vlastní pracovní jednotky a k potvrzení nebo vrácení této jednotky práce; to vše lze provést, aniž by to mělo vliv na jednotku práce v aplikaci.

   Protože funkce uživatelské procedury používá manipulátor připojení, který se liší od popisovače používaného aplikací, volání MQ vydaná funkcí uživatelské procedury má za následek vyvolání příslušných ukončovacích funkcí rozhraní API. Výstupní funkce lze proto vyvolat rekurzivně. Povšimněte si, že pole *ExitUserArea* v MQAXP a oblast uživatelských řetězců mají rozsah připojení-popisovač. Z toho vyplývá, že funkce uživatelské procedury nemůže tyto oblasti použít k signalizaci na jinou instanci, která byla vyvolána rekurzivně, že je již aktivní.

6. Funkce ukončení mohou také vkládat a získávat zprávy v rámci pracovní jednotky aplikace. Když aplikace potvrdí nebo zazálohuje pracovní jednotku, všechny zprávy v rámci pracovní jednotky jsou potvrzeny nebo vráceny společně, bez ohledu na to, kdo je umístil do pracovní jednotky (aplikace nebo výstupní funkce). Ukončení však může způsobit, že aplikace překročí limity systému dříve, než by tomu bylo jinak (například překročením maximálního počtu nepotvrzených zpráv v pracovní jednotce).

   Když funkce uživatelské procedury používá jednotku práce aplikace tímto způsobem, měla by se výstupní funkce obvykle vyvarovat zadání volání MQCMIT, protože tato akce potvrzuje pracovní jednotku aplikace a může narušit správné fungování aplikace. Funkce uživatelské procedury však může někdy vyžadovat odeslání volání MQBACK, pokud funkce uživatelské procedury narazí na závažnou chybu, která zabraňuje tomu, aby byla jednotka práce potvrzena (například chyba při vkládání zprávy jako součásti pracovní jednotky aplikace). Je-li volána operace MQBACK, dbají na to, aby nebyla změněna hranice pracovní jednotky aplikace. V této situaci funkce uživatelské procedury musí nastavit odpovídající hodnoty, aby bylo zajištěno, že kód dokončení MQCC_WARNING a kód příčiny MQRC_BACKED_OUT jsou vráceny aplikaci, aby mohla aplikace zjistit, že byla pracovní jednotka vrácena.

   Pokud uživatelská funkce používá popisovač připojení k vyvolání volání MQ , tato volání sama o sobě nevedou k dalším vyvolání funkcí ukončení rozhraní API.

7. Je-li funkce uživatelské procedury MQXR_BEFORE ukončena nestandardním způsobem, může být správce front schopen provést zotavení ze selhání. Pokud ano, správce front bude pokračovat ve zpracování, jako by funkce uživatelské procedury vrátila MQXCC_FAILED. Pokud se správce front nemůže zotavit, aplikace se ukončí.

8. Dojde-li k nestandardnímu ukončení funkce ukončení MQXR_AFTER, může být správce front schopen provést zotavení ze selhání. Pokud ano, správce front bude pokračovat ve zpracování, jako by funkce uživatelské procedury vrátila MQXCC_FAILED. Pokud se správce front nemůže zotavit, aplikace se ukončí. Uvědomte si, že v posledním případě jsou zprávy načtené mimo pracovní jednotku 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 zachytává objekt MQCMIT z připraveného procesu MCA a pokusí se provést akci v rámci pracovní jednotky, pak akce selže s kódem příčiny MQRC_UOW_NOT_AVAILABLE.

10. Pro prostředí s více instalacemi platí, že jediným způsobem, jak mít uživatelskou proceduru pro práci s produkty IBM WebSphere MQ 7.0 a IBM WebSphere MQ 7.1 , je zápis výstupu způsobem, který odkazuje na položku IBM WebSphere MQ 7.0 s názvem mqm.Lib , a v případě neprimárních nebo přesunutých uživatelských procedur, aby se zajistilo, že aplikace najde správnou cestu mqm.Lib pro instalaci, se kterou je správce front aktuálně přidružen, před spuštěním aplikace. (Například spusťte příkaz **setmqenv -m QM** před spuštěním aplikace, a to i v případě, že je správce front vlastněn instalací produktu IBM WebSphere MQ 7.0 .)

11. Je-li k dispozici více instalací produktu IBM MQ , použijte výstupní procedury vytvořené pro dřívější verzi produktu IBM MQ, protože nové funkce přidané v novější verzi nemusí pracovat s dřívějšími verzemi. Další informace o změnách mezi verzemi najdete v tématu Co se změnilo v IBM MQ 8.0.

Struktura výstupních parametrů rozhraní API produktu IBM MQ (MQAXP)

Struktura MQAXP, externí řídicí blok, se používá jako vstupní nebo výstupní parametr pro uživatelskou proceduru rozhraní API. Toto téma také poskytuje informace o tom, jak správci front zpracovávají výstupní funkce.

Aplikace MQAXP má následující deklaraci jazyka 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 uživatelské procedury, jehož hodnota je:

```
MQAXP_STRUC_ID.
```

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

Verze (MQLONG)-vstup
Číslo verze struktury s hodnotou:

MQAXP_VERSION_1
Struktura výstupního parametru rozhraní API verze 1.

MQAXP_VERSION_2
Strukturu parametrů uživatelské procedury rozhraní API verze 2.

AKTUÁLNÍ_VERZE MQAXP_
Aktuální číslo verze pro strukturu parametru uživatelské procedury rozhraní API.

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

ExitId (MQLONG)-vstup
Identifikátor ukončení, nastavený při vstupu do uživatelské procedury, označující typ uživatelské procedury:

UŽIVATELSKÁ PROCEDURA MQXT_API_EXIT
Ukončení API.

ExitReason (MQLONG)-vstup
Důvod vyvolání uživatelské procedury, který je nastaven při vstupu do každé výstupní funkce:

PŘIPOJENÍ MQXR_CONNECTION
Probíhá vyvolání uživatelské procedury pro inicializaci před voláním MQCONN nebo MQCONNX nebo po volání funkce MQDISC.

MQXR_PŘED
Ukončení je vyvoláno před provedením volání API, nebo před převodem dat na MQGET.

MQXR_PO
Ukončení je vyvoláno po provedení volání API.

ExitResponse (MQLONG)-výstup
Odezva z uživatelské procedury, která byla inicializována při vstupu do každé výstupní funkce, na:

MQXCC_OK
Pokračujte normálně.

Toto pole musí být nastaveno uživatelskou funkcí pro komunikaci s správce front o výsledku provedení uživatelské procedury. Hodnota musí být jedna z následujících:

MQXCC_OK
Funkce ukončení byla úspěšně dokončena. Pokračujte normálně.

Tato hodnota může být nastavena všemi funkcemi ukončení MQXR_ *. ExitResponse2 se používá k rozhodnutí, zda vyvolat výstupní funkce později v řetězci.

SELHÁNÍ MQXCC_FAILED
Funkce uživatelské procedury se nezdařila kvůli chybě.

Tato hodnota může být nastavena všemi funkcemi ukončení MQXR_ *. Správce front nastaví kód CompCode na hodnotu MQCC_FAILED a důvod pro:

- Funkce MQRC_API_EXIT_INIT_ERROR, pokud je funkce MQ_INIT_EXIT
- Funkce MQRC_API_EXIT_TERM_ERROR, pokud je funkce MQ_TERM_EXIT
- Funkce MQRC_API_EXIT_ERROR pro všechny ostatní funkce ukončení

Sada hodnot může být změněna uživatelskou procedurou později v řetězci.

Hodnota ExitResponse2 se ignoruje; správce front bude pokračovat ve zpracování, jako by byl vrácen objekt MQXR2_SUPPRESS_CHAIN .

FUNKCE MQXCC_SUPPRESS_FUNCTION
Potlačit funkci rozhraní API IBM MQ .

Tuto hodnotu lze nastavit pouze pomocí uživatelské funkce MQXR_BEFORE. Vyvolá volání rozhraní API. Pokud je vrácena programem MQ_DATA_CONV_ON_GET_EXIT, převod dat je vynechán. Správce front nastaví parametr CompCode na hodnotu MQCC_FAILED a důvod MQRC_SUPPRESDAT_B_BY_EXIT, ale sady hodnot lze později v řetězci změnit funkcí uživatelské procedury. Ostatní parametry pro volání zůstávají, protože je výstup opustil. ExitResponse2 se používá k rozhodnutí, zda vyvolat výstupní funkce později v řetězci.

Je-li tato hodnota nastavena v rámci uživatelské funkce MQXR_AFTER nebo MQXR_CONNECTION, správce front bude pokračovat ve zpracování, jako by byla vrácena hodnota MQXCC_FAILED.

FUNKCE MQXCC_SKIP_FUNCTION
Vynechat funkci rozhraní API IBM MQ .

Tuto hodnotu lze nastavit pouze pomocí uživatelské funkce MQXR_BEFORE. Vyvolá volání rozhraní API. Pokud je vrácena programem MQ_DATA_CONV_ON_GET_EXIT, převod dat je vynechán. Výstupní funkce musí nastavit CompCode a příčinu k hodnotám, které se mají vrátit do aplikace, ale sady hodnot mohou být změněny uživatelskou procedurou později v řetězci. Ostatní parametry pro volání zůstávají, protože je výstup opustil. ExitResponse2 se používá k rozhodnutí, zda vyvolat výstupní funkce později v řetězci.

Je-li tato hodnota nastavena v rámci uživatelské funkce MQXR_AFTER nebo MQXR_CONNECTION, správce front bude pokračovat ve zpracování, jako by byla vrácena hodnota MQXCC_FAILED.

UŽIVATELSKÁ PROCEDURA MQXCC_SUPPRESS_EXIT
Potlačit všechny uživatelské funkce náležející k sadě východů.

Tuto hodnotu lze nastavit pouze pomocí ukončovacích funkcí MQXR_BEFORE a MQXR_AFTER. Neobejde *všechna* následná vyvolání ukončovacích funkcí náležících k této sadě uživatelských

procedur pro toto logické připojení. Vynechání tohoto obejití pokračuje, dokud nedojde k logickému požadavku na odpojení, když je vyvolána funkce MQ_TERM_EXIT s parametrem ExitReason MQXR_CONNECTION.

Výstupní funkce musí nastavit CompCode a příčinu k hodnotám, které se mají vrátit do aplikace, ale sady hodnot mohou být změněny uživatelskou procedurou později v řetězci. Ostatní parametry pro volání zůstávají, protože je výstup opustil. ExitResponse2 se ignoruje.

Je-li tato hodnota nastavena pomocí uživatelské funkce MQXR_CONNECTION, správce front bude pokračovat ve zpracování, jako by byla vrácena hodnota MQXCC_FAILED.

Informace o interakci mezi ExitResponse a ExitResponse2a jejím dopadem na ukončení zpracování viz "Jak správci front zpracovávají výstupní funkce" na stránce 1551.

ExitResponse2 (MQLONG)-výstup
Jedná se o sekundární kód odezvy, který kvalifikuje primární kód odezvy pro výstupní funkce MQXR_BEFORE. Inicializuje se na:

```
MQXR2_DEFAULT_CONTINUATION
```

při vstupu do funkce ukončení volání rozhraní API produktu IBM MQ . Poté může být nastavena na jednu z následujících hodnot:

MQXR2_DEFAULT_CONTINUATION
Zda se má pokračovat s dalším ukončením v řetězci, v závislosti na hodnotě ExitResponse.

Má-li parametr ExitResponse hodnotu MQXCC_SUPPRESS_FUNCTION nebo MQXCC_SKIP_FUNCTION, dojde k pozdějšímu předání funkcí ukončení v řetězu MQXR_BEFORE a k odpovídajícím funkcím uživatelské procedury v řetězci MQXR_AFTER. Vyvolání uživatelské procedury v řetězci MQXR_AFTER, které odpovídají funkcím ukončení v řetězci MQXR_BEFORE.

Jinak vyvolejte další uživatelskou proceduru v řetězu.

MQXR2_SUPPRESS_CHAIN
Potlačte řetězec.

Vynechání uživatelských funkcí později v řetězci MQXR_BEFORE a odpovídající uživatelské funkce v řetězci MQXR_AFTER pro toto vyvolání volání rozhraní API. Vyvolání uživatelské procedury v řetězci MQXR_AFTER, které odpovídají funkcím ukončení v řetězci MQXR_BEFORE.

MQXR2_CONTINUE_CHAIN
Pokračujte s další uživatelskou procedurou v řetězci.

Informace o interakci mezi ExitResponse a ExitResponse2a jejím dopadem na ukončení zpracování viz "Jak správci front zpracovávají výstupní funkce" na stránce 1551.

Zpětná vazba (MQLONG)-vstup/výstup
Komunikujte kódy zpětné vazby mezi vyvoláními ukončovacích funkcí. To je inicializováno na:

```
MQFB_NONE (0)
```

před vyvoláním první funkce prvního ukončení v řetězci.

Uživatelské procedury mohou toto pole nastavit na jakoukoli hodnotu, včetně libovolné platné hodnoty MQFB_ * nebo MQRC_ *. Exits může 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.

Oblast ExitUser(MQBYTE16)-vstupní/výstupní
Oblast uživatele, která je k dispozici pro všechny uživatelské procedury přidružené k určitému objektu ExitInfo. Inicializuje se na MQXUA_NONE (binární nuly pro délku oblasti ExitUserArea) před vyvoláním

první uživatelské funkce (MQ_INIT_EXIT) pro připojení Hconn. Od této doby jsou všechny změny provedené v tomto poli pomocí výstupní funkce zachovány napříč voláními funkcí stejného ukončení.

Toto pole je zarovnáno s násobkem 4 MQLONG.

Ukončením mohou také kotvit veškeré úložiště, které přidělíte z této oblasti.

Pro každý kanál hconn má každá výjezd v řetězci uživatelských procedur jinou oblast ExitUser. Oblast ExitUsernemůže být sdílena uživatelskými procedurami v řetězci a obsah oblasti ExitUserpro jednu uživatelskou proceduru není k dispozici pro jinou uživatelskou proceduru v řetězu.

Pro programy v jazyce C je konstanta MQXA_NONE_ARRAY také definována se stejnou hodnotou jako MQXUA_NONE, ale jako pole znaků namísto řetězce.

Délka tohoto pole je dána proměnnou MQ_EXIT_USER_AREA_LENGTH.

ExitData (MQCHAR32)-vstup
Data ukončení, nastavená na vstupu do každé výstupní funkce, k 32 znakům dat specifických pro ukončení, která jsou poskytnuta v uživatelské proceduře. Pokud nedefinujete žádnou hodnotu v tomto poli, budou všechny prázdné znaky.

Délka tohoto pole je dána hodnotou MQ_EXIT_DATA_LENGTH.

Název ExitInfo(MQCHAR48)-vstup
Název informace o ukončení, nastavený na vstupu pro každou uživatelskou funkci do pole ApiExit_name uvedený ve stanzách ve stanzách.

ExitPDArea (MQBYTE48)-vstupní/výstupní
Oblast pro určování problémů, inicializovaná na MQXPDA_NONE (binární nuly pro délku pole) pro každé vyvolání uživatelské procedury.

V případě programů C je konstanta MQXPDA_NONE_ARRAY také definována se stejnou hodnotou jako MQXPDA_NONE, ale jako pole znaků namísto řetězce.

Obslužná rutina ukončení vždy zapíše tuto oblast do trasování IBM MQ na konci ukončení, a to i v případě, že 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, která vyvolala proceduru jako výsledek zpracování volání rozhraní API produktu IBM MQ .

Je-li název správce front zadaný v rámci volání MQCONN nebo MQCONNX prázdný, je toto pole stále nastaveno na název správce front, ke kterému je aplikace připojena, ať už je aplikace server nebo klient.

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

Délka tohoto pole je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

ExitChainAreaPtr (PMQACH)-vstupní/výstupní
Používá se ke komunikaci dat napříč voláními různých výstupních řetězců v řetězci. Je nastaven na ukazatel NULL před vyvoláním první funkce (MQ_INIT_EXIT s ExitReason MQXR_CONNECTION) první uživatelské procedury v řetězci ukončení. Hodnota vrácená uživatelskou procedurou při jednom vyvolání je předána dalšímu vyvolání.

Podrobnější informace o používání oblasti výstupních řetězců naleznete v příručce “Hlavička výstupních řetězců a záhlaví oblasti uživatelských řetězců (MQACH)” na stránce 1555 .

Hconfig (MQHCONFIG)-vstup
Manipulátor konfigurace představující sadu inicializovaných funkcí. Tato hodnota je generována správcem front ve funkci MQ_INIT_EXIT a později je předána do funkce uživatelské procedury rozhraní API. Je nastaven na položku pro každou uživatelskou funkci.

Pomocí Hconfig jako ukazatele na strukturu MQIEP lze provádět volání MQI a DCI. Před použitím parametru HConfig jako ukazatele na strukturu MQIEP musíte zkontrolovat, zda prvních 4 bajtů HConfig odpovídá struktuře StrucId struktury MQIEP.

Funkce (MQLONG)-vstup
Identifikátor funkce, platné hodnoty, pro které jsou konstanty MQXF_ * popsané v "Externí konstanty" na stránce 1556.

Obslužná rutina ukončení nastaví toto pole na správnou hodnotu při každém vstupu do každé funkce ukončení, v závislosti na volání rozhraní API produktu IBM MQ , které vedlo k vyvolání procedury.

ExitMsgHandle (MQHMSG)-vstup/výstup
Má-li funkce MQXF_GET a ExitReason hodnotu MQXR_AFTER, je v tomto poli vrácen platný manipulátor zpráv, který umožňuje přístup k polím deskriptoru zpráv a všechny další vlastnosti odpovídající řetězci ExitProperties určené ve struktuře MQXEPO při registraci uživatelské procedury rozhraní API.

Všechny vlastnosti deskriptoru jiné než zprávy, které jsou vráceny v obslužné rutiny ExitMsg, nebudou k dispozici v objektu MsgHandle ve struktuře MQGMO, pokud byla určena, nebo v datech zprávy.

Je-li funkcí MQXF_GET a ExitReason je MQXR_BEFORE, pokud výstupní program nastaví toto pole na hodnotu MQHM_NONE, potlačí vlastnosti ExitMsgve vlastnostech manipulátoru.

Toto pole není nastaveno, pokud je verze nižší než hodnota MQAXP_VERSION_2.

Jak správci front zpracovávají výstupní funkce

Zpracování prováděné správcem front při návratu z uživatelské funkce závisí na obou funkcích ExitResponse a ExitResponse2.

Tabulka 836 na stránce 1551 shrnuje možné kombinace a jejich účinky pro funkci ukončení MQXR_BEFORE, zobrazující:

- Kdo nastavuje parametry CompCode a parametry příčiny volání rozhraní API
- Určuje, zda jsou vyvolány zbývající uživatelské funkce v řetězci MQXR_BEFORE a odpovídající uživatelské funkce v řetězci MQXR_AFTER.
- Zda je vyvoláno volání API

Pro výstupní funkci MQXR_AFTER:

- CompCode a příčina jsou nastaveny stejným způsobem jako MQXR_BEFORE
- ExitResponse2 se ignoruje (zbývající uživatelské funkce v řetězci MQXR_AFTER jsou vždy vyvolány)
- MQXCC_SUPPRES_FUNCTION a MQXCC_SKIP_FUNCTION nejsou platné

Pro uživatelskou proceduru MQXR_CONNECTION:

- CompCode a příčina jsou nastaveny stejným způsobem jako MQXR_BEFORE
- ExitResponse2 je ignorována.
- MQXCC_SUPPRES_FUNCTION, MQXCC_SKIP_FUNCTION, MQXCC_SUPPRESS_EXIT není platné

Ve všech případech, kdy procedura nebo správce front nastaví kód CompCode a příčinu, může být sada hodnot změněna ukončením vyvolanou později nebo voláním rozhraní API (pokud je volání API vyvoláno později).

Tabulka 836. Zpracování ukončení MQXR_BEFORE

Hodnota ExitResponse	CompCode a příčiny nastavené pomocí	Hodnota řetězce ExitResponse2 (výchozí pokračování)	Hodnota rozhraní API ExitResponse2 (výchozí pokračování)
MQXCC_OK	exit	Y	Y
UŽIVATELSKÁ PROCEDURA MQXCC_SUPPRESS_EXIT	exit	Y	Y

Tabulka 836. Zpracování ukončení MQXR_BEFORE (pokračování)			
Hodnota ExitResponse	**CompCode a příčiny nastavené pomocí**	**Hodnota řetězce ExitResponse2 (výchozí pokračování)**	**Hodnota rozhraní API ExitResponse2 (výchozí pokračování)**
FUNKCE MQXCC_SUPPRESS_FUNCTION	správce front	N	N
FUNKCE MQXCC_SKIP	exit	N	N
SELHÁNÍ MQXCC_FAILED	správce front	N	N

Jak klienti procesu ukončí funkce

Obecně platí, že klienti zpracovávají výstupní funkce stejným způsobem jako serverová aplikace a atribut *QMgrName* v této struktuře platí, zda je funkce na serveru nebo na klientovi.

Klient však nemá žádný koncept souboru *mqs.ini* , takže se neaplikují sekce *ApiExitCommon* a *APIExitTemplate* . Platí pouze stanza *ApiExitLocal* a tento oddíl je nakonfigurován v souboru *mqclient.ini* .

Struktura kontextu uživatelské procedury rozhraní API produktu 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 uživatelské procedury, který má hodnotu MQAXC_STRUC_ID. Pro programy v jazyce C je také definována konstanta MQAXC_STRUC_ID_ARRAY, která má stejnou hodnotu jako MQAXC_STRUC_ID, ale jako pole znaků namísto řetězce.

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

Verze (MQLONG)-vstup
Číslo verze struktury s hodnotou:

MQAXC_VERSION_2
Číslo verze pro strukturu kontextu uživatelské procedury.

AKTUÁLNÍ_VERZE MQAXC_VERSION
Aktuální číslo verze pro strukturu kontextu uživatelské procedury.

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

Prostředí (MQLONG)-vstup
Prostředí, ze kterého bylo vydáno volání rozhraní API produktu IBM MQ , které vedlo k vyvolání výstupní funkce. Platné hodnoty pro toto pole jsou:

MQXE_OTHER
Tato hodnota je konzistentní s vyvoláním uživatelské procedury rozhraní API, pokud je tato uživatelská procedura volána z aplikace serveru. To znamená, že uživatelská procedura rozhraní API je v klientovi ponechána beze změny a nevidí nic jiného.

Pokud uživatelská procedura skutečně potřebuje určit, zda je spuštěna na klientovi, může tato procedura provést tak, že se podívá na pole *ChannelName* a *ChannelDefinition* .

MQXE_MCA
Agent oznamovacího kanálu

MQXE_MCA_SVRCONN
agent kanálu zpráv jednající jménem klienta,

MQXE_PŘÍKAZOVÝ_SERVER
Příkazový server

MQXE_MQSC
Interpret příkazu runmqsc

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci.

UserId (MQCHAR12)-vstup
ID uživatele přidružené k aplikaci. Zejména v případě připojení klienta toto pole obsahuje ID uživatele adoptovaného uživatele, který se liší od ID uživatele, pod kterým je spuštěn kód kanálu. Pokud z klienta neplyne prázdné ID uživatele, nebude provedena žádná změna ID uživatele, které se již používá. To znamená, že není přijato žádné nové ID uživatele.

Obslužná rutina ukončení nastaví toto pole při záznamu na každou výstupní funkci. 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 toto nemusí být efektivní ID uživatele, které klient spouští ve správci front, protože může existovat konfigurace MCAUser nebo CHLAUTH, která změní ID uživatele.

SecurityId (MQBYTE40)-vstup
Rozšíření pro 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 toto nemusí být efektivní ID uživatele, které klient spouští ve správci front, protože může existovat konfigurace MCAUser nebo CHLAUTH, která změní ID uživatele.

ConnectionName (MQCHAR264)-vstup
Pole názvu připojení je nastaveno na adresu klienta. Například pro TCP/IP by to byla IP adresa klienta.

Délka tohoto pole je dána hodnotou MQ_CONN_NAME_LENGTH.

V případě klienta se jedná o partnerskou adresu správce front.

LongMCAUserIdLength (MQLONG)-vstup
Délka dlouhého identifikátoru uživatele MCA.

Pokud se agent MCA připojí ke správci front, je toto pole nastaveno na délku dlouhého identifikátoru uživatele MCA (nebo nula, pokud takový identifikátor neexistuje).

V případě klienta se jedná o dlouhý identifikátor uživatele klienta.

LongRemoteUserIdLength (MQLONG)-vstup
Délka dlouhého vzdáleného identifikátoru uživatele.

Když se agent MCA připojí ke správci front, je toto pole nastaveno na délku dlouhého vzdáleného identifikátoru uživatele. Jinak bude toto pole nastaveno na hodnotu nula.

V případě klienta nastavte toto pole na hodnotu nula.

LongMCAUserIdPtr (MQPTR)-Vstup
Adresa dlouhého identifikátoru uživatele MCA.

Pokud se agent MCA připojí ke správci front, je toto pole nastaveno na adresu dlouhého identifikátoru uživatele MCA (nebo na ukazatel Null, pokud takový identifikátor neexistuje).

V případě klienta se jedná o dlouhý identifikátor uživatele klienta.

LongRemoteUserIdPtr (MQPTR)-input
Adresa dlouhého vzdáleného identifikátoru uživatele.

Když se agent MCA připojí ke správci front, je toto pole nastaveno na adresu dlouhého vzdáleného identifikátoru uživatele (nebo na ukazatel s hodnotou null, pokud takový identifikátor neexistuje).

V případě klienta nastavte toto pole na hodnotu nula.

ApplName (MQCHAR28)-Vstup
Název aplikace nebo komponenty, která vydala volání rozhraní API produktu IBM MQ .

Pravidla pro generování názvu ApplName jsou stejná jako pravidla pro generování výchozího názvu pro požadavek MQPUT.

Hodnota tohoto pole se zjistí dotazem na operační systém pro název programu. Jeho délka je dána hodnotou MQ_APPL_NAME_LENGTH.

ApplType (MQLONG)-vstup
Typ aplikace nebo komponenty, která vydala volání rozhraní API produktu 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 záznamu na každou výstupní funkci.

ProcessId (MQPID)-vstup
Identifikátor procesu operačního systému.

Je-li to vhodné, obslužná rutina ukončení nastaví toto pole na záznam pro každou výstupní funkci.

ThreadId (MQTID)-input
Identifikátor podprocesu MQ . Jedná se o stejný identifikátor, který je použit v trasování MQ a výpisů paměti produktu FFST , ale může se lišit od identifikátoru podprocesu operačního systému.

Je-li to vhodné, obslužná rutina ukončení nastaví toto pole na záznam pro každou výstupní funkci.

ChannelName (MQCHAR)-vstup
Název kanálu doplněný mezerami, je-li to vhodné a známé.

Není-li to vhodné, je toto pole nastaveno na hodnotu NULL.

Reserved1 (MQBYTE4)-vstup
Toto pole je vyhrazené.

ChanneDefinition (PMQCD)-vstup
Ukazatel na použitou definici kanálu, je-li to možné a známé.

Není-li to vhodné, je toto pole nastaveno na hodnotu NULL.

Povšimněte si, že ukazatel je dokončen pouze v případě, že připojení zpracovává v zastoupení kanálu produktu IBM MQ a že byla načtena definice kanálu.

Zejména definice kanálu není na serveru poskytnuta, když je pro kanál vytvořeno první volání MQCONN. Kromě toho, je-li ukazatel vyplněn, struktura (a všechny struktury), na kterou se odkazuje ukazatel, musí být považována za jen pro čtení; každá aktualizace struktury by vedla k nepředvídatelným výsledkům a není podporována.

V případě klienta, pole, která nejsou s hodnotou zadanou pro klienta, obsahují hodnoty, které jsou vhodné pro klientskou aplikaci.

Hlavička výstupních řetězců a záhlaví oblasti uživatelských řetězců (MQACH)

Je-li to nutné, funkce ukončení může získat úložiště pro oblast uživatelských procedur a nastavit ExitChainAreaPtr v MQAXP tak, aby ukazovala na toto úložiště.

Uživatelské procedury (buď stejné nebo odlišné uživatelské funkce) mohou získat více oblastí výstupních řetězců a propojit je dohromady. Oblasti výstupního řetězce musí být přidány nebo odebrány z tohoto seznamu při volání z obslužné rutiny ukončení. Tím je zajištěno, že neexistují žádné problémy se serializací způsobené různými podprocesy přidáváním nebo odebíráním oblastí ze seznamu současně.

Oblast výstupního řetězce musí začínat strukturou záhlaví MQACH, což je deklarace C, pro kterou je toto:

```
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 výstupního řetězce jsou:

StrucId (MQCHAR4)-vstup.
Identifikátor struktury oblasti výstupního řetězce, s počáteční hodnotou definovanou hodnotou MQACH_DEFAULT, z ID MQACH_STRUC_ID.

Pro programy v jazyce C je také definován konstantní MQACH_STRUC_ID_ARRAY; má stejnou hodnotu jako MQACH_STRUC_ID, ale jako pole znaků namísto řetězce.

Verze (MQLONG)-vstup
Číslo verze struktury, jak je uvedeno:

MQACH_VERSION_1
Číslo verze pro strukturu výstupního parametru.

AKTUÁLNÍ_VERZE MQACH_CURRENT_VERSION
Aktuální číslo verze pro strukturu kontextu uživatelské procedury.

Počáteční hodnota tohoto pole, definovaná parametrem MQACH_DEFAULT, je MQACH_CURRENT_VERSION.

Poznámka: Pokud představujete novou verzi této struktury, rozvržení existující součásti se nezmění. Výstupní funkce musí zkontrolovat, zda je číslo verze stejné nebo větší než nejnižší verze obsahující pole, která funkce uživatelské procedury potřebuje použít.

StrucLength (MQLONG)-vstup
Délka struktury MQACH. Pomocí tohoto pole lze určit začátek výstupních dat a nastavit jej na délku struktury vytvořené uživatelskou procedurou.

Počáteční hodnota tohoto pole, definované parametrem MQACH_DEFAULT, je MQACH_CURRENT_LENGTH.

ChainAreaLength (MQLONG)-Vstup
Délka oblasti výstupního řetězce, která je nastavena na celkovou délku aktuální oblasti řetězu ukončení, včetně záhlaví MQACH.

Počáteční hodnota tohoto pole, definovaná parametrem MQACH_DEFAULT, je nula.

Název ExitInfo(MQCHAR48)-vstup
Název informace o ukončení.

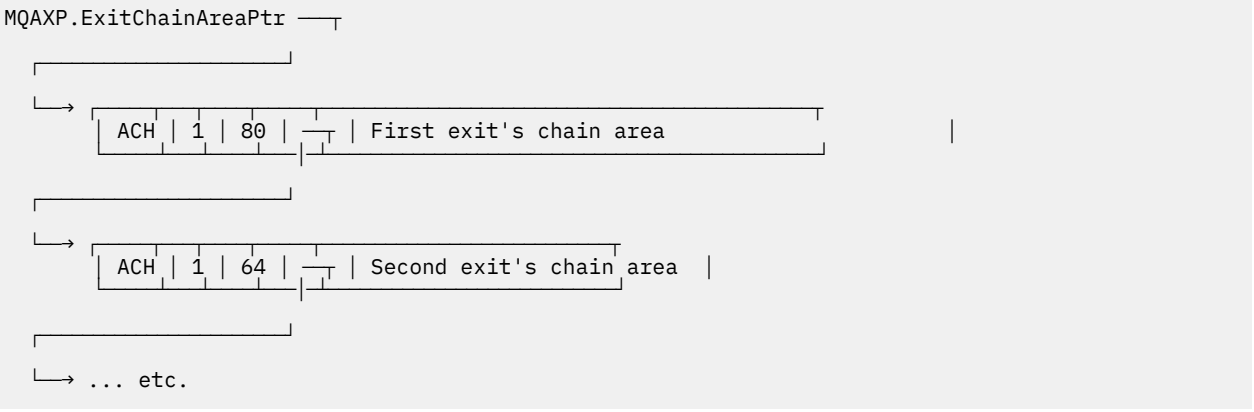
Když uživatelská procedura vytvoří strukturu MQACH, musí inicializovat toto pole vlastním názvem ExitInfo, takže později může být tato struktura MQACH nalezena buď jinou instancí této uživatelské procedury, nebo spolupracujícím výstupem.

Počáteční hodnota tohoto pole, definovaná parametrem MQACH_DEFAULT, je řetězec s nulovou délkou ({""}).

NextChainAreaPtr (PMQACH)-vstup
Ukazatel na další oblast výstupního řetězce s počáteční hodnotou definovanou parametrem MQACH_DEFAULT s hodnotou Null ukazatelem (NULL).

Funkce ukončení musí uvolnit úložiště pro všechny oblasti uživatelských řetězců, které získávají, a manipulovat s ukazateli řetězce k odebrání jejich oblastí výstupních řetězců ze seznamu.

Oblast výstupního řetězce může být konstruována takto:



Externí konstanty

Toto téma se používá jako referenční informace pro externí konstanty dostupné pro rozhraní API.

Pro uživatelské procedury rozhraní API jsou k dispozici následující externí konstanty:

MQXF_ * (identifikátory výstupních funkcí)

```
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_ * (výstupní důvody)

```
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 výstupní funkce vrátilo neplatný kód odezvy, nebo se nějakým způsobem nezdařilo, a správce front nemůže určit další akci, která má být provedena.

Zkontrolujte pole ExitResponse a ExitResponse2 v aplikaci MQAXP, abyste určili špatný kód odezvy, a změňte uživatelskou proceduru tak, aby vracela platný kód odezvy.

MQRC_API_EXIT_INIT_ERROR 2375 X'00000947'
Ve správci front došlo k chybě při inicializaci prováděcího prostředí pro uživatelskou proceduru rozhraní API.

MQRC_API_EXIT_TERM_ERROR 2376 X'00000948'
Správce front zjistil chybu při zavírání prováděcího prostředí pro funkci ukončení rozhraní API.

MQRC_EXIT_REASON_ERROR 2377 X'00000949'
Hodnota pole ExitReason dodaná ve volání procedury registrace bodu předání řízení uživatelskému programu (MQXEP) se vyskytla v chybě.

Zkontrolujte hodnotu v poli ExitReason , abyste určili a opravili nesprávnou hodnotu příčiny ukončení.

MQRC_RESERVED_VALUE_ERROR 2378 X'0000094A'
Hodnota pole Rezervováno je chybná.

Zkontrolujte hodnotu pole Rezervováno, abyste určili a opravili vyhrazenou hodnotu.

Typ C language typedefs

Toto téma obsahuje informace o definici typů asociovaných s uživatelskými procedurami rozhraní API dostupných v jazyce C.

Níže jsou uvedeny definice typů jazyka C přidružené k uživatelským procedurám 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)

Tato část obsahuje informace o vyvolávání MQXEP, vyvolání jazyka MQXEP C a prototypu funkce MQXEP C.

Volání MQXEP použijte k:

1. Registrujte se před a za bodem vyvolání ukončení rozhraní API produktu IBM MQ , ve kterém chcete vyvolat výstupní funkce.
2. Určení vstupních bodů uživatelské funkce
3. Zrušit registraci vstupních bodů uživatelské funkce

Volání funkce MQXEP obvykle kódujete ve funkci uživatelské procedury MQ_INIT_EXIT, ale můžete je zadat v libovolné následné funkci ukončení.

Pokud jste použili volání MQXEP k registraci již registrované funkce ukončení, druhé volání MQXEP se úspěšně dokončí a nahradí registrovanou výstupní funkci.

Pokud k registraci funkce uživatelské procedury MQXEP použijete volání MQXEP, volání MQXEP bude úspěšně dokončeno a funkce uživatelské procedury bude zrušena registrace.

Pokud se volání MQXEP používají při registraci, zrušení registrace a opětovné registraci určité funkce ukončení během životnosti požadavku na připojení, je již dříve registrovaná funkce ukončení znovu aktivována. Veškerá paměť, která je stále přidělena a přidružená k této instanci uživatelské funkce, je k dispozici pro použití funkcemi uživatelské procedury. (Toto úložiště je obvykle uvolněno při 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 funkcí, které jsou inicializovány. Tato hodnota je generována správcem front bezprostředně před vyvoláním funkce MQ_INIT_EXIT a je předávána ve funkci MQAXP pro každou uživatelskou proceduru 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 produktu IBM MQ (MQXR_BEFORE)
- Po volání rozhraní API produktu IBM MQ (MQXR_AFTER)

Funkce (MQLONG)-vstup
Identifikátor funkce, platné hodnoty, pro které jsou konstanty MQXF_ * (viz “Externí konstanty” na stránce 1556).

EntryPoint (PMQFUNC)-vstup
Adresa vstupního bodu pro funkci uživatelské procedury, která má být registrována. Hodnota NULL označuje, že funkce uživatelské procedury nebyla poskytnuta nebo že byla zrušena registrace předchozí registrace funkce ukončení.

ExitOpts(MQXEPO)
Uživatelské procedury rozhraní API mohou určovat volby, které určují způsob registrace uživatelských procedur rozhraní API. Je-li pro toto pole zadán ukazatel Null, předpokládá se výchozí hodnota struktury MQXEPO.

CompCode (MQLONG)-výstup
Kód dokončení, platné hodnoty pro které jsou:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny, který kvalifikuje kód dokončení.

Je-li kód dokončení MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED:

CHYBA MQRC_HCONFIG_ERROR
(2280, X'8E8') Dodaný popisovač konfigurace je neplatný. Použijte popisovač konfigurace z MQAXP.

MQRC_EXIT_REASON_ERROR
(2377, X' 949 ') Dodaná příčina vyvolání výstupní funkce je buď neplatná, nebo není platná pro dodaný identifikátor funkce ukončení.

Buď použijte jednu z platných příčin vyvolání funkce ukončení (hodnota MQXR_ *), nebo použijte platný identifikátor funkce a kombinaci příčiny ukončení. (Viz Tabulka 837 na stránce 1560.)

CHYBA FUNKCE MQRC_FUNCTION_ERROR
(2281, X'8E9') Dodaný identifikátor funkce není platný pro důvod ukončení API. Následující tabulka obsahuje platné kombinace identifikátorů funkcí a ExitReasons.

Tabulka 837. Platné kombinace identifikátorů funkcí a ExitReasons	
Funkce	**ExitReason**
KONFIGURACE MQXF_INIT VÝRAZ MQXF_TERM	PŘIPOJENÍ MQXR_CONNECTION
PŘIPOJENÍ MQXF_CONN MQXF_CONNX DISK MQXF_DISK MQXF_OPEN MQXF_CLOSE MQXF_PUT1 MQXF_PUT MQXF_GET MQXF_INQ SADA MQXF_SET ZAČÁTEK MQXF_ZAČÁTEK VYKOŘITKA MQXF_ MQXF_BACK MQXF_STAT MQXF_CB MQXF_CTL MQXF_CALLBACK MQXF_SUB MQXF_SUBRQ	MQXR_PŘED MQXR_PO
MQXF_DATA_CONV_ON_GET	MQXR_PŘED

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Pokus o registraci nebo deregistraci uživatelské funkce selhal v důsledku problému prostředku.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'893 ') Pokus o registraci nebo zrušení registrace funkce uživatelské procedury neočekávaně selhal.

CHYBA MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatné jméno ExitProperties .

CHYBA MQRC_XEPO_ERROR
(2507, X'09CB') Výstupní struktura voleb není platná.

Vyvolání jazyka C MQXEP

```
MQXEP (Hconfig, ExitReason, Function, EntryPoint, &ExitOpts, &CompCode, &Reason);
```

Prohlášení 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 */
```

Výstupní funkce

Tato část obsahuje některé obecné informace, které vám pomohou při používání volání funkcí a popisuje, jak vyvolat individuální výstupní funkce.

Tyto informace použijte k pochopení obecných pravidel pro uživatelské rutiny rozhraní API a nastavení a vyčištění prostředí pro provádění uživatelské procedury.

Obecná pravidla pro uživatelské procedury rozhraní API

Při vyvolání uživatelských procedur rozhraní API se používají následující obecná pravidla:

- Ve všech případech jsou funkce uživatelské procedury rozhraní API řízeny před ověřením parametrů volání rozhraní API a před všemi kontrolami zabezpečení (v případě MQCONN, MQCONNX nebo MQOPEN).
- Hodnoty polí zadaných do výstupní rutiny a výstupu z této rutiny jsou:
  - Na vstupu do funkce uživatelské procedury rozhraní API produktu IBM MQ *před* , lze hodnotu pole nastavit aplikačním programem nebo předchozí vyvolání funkce ukončení.
  - Na výstupu z funkce uživatelské procedury rozhraní API *před* IBM MQ lze hodnotu pole ponechat beze změny nebo ji nastavit na některou jinou hodnotu uživatelskou funkcí.
  - Na vstupu do funkce uživatelské procedury rozhraní API *po* IBM MQ může být hodnotou pole hodnota nastavená správcem front po zpracování volání API IBM MQ nebo může být nastavena na hodnotu předchozím vyvoláním funkce ukončení v řetězci funkcí uživatelské procedury.
  - Na výstupu z *po* IBM MQ funkci ukončení volání API, hodnota pole může zůstat nezměněná, nebo může být nastavena na nějakou jinou hodnotu uživatelskou funkcí.
- Ukončovací funkce musí komunikovat se správcem front pomocí polí ExitResponse a ExitResponse2 .
- Pole CompCode a Kód příčiny komunikují zpět do aplikace. Funkce správce front a funkce ukončení mohou nastavit pole kódu CompCode a Kód příčiny.
- Volání MQXEP vrací nové kódy příčiny k ukončovacím funkcím, které volají rozhraní MQXEP. Funkce uživatelské procedury však mohou tyto nové kódy příčiny převést na všechny existující kódy příčin, které mohou existující a nové aplikace pochopit.
- Každý prototyp výstupní funkce má podobné parametry pro funkci rozhraní API s extra úrovní indirection s výjimkou CompCode a Reason.
- Uživatelské procedury rozhraní API mohou volat volání MQI (s výjimkou MQDISC), ale tato volání MQI sama o sobě rozhraní API pro vyvolání nepoužívají.

Všimněte si, že bez ohledu na to, zda se aplikace nachází na serveru nebo na klientovi, nelze předpovědět posloupnost volání ukončení rozhraní API. Volání uživatelské procedury rozhraní API BEFORE nemusí být okamžitě následováno voláním AFTER .

Volání BEFORE může být následováno dalším voláním BEFORE . Příklad:

PŘED MQCTL
Před zpětným voláním
PŘED MQPUT
PO MQPUT

PO zpětné volání
PO PŘÍKAZU MQCTL

, nebo

PŘED XAKOPEN
PŘED MQCONNX
PO PŘÍKAZU MQCONNX
PO XAKOPEN

Na straně klienta existuje uživatelská procedura, která může upravit chování volání MQCONN nebo MQCONNX, které se nazývá uživatelská procedura produktu `PreConnect` . Uživatelská procedura `PreConnect` může upravit kterýkoli z parametrů volání MQCONN nebo MQCONNX včetně názvu správce front. Klient nejprve zavolá tuto proceduru a poté vyvolá volání MQCONN nebo MQCONNX. Všimněte si, že pouze počáteční volání MQCONN nebo MQCONNX vyvolá ukončení rozhraní API; všechna následná volání opětovného připojení nemají žádný účinek.

Prováděcí prostředí

Obecně platí, že všechny chyby z ukončovacích funkcí jsou komunikovány zpět do obslužné rutiny ukončení pomocí polí ExitResponse a ExitResponse2 v MQAXP.

Tyto chyby jsou převedeny na hodnoty MQCC_ * a MQRC_ * a jsou předávány zpět aplikaci v polích CompCode a Reason. Nicméně všechny chyby zjištěné v logice obslužné rutiny ukončení se sdělují zpět aplikaci jako hodnoty MQCC_ * a MQRC_ * v polích CompCode a Reason.

Vrátí-li funkce MQ_TERM_EXIT chybu:

- Volání MQDISC již bylo na místě
- Neexistuje žádná jiná příležitost k řízení uživatelské funkce *after* MQ_TERM_EXIT (a tím provést vyčištění prostředí pro provedení ukončení).
- Čištění prováděcího prostředí provedení není provedeno

Uživatelská procedura nemůže být uvolněna, protože by mohla být stále v použití. Další registrované uživatelské procedury dále v řetězci uživatelských procedur, pro které byla ukončena *před* uživatelskou procedurou, budou vedena v opačném pořadí.

Nastavení prováděcího prostředí pro ukončení

Při zpracování explicitního volání MQCONN nebo MQCONNX nastaví logika zpracování ukončení před vyvoláním inicializační funkce uživatelské procedury (MQ_INIT_EXIT) výstupní prostředí pro zpracování ukončení. Nastavení prostředí pro provádění uživatelských procedur zahrnuje načtení uživatelské procedury, získání úložiště pro struktury výstupních parametrů a inicializaci struktur výstupních parametrů. Popisovač konfigurace uživatelské procedury je také přidělen.

Dojde-li během této fáze k chybě, volání MQCONN nebo MQCONNX selže s kódem CompCode MQCC_FAILED a jedním z následujících kódů příčiny:

CHYBA MQRC_API_EXIT_LOAD_ERROR
Pokus o načtení modulu uživatelské procedury rozhraní API se nezdařil.

MQRC_API_EXIT_NOT_FOUND
Funkce uživatelské procedury API nebyla nalezena v modulu uživatelské procedury rozhraní API.

MQRC_STORAGE_NOT_AVAILABLE
Pokus o inicializaci prováděcího prostředí pro funkci uživatelské procedury rozhraní API se nezdařil, protože bylo k dispozici nedostatečné úložiště.

MQRC_API_EXIT_INIT_ERROR
Byla zjištěna chyba při inicializaci prováděcího prostředí pro funkci ukončení rozhraní API.

Vyčištění prováděcího prostředí procedury

Při zpracování explicitního volání MQDISC nebo implicitního požadavku na odpojení v důsledku ukončení aplikace může být nutné po vyvolání funkce ukončení ukončení (MQ_TERM_EXIT) po vyvolání funkce ukončení ukončení (MQ_TERM_EXIT) po vyvolání funkce ukončení ukončení (MQ_TERM_EXIT) vyčistit prostředí pro ukončení zpracování.

Vyčištění prostředí provedení uživatelské procedury zahrnuje uvolnění paměti pro struktury výstupních parametrů, případně odstranění všech modulů, které byly dříve zavedeny do paměti.

Dojde-li během této fáze k chybě, explicitní volání MQDISC selže s kódem CompCode MQCC_FAILED a s následujícím kódem příčiny (chyby nejsou zvýrazněny v implicitních požadavcích na odpojení):

CHYBA MQRC_API_EXIT_TERM_ERROR
Byla zjištěna chyba při zavírání prováděcího prostředí pro funkci ukončení rozhraní API. Ukončení by nemělo vracet žádné selhání z MQDISC před nebo po voláních funkce ukončení 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.

Uživatelská procedura PreConnect

Na klientovi lze použít uživatelskou proceduru PreConnect k vyhledání definice kanálu z centrálního úložiště, jako je například server LDAP.

Uživatelská procedura PreConnect může také upravit libovolný parametr nebo všechny parametry, které se nacházejí ve volání MQCONN nebo MQCONNX, jako například název správce front.

V případě klientských aplikací musí být před uživatelskou procedurou rozhraní API volána uživatelská procedura PreConnect , protože uživatelská procedura MQCONN nebo MQCONNX API se volá pouze tehdy, je-li známý název správce front a tento název lze změnit pomocí uživatelské procedury PreConnect .

Mějte na zřeteli, že volání se vyvolá pouze pro počáteční volání MQCONN nebo MQCONNX.

Vlastnosti uživatelské procedury API

Na serveru mohou uživatelské procedury rozhraní API registrovat strukturu MQXEPO v době inicializace. Struktura MQXEPO obsahuje pole ExitProperties , které uvádí podrobnosti o skupině vlastností, o které se tato uživatelská procedura zajímá. To má za následek generování samostatného manipulátoru vlastností zprávy, který může uživatelská procedura manipulovat odděleně od popisovače vlastnosti zprávy aplikace.

Vlastnosti uživatelské procedury rozhraní API na straně klienta nejsou podporovány. Je-li proveden pokus o registraci názvu skupiny vlastností na straně klienta, funkce selže s kódem příčiny MQRC_EXIT_PROPS_NOT_SUPPORTED.

Vrátit zpět-MQ_BACK_EXIT

Funkce MQ_BACK_EXIT poskytuje funkci ukončení odvolání, která má provést *před* a *po* zpracování odvolání. Použijte identifikátor funkce MQXF_BACK s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení funkce ukončení volání.

Rozhraní k této funkci je:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_BACK_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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

Funkce MQ_BEGIN_EXIT poskytuje funkci zahájení ukončení, která má provést *před* a *po* zpracování volání MQBEGIN. Použít identifikátor funkce MQXF_BEGIN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení volání funkce ukončení volání MQBEGIN.

Rozhraní k této funkci je:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pBeginVolby (PMQBO)-vstupní/výstupní
Ukazatel na začátek voleb.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_BEGIN_EXIT (&ExitParms, &ExitContext, &Hconn, &pBeginOptions, &CompCode,
               &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 MQCALLBACK_EXIT

MQ_CALLBACK_EXIT poskytuje funkci ukončení, která má provést *před* a *po* zpracování zpětného volání. Pomocí identifikátoru funkce MQXF_CALLBACK s ukončovacími příčinami MQXR_BEFORE a MQXR_AFTER zaregistrujte funkce *před* a *po* ukončení volání funkce ukončení volání.

Rozhraní k této funkci je:

```
MQ_CALLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts,
                  &pBuffer, &pMQCBContext)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu uživatelské procedury

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 logicky zavolá proceduru následujícím způsobem:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pMsgDesc, &pGetMsgOpts, &pBuffer,
               &pContext);
```

Váš výstup se musí shodovat s následujícím prototypem 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 odběratele a poté, co byla dokončena zákaznický funkce odběratele. Ačkoli struktury MQMD a MQGMO jsou alterovatelné, změna hodnot ve výstupu před ukončením se znovu neřídí načítání zprávy z fronty, protože tato zpráva již byla odebrána z fronty k doručení do funkce odběratele.

Správa funkcí zpětného volání-MQ_CB_EXIT

MQ_CB_EXIT poskytuje funkci ukončení, která má provést *před* a *po* volání MQCB. Použijte identifikátor funkce MQXF_CB s důvody ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení volání funkce ukončení volání MQCB 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 výstupního parametru

ExitContext (MQAXC)-vstupní/výstupní
Struktura kontextu uživatelské procedury

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení

Operace (MQLONG)-Vstup/výstup
Hodnota operace

pCallbackDesc (PMQCBD)-vstupní/výstupní
Deskriptor zpětného volání

Hobj (MQHOBJ)-vstupní/výstupní
Popisovač objektu

pMsgDesc (PMQMD)-vstupní/výstupní
deskriptor zprávy

pGetMsgOpts (PMQGMO)-vstup/výstup
Volby, které řídí akci MQCB

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_CB_EXIT (&ExitParms, &ExitContext, &Hconn, &Operation, &Hobj, &pMsgDesc,
            &pGetMsgOpts, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 funkce ukončení *před* a *po* zpracování volání MQCLOSE, která má být provedena. Použijte identifikátor funkce MQXF_CLOSE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* funkcích ukončení 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pHobj (PMQHOBJ)-vstup
Ukazatel na popisovač objektu.

Volby (MQLONG)-vstupní/výstupní
Volby zavření.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_CLOSE_EXIT (&ExitParms, &ExitContext,&Hconn, &pHobj, &Options,
               &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 */
```

Potvrdit-MQ_CMIT_EXIT

MQ_CMIT_EXIT poskytuje funkci ukončení potvrzení k provedení *před* a *po* zpracování potvrzení. Použijte identifikátor funkce MQXF_CMIT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení volání funkcí ukončení volání.

Pokud operace potvrzení selže a transakce je vrácena, volání MQCMIT selže s chybou MQCC_WARNING a MQRC_BACKED_OUT. Tyto návratové kódy a kódy příčiny se předávají do libovolných *následujících* výstupních funkcí MQCMIT, aby funkce uživatelské procedury dala indikaci, že byla jednotka práce vrácena.

Rozhraní k této funkci je:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_CMIT_EXIT (&ExitParms, &ExitContext,&Hconn, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 pro výstupní funkci MQXF_GET i pro výstupní funkci “MQXF_DATA_CONV_ON_GET” na stránce 1576 .

   Pro tyto dvě výstupní funkce jsou definovány samostatné vstupní body, aby bylo možné zachytit *oba* , že volání MQXEP musí být použito dvakrát; pro toto volání je použit identifikátor funkce MQXF_GET.

   Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro objekty MQXF_GET a MQXF_DATA_CONV_ON_GET, lze pro obě funkce použít jednu funkci ukončení; pole *Function* ve struktuře MQAXP označuje, která výstupní funkce byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých ukončovacích funkcí pro tyto dva případy.

Rozšíření připojení a připojení-MQ_CONNX_EXIT

MQ_CONNX_EXIT poskytuje funkci ukončení připojení pro provádění zpracování *před* a *po* zpracování MQCONN a funkce ukončení rozšíření připojení pro provedení *před* a *po* zpracování MQCONNX.

Stejné rozhraní, jak je popsáno zde, je vyvoláno pro funkce ukončení volání MQCONN a MQCONNX.

Když agent kanálu zpráv (MCA) odpovídá na připojení příchozího klienta, může se agent MCA připojit a vytvořit několik volání rozhraní API produktu IBM MQ před tím, než je stav klienta plně známý. Tato volání API volají funkce uživatelské procedury rozhraní API s rozhraním MQAXC na základě samotného programu MCA (například v polích UserId a ConnectionName v souboru MQAXC).

Když agent MCA odpoví na další příchozí volání rozhraní API klienta, struktura MQAXC je založena na příchozím klientovi a odpovídajícím způsobem nastaví pole UserId a ConnectionName .

Název správce front nastavený aplikací na volání MQCONN nebo MQCONNX je předán do volání připojaného připojení. Jakýkoliv pokus *před* MQ_CONNX_EXIT ke změně názvu správce front nemá žádný účinek.

Použijte identifikátory funkcí MQXF_CONN a MQXF_CONNX s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení funkcí volání MQCONN a MQCONNX.

Uživatelská procedura MQ_CONNX_EXIT volaná z důvodu MQXR_BEFORE *nesmí* volat žádné volání rozhraní API produktu IBM MQ , protože v tomto okamžiku nebylo nastaveno správné prostředí.

MQ_CONNX_EXIT nemůže volat MQDISC z volání uživatelské procedury API pro připojení, pro které se volá. Toto omezení lze použít pro uživatelské procedury rozhraní API klienta i serveru.

Rozhraní pro MQCONN a MQCONNX je identické:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOpts,
&pHconn, &CompCode, &Reason);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

pQMgrNázev (PMQCHAR)-vstup
Ukazatel na název správce front dodaný v rámci volání MQCONNX. Uživatelská procedura nesmí změnit tento název ve volání MQCONN nebo MQCONNX.

pConnectOpts (PMQCNO)-vstupní/výstupní
Ukazatel na volby, které řídí akci volání MQCONNX.

Podrobnosti viz “MQCNO-Volby připojení” na stránce 315.

Pro výstupní funkci MQXF_CONN odkazuje příkaz pConnectOpts k výchozí struktuře voleb připojení (MQCNO_DEFAULT).

pHconn (PMQHCONN)-vstup
Ukazatel na popisovač připojení.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení)

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_CONNX_EXIT (&ExitParms, &ExitContext, &pQMgrName, &pConnectOps,
               &pHconn, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 funkční rozhraní MQ_CONNX_EXIT je použito pro volání MQCONN i pro volání MQCONNX. Pro tyto dvě volání jsou však definována samostatná vstupní body. Chcete-li zachycovat *obě* volání, volání MQXEP musí být použito alespoň dvakrát s identifikátorem funkce MQXF_CONN a znovu s MQXF_CONNX.

   Vzhledem k tomu, že rozhraní MQ_CONNX_EXIT je stejné pro volání MQCONN a MQCONNX, lze pro obě volání použít jednu funkci uživatelské procedury; pole *Function* ve struktuře MQAXP označuje, které volání probíhá. Alternativně lze volání MQXEP použít k registraci různých výstupních funkcí pro obě volání.
2. Když agent kanálu zpráv (MCA) odpovídá na příchozí připojení klienta, může agent MCA zadat počet volání produktu MQ před tím, než je stav klienta plně známý. Tyto výzvy MQ vedou k vyvolání funkcí ukončení rozhraní API ve struktuře MQAXC obsahující data související s agentem MCA a nikoli pro klienta (například identifikátor uživatele a název připojení). Jakmile je však stav klienta plně známý, budou následné volání funkce MQ výsledkem vyvolání funkcí ukončení rozhraní API s příslušnými daty klienta ve struktuře MQAXC.
3. Všechny výstupní funkce MQXR_BEFORE jsou vyvolány před provedením jakýchkoli ověření platnosti parametru správcem front. Parametry mohou být proto neplatné (včetně neplatných ukazatelů pro adresy parametrů).

   Funkce MQ_CONNX_EXIT je vyvolána před tím, než správce front provede jakékoli kontroly autorizace.
4. Funkce uživatelské procedury nesmí změnit název správce front určeného v rámci volání MQCONN nebo MQCONNX. Je-li název změněn funkcí uživatelské procedury, výsledky nejsou definovány.
5. Uživatelská funkce MQXR_BEFORE pro MQ_CONNX_EXIT nemůže vydat volání MQ jinou než MQXEP.

Řízení zpětného volání-MQ_CTL_EXIT

MQ_CTL_EXIT poskytuje funkci uživatelské procedury požadavku na odběr, která má provést *před* a *po* zpracování zpětného volání řídicího prvku. Použijte identifikátor funkce MQXF_CTL s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení funkce zpětného 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 zpracovávaná na zpětném volání definovaném pro zadaný popisovač objektu

vstup/výstup ControlOpts (MQCTLO)
Volby, které řídí akci MQCTL

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_CTL_EXIT (&Hconn, &Operation, &ControlOpts, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 ukončení odpojení, která má provést *před* a *po* zpracování ukončení MQDISC. Použijte identifikátor funkce MQXF_DISC s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení funkcí volání funkce MQDISC.

Rozhraní k této funkci je

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
&CompCode, &Reason);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

pHconn (PMQHCONN)-vstup
Ukazatel na popisovač připojení.

*Pro volání před MQDISC*je hodnota tohoto pole jedna z následujících hodnot:

- Manipulátor připojení vrácený při volání MQCONN nebo MQCONNX

- Zero, pro prostředí, kde je adaptér specifický pro prostředí připojen ke správci front
- Hodnota nastavená při předchozím vyvolání funkce uživatelské procedury

*Pro volání po volání MQDISC*je hodnota tohoto pole nula nebo hodnota nastavená předchozím vyvoláním funkce ukončení.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_DISC_EXIT (&ExitParms, &ExitContext, &pHconn,
              &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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

MQ_GET_EXIT poskytuje funkci získání uživatelské procedury, která má provést *před* a *po* zpracování volání MQGET.

Jsou zde dva identifikátory funkce:

1. Pomocí MQXF_GET s důvody ukončení MQXR_BEFORE a MQXR_AFTER zaregistrujte *před* a *po* ukončení funkcí volání MQGET.
2. Informace o použití identifikátoru funkce MQXF_DATA_CONV_ON_GET naleznete v příručce “MQXF_DATA_CONV_ON_GET” na stránce 1576 .

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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstupní/výstupní
Popisovač objektu.

pMsgDesc (PMQMD)-vstupní/výstupní
Ukazatel na deskriptor zprávy.

pGetMsgOpts (PMQGMO)-vstup/výstup
Ukazatel pro získání voleb zpráv.

BufferLength (MQLONG)-vstupní/výstupní
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

pDataLength (PMQLONG)-vstupní/výstupní
Ukazatel na pole délky dat.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_GET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pGetMsgOpts, &BufferLength, &pBuffer, &pDataLength,
             &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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 pro výstupní funkci MQXF_GET i pro výstupní funkci “MQXF_DATA_CONV_ON_GET” na stránce 1576 .

   Pro tyto dvě výstupní funkce jsou definovány samostatné vstupní body, aby bylo možné zachytit *oba* , že volání MQXEP musí být použito dvakrát; pro toto volání je použit identifikátor funkce MQXF_GET.

   Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro objekty MQXF_GET a MQXF_DATA_CONV_ON_GET, lze pro obě funkce použít jednu funkci ukončení; pole *Function* ve struktuře MQAXP označuje, která výstupní funkce byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých ukončovacích funkcí pro tyto dva případy.

MQXF_DATA_CONV_ON_GET

Identifikátor funkce MQXF_DATA_CONV_ON_GET se používá s hodnotou MQ_GET_EXIT.

Informace o rozhraní k tomuto volání a ukázkové deklaraci jazyka C najdete v tématu MQ_GET_EXIT .

Poznámky k použití

Je-li registrována, tento vstupní bod se zavolá, když se zprávy dorazí do aplikace, ale před jakýmkoli převodem dat. To může být užitečné, pokud uživatelská procedura rozhraní API potřebuje provést zpracování, jako je dešifrování nebo dekomprimace, než se zpráva předá do převodu dat. Uživatelská procedura může v případě potřeby způsobit, že převod dat bude vynechán návratem MQXCC_SUPPRES_FUNCTION;, kde získáte další informace, viz struktura MQAXP .

Registrace pro tento vstupní bod na klientovi má za následek, že převod dat bude proveden lokálně na klientském počítači. Pro správnou operaci by proto mohla být nutná instalace uživatelských procedur pro převod aplikací na straně klienta. Nezapomeňte, že pro asynchronní spotřebu je použit také objekt MQXF_DATA_CONV_GET ON_GET.

Při použití volání MQ_GET_EXITpoužijte položku MQXF_DATA_CONV_ON_GET s příčinou ukončení MQXR_BEFORE, aby bylo možné zaregistrovat funkci ukončení převodu dat *před* MQGET.

Pro funkci MQXF_DATA_CONV_ON_GET není k dispozici žádná výstupní funkce MQXR_AFTER; funkce ukončení MQXR_AFTER pro funkci MQXF_GET poskytuje požadovanou schopnost zpracování ukončení po převodu dat.

Pro volání MQ_GET_EXITjsou definovány oddělené vstupní body, aby bylo možné zachytit *obě* uživatelské funkce, musí být volání MQXEP použito dvakrát; pro tento hovor bude použit identifikátor funkce MQXF_DATA_CONV_GET ON_GET.

Vzhledem k tomu, že rozhraní MQ_GET_EXIT je stejné pro objekty MQXF_GET a MQXF_DATA_CONV_ON_GET, lze pro obě funkce použít jednu funkci ukončení; pole *Function* ve struktuře MQAXP označuje, která výstupní funkce byla vyvolána. Alternativně lze volání MQXEP použít k registraci různých ukončovacích funkcí pro tyto dva případy.

Inicializace-MQ_INIT_EXIT

MQ_INIT_EXIT poskytuje inicializaci na úrovni připojení označeným nastavením ExitReason v MQAXP do MQXR_CONNECTION.

Během inicializace si všimněte následujících položek:

- Funkce MQ_INIT_EXIT volá aplikaci MQXEP k registraci příkazových slov rozhraní IBM MQ API a bodů ENTRY a EXIT, v nichž má zájem.
- Ukončí příkaz pro zachycení všech příkazových slov rozhraní API IBM MQ . Funkce ukončení jsou vyvolány pouze v případě, že byl zaregistrován zájem.
- Paměť, která má být použita při ukončení, může být získána při inicializaci.
- Pokud volání této funkce selže, volání MQCONN nebo MQCONNX, které je vyvoláno, selže také s kódem CompCode a s odůvodněním, které závisí na hodnotě pole ExitResponse v MQAXP.
- Uživatelská procedura MQ_INIT_EXIT nesmí vydat volání rozhraní API produktu IBM MQ , protože v tomto okamžiku nebylo nastaveno správné prostředí.
- Pokud došlo k selhání příkazu MQ_INIT_EXIT s chybou MQXCC_FAILED, vrátí se správce front z volání MQCONN nebo MQCONNX, které bylo voláno, s MQCC_FAILED a MQRC_API_EXIT_ERROR.
- Pokud správce front zjistí chybu při inicializaci prováděcího prostředí funkce ukončení rozhraní API před vyvoláním první proměnné MQ_INIT_EXIT, vrátí se správce front z volání MQCONN nebo MQCONNX, které vyvolalo volání MQ_INIT_EXIT s funkcí 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

CompCode (MQLONG)-vstupní/výstupní
Ukazatel na kód dokončení, platné hodnoty pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Ukazatel na kód příčiny, který kvalifikují kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli platnou hodnotu MQRC_ *.

Hodnota CompCode a příčina vrácená aplikaci závisí na hodnotě pole ExitResponse v 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_INIT_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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 vydat volání MQXEP pro registraci adres funkcí ukončení pro konkrétní volání MQ , která mají být zachycena. Není nutné zachytávat všechna volání MQ nebo zachytávat volání MQXR_BEFORE a MQXR_AFTER. Například, výstupní sada může zvolit zachycení pouze volání MQXR_BEFORE příkazu MQPUT.
2. Úložiště, které má být použito funkcemi ukončení ve výstupní sadě, může být získáno pomocí funkce MQ_INIT_EXIT. Funkce uživatelské procedury mohou případně získávat paměť při jejich vyvolání a v případě potřeby i tyto funkce. Před ukončením uživatelské procedury by však měla být uvolněna veškerá paměť; funkce MQ_TERM_EXIT může uvolnit paměť nebo se dříve vyvolá uživatelská procedura ukončení.
3. Pokud hodnota MQ_INIT_EXIT vrátí hodnotu MQXCC_FAILED v poli `ExitResponse` MQAXP nebo selže jiným způsobem, volání MQCONN nebo MQCONNX, které způsobilo vyvolání MQ_INIT_EXIT, také selže, s parametry **CompCode** a **Reason** nastaveným na odpovídající hodnoty.
4. Funkce MQ_INIT_EXIT nemůže vydat volání MQ jiná než MQXEP.

Dotaz-MQ_INQ_EXIT

MQ_INQ_EXIT poskytuje funkci ukončení dotazu, která má provést *před* a *po* zpracování volání MQINQ. Použijte identifikátor funkce MQXF_INQ s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* funkcích ukončení volání MQINQ 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup
Popisovač objektu.

SelectorCount (MQLONG)-vstup
Počet selektorů

pSelectors (PMQLONG)-vstupní/výstupní
Ukazatel na pole hodnot selektoru.

Počet IntAttrCount (MQLONG)-input
Počet celočíselných atributů.

pIntAttrs (PMQLONG)-vstupní/výstupní
Ukazatel na pole celočíselných hodnot atributu.

CharAttrDélka (MQLONG)-vstupní/výstupní
Délka pole znakového atributu.

pCharAttrs (PMQCHAR)-vstupní/výstupní
Ukazatel na pole znakových atributů.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_INQ_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
```

```
              &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
              &pCharAttrs, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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 otevřenou funkci ukončení, která má provést *před* a *po* zpracování volání MQOPEN. Použijte identifikátor funkce MQXF_OPEN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* funkcích ukončení volání MQOPEN 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pObjDesc (PMQOD)-vstupní/výstupní
Ukazatel na deskriptor objektu.

Volby (MQLONG)-vstupní/výstupní
Volby otevření.

pHobj (PMQHOBJ)-vstup
Ukazatel na popisovač objektu.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &Options,
              &pHobj, &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 */
```

Put-MQ_PUT_EXIT

MQ_PUT_EXIT poskytuje funkci put exit, která má provést *před* a *po* zpracování volání MQPUT. Pomocí identifikátoru funkce MQXF_PUT s důvody ukončení MQXR_BEFORE a MQXR_AFTER zaregistrujte *před* a *po* ukončení volání funkce 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstupní/výstupní
Popisovač objektu.

pMsgDesc (PMQMD)-vstupní/výstupní
Ukazatel na deskriptor zprávy.

pPutMsgOpts (PMQPMO)-vstup/výstup
Ukazatel pro vložení voleb zpráv.

BufferLength (MQLONG)-vstupní/výstupní
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_PUT_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &pMsgDesc,
             &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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 vynechávají běžné zpracování volání. V důsledku toho nemohou být takové zprávy zachyceny funkcí MQ_PUT_EXIT nebo funkce MQPUT1 . Nicméně zprávy sestavy generované agentem kanálu zpráv se zpracovávají normálně, a proto je lze zachytit pomocí

funkce MQ_PUT_EXIT nebo funkce MQ_PUT1_EXIT . Chcete-li zajistit zachycení všech zpráv sestav generovaných agentem MCA, měly by být použity jak MQ_PUT_EXIT, tak i MQ_PUT1_EXIT .

Put1 - MQ_PUT1_EXIT

MQ_PUT1_EXIT poskytuje funkci uživatelské procedury *vložit pouze jednu zprávu* , která má provést *před* a *po* zpracování volání MQPUT1 . Použijte identifikátor funkce MQXF_PUT1 s výstupnými příčinami MQXR_BEFORE a MQXR_AFTER pro registraci *before* a *after* MQPUT1 volání ukončení volání.

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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pObjDesc (PMQOD)-vstupní/výstupní
Ukazatel na deskriptor objektu.

pMsgDesc (PMQMD)-vstupní/výstupní
Ukazatel na deskriptor zprávy.

pPutMsgOpts (PMQPMO)-vstup/výstup
Ukazatel pro vložení voleb zpráv.

BufferLength (MQLONG)-vstupní/výstupní
Délka vyrovnávací paměti zpráv.

pBuffer (PMQBYTE)-vstupní/výstupní
Ukazatel na vyrovnávací paměť zpráv.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_PUT1_EXIT (&ExitParms, &ExitContext, &Hconn, &pObjDesc, &pMsgDesc,
              &pPutMsgOpts, &BufferLength, &pBuffer, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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

Funkce MQ_SET_EXIT poskytuje funkci uživatelské procedury pro provedení zpracování volání *před* a *po* zpracování volání MQSET. Použijte identifikátor funkce MQXF_SET s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

Hobj (MQHOBJ)-vstup
Popisovač objektu.

SelectorCount (MQLONG)-vstup
Počet selektorů

pSelectors (PMQLONG)-vstupní/výstupní
Ukazatel na pole hodnot selektoru.

Počet IntAttrCount (MQLONG)-input
Počet celočíselných atributů.

pIntAttrs (PMQLONG)-vstupní/výstupní
Ukazatel na pole celočíselných hodnot atributu.

CharAttrDélka (MQLONG)-vstupní/výstupní
Délka pole znakového atributu.

pCharAttrs (PMQCHAR)-vstupní/výstupní
Ukazatel na hodnoty znakových atributů.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_SET_EXIT (&ExitParms, &ExitContext, &Hconn, &Hobj, &SelectorCount,
             &pSelectors, &IntAttrCount, &pIntAttrs, &CharAttrLength,
             &pCharAttrs, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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, která má provést *před* a *po* zpracování volání MQSTAT. Použijte identifikátor funkce MQXF_STAT s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení funkcí ukončení 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

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)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli platnou hodnotu MQRC_ *.

Vyvolání jazyka C

Váš výstup se musí shodovat s následujícím prototypem 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 na úrovni připojení, registrované s identifikátorem funkce MQXF_TERM a ExitReason MQXR_CONNECTION. Je-li zaregistrován, hodnota MQ_TERM_EXIT je volána jednou pro každý požadavek na odpojení.

V rámci ukončení je možné uvolnit úložiště, které již nelze ukončit, a může být provedeno jakékoli vyčištění.

Pokud funkce MQ_TERM_EXIT selže s chybou MQXCC_FAILED, vrátí se správce front z MQDISC, který ji volal pomocí funkce MQCC_FAILED a MQRC_API_EXIT_ERROR.

Pokud správce front zjistí chybu při ukončování prováděcího prostředí funkce ukončení rozhraní API po vyvolání poslední proměnné MQ_TERM_EXIT, vrátí správce front z volání MQDISC, které vyvolalo výjimku 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli platnou hodnotu MQRC_ *.

Hodnota CompCode a příčina vrácená aplikaci závisí na hodnotě pole ExitResponse v 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_TERM_EXIT (&ExitParms, &ExitContext, &CompCode, &Reason)
```

Váš výstup se musí shodovat s následujícím prototypem 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 výstupní sada zaregistrovala ukončení ukončení, pokud není zpracování ukončení dokončeno.

   Pokud funkce náležející do výstupní sady získají prostředky během připojení, funkce MQ_TERM_EXIT je pohodlným bodem, v němž mohou uvolnit tyto prostředky, například uvolnění dynamicky získaného úložiště.
2. Je-li při volání MQDISC registrována funkce MQ_TERM_EXIT, je po vyvolání všech návratných funkcí MQDISC vyvolána funkce ukončení.
3. Pokud funkce MQ_TERM_EXIT vrátí hodnotu MQXCC_FAILED v poli `ExitResponse` MQAXP nebo selže jiným způsobem, volání MQDISC, které způsobilo vyvolání MQ_TERM_EXIT, selže také s parametry **CompCode** a **Reason** nastaveným na odpovídající hodnoty.

Registrovat odběr-MQ_SUB_EXIT

MQ_SUB_EXIT poskytuje funkci ukončení, která má provést *před* a *po* zpracování registrace. Použijte identifikátor funkce MQXF_SUB s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení registračních funkcí registrationvolání 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení.

pSubsest-vstup/výstup
Pole selektorů atributů.

pHobj -vstupní/výstupní
Popisovač objektu

pHsub (MQHOBJ) vstupní/výstupní
Popisovač odběru

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_SUB_EXIT (&ExitParms, &ExitContext, &Hconn, &pSubDesc, &pHobj, &pHsub,
             &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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, která má provést zpracování *před* a *po* zpracování požadavku na odběr. Použijte identifikátor funkce MQXF_SUBRQ s ukončovacími příčinami MQXR_BEFORE a MQXR_AFTER pro registraci *před* a *po* ukončení volání funkcí ukončení volání.

Rozhraní k této funkci je:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstupní/výstupní
Manipulátor připojení.

pHsub (MQHOBJ) vstupní/výstupní
Popisovač odběru

Vstup/výstup akce (MQLONG)
Akce

pSubRqOpts (MQSRO) I/O

CompCode (MQLONG)-vstupní/výstupní
Kód dokončení, platné hodnoty, pro které jsou:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo

Důvod (MQLONG)-vstupní/výstupní
Kód příčiny opravňující kód dokončení.

Je-li kód dokončení MQCC_OK, jediná platná hodnota je:

MQRC_NONE
(0, x '000') Chybí důvod k vytvoření sestavy.

Je-li kód dokončení MQCC_FAILED nebo MQCC_WARNING, může funkce uživatelské procedury nastavit pole s kódem příčiny na jakoukoli 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 logicky zavolá proceduru následujícím způsobem:

```
MQ_SUBRQ_EXIT (&ExitParms, &ExitContext, &Hconn, &pHsub, &Action, &pSubRqOpts,
               &CompCode, &Reason);
```

Váš výstup se musí shodovat s následujícím prototypem 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 zpracováním xa_close a po něm. Použijte identifikátor funkce MQXF_XACLOSE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce ukončení 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXa_info (PMQCHAR)-vstupní/výstupní
Informace o správci prostředků specifické pro instanci.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_CLOSE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 funkci ukončení xa_commit, která má být provedena před zpracováním xa_commit a po něm. Použijte identifikátor funkce MQXF_XACOMMIT s ukončovacími příčinami MQXR_BEFORE a MQXR_AFTER pro registraci před ukončovacími funkcemi volání xa_commit a po něm.

Rozhraní k této funkci je:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_COMMIT_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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

Funkce XA_COMPLETE_EXIT poskytuje funkci ukončení xa_complete, která má být provedena před zpracováním a po zpracování xa_complete. Použijte identifikátor funkce MQXF_XACOMPLETE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_complete volání ukončení.

Rozhraní k této funkci je:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pHandle (PMQLONG)-vstupní/výstupní
Ukazatel na asynchronní operaci.

pRetVal (PMQLONG)-vstupní/výstupní
Návratová hodnota asynchronní operace.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_COMPLETE_EXIT (&ExitParms, &ExitContext, &Hconn, &pHandle, &pRetval, &Rmid, &Flags,
&XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 má být provedena před a po zpracování xa_end. Použijte identifikátor funkce MQXF_XAEND s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_end volání ukončení.

Rozhraní k této funkci je:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_END_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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_FORGET_EXIT

Funkce XA_FORGET_EXIT poskytuje funkci ukončení xa_forget, která má být provedena před zpracováním xa_forget a po něm. Použijte identifikátor funkce MQXF_XAFORGET s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_forget volání ukončení volání.

Rozhraní k této funkci je:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_FORGET_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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

Funkce XA_OPEN_EXIT poskytuje funkci ukončení xa_open, která má být provedena před zpracováním xa_open a po něm. Použijte identifikátor funkce MQXF_XAOPEN s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_open volání ukončení.

Rozhraní k této funkci je:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXa_info (PMQCHAR)-vstupní/výstupní
Informace o správci prostředků specifické pro instanci.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_OPEN_EXIT (&ExitParms, &ExitContext, &Hconn, &pXa_info, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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

Funkce XA_PREPARE_EXIT poskytuje funkci ukončení xa_prepare, která má být provedena před zpracováním xa_prepare a po něm. Použijte identifikátor funkce MQXF_XAPREPARE s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_prepare volání funkce ukončení volání.

Rozhraní k této funkci je:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_PREPARE_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 zpracováním xa_recover a po něm. Použijte identifikátor funkce MQXF_XARECONVER s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_recover volání funkce 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 výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Počet (MQLONG)-vstupní/výstupní
Maximální počet XID v poli XID

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_RECOVER_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Count, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 ukončení xa_rollback, která má být provedena před zpracováním xa_rollback a po něm. Použijte identifikátor funkce MQXF_XAROLLBACK s příčinami ukončení MQXR_BEFORE a MQXR_AFTER, abyste zaregistrovali před a po ukončení funkce xa_rollback výstupní funkce.

Rozhraní k této funkci je:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_ROLLBACK_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 funkci ukončení xa_start, která má být provedena před zpracováním xa_start a po něm. Použijte identifikátor funkce MQXF_XASTART s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce xa_start pro volání ukončení.

Rozhraní k této funkci je:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
XA_START_EXIT (&ExitParms, &ExitContext, &Hconn, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 */
```

axi_reg-AX_REG_EXIT

Volání AX_REG_EXIT poskytuje funkci ax_reg před a po zpracování axi_reg. Použijte identifikátor funkce MQXF_AXREG s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení funkce volání funkce ax_reg.

Rozhraní k této funkci je:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode)
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Hconn (MQHCONN)-vstup
Manipulátor připojení.

pXID (MQPTR)-vstup/výstup
ID větve transakce.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
AX_REG_EXIT (&ExitParms, &ExitContext, &pXID, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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

Volání AX_UNREG_EXIT poskytuje funkci ax_unreg k provedení před a po zpracování axi_unreg. Použijte identifikátor funkce MQXF_AXUNREG s příčinami ukončení MQXR_BEFORE a MQXR_AFTER pro registraci před a po ukončení volání funkce ukončení volání funkce ax_unreg.

Rozhraní k této funkci je:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

kde parametry jsou:

ExitParms (MQAXP)-vstup/výstup
Struktura výstupního parametru.

ExitContext (MQAXC)-vstupní/výstupní
Ukončení struktury kontextu.

Rmid (MQLONG)-vstupní/výstupní
Identifikátor správce prostředků.

Příznaky (MQLONG)-vstupní/výstupní
Volby správce prostředků.

XARetCode (MQLONG)-vstupní/výstupní
Odezva na 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 logicky zavolá proceduru následujícím způsobem:

```
AX_UNREG_EXIT (&ExitParms, &ExitContext, &Rmid, &Flags, &XARetCode);
```

Váš výstup se musí shodovat s následujícím prototypem 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 uživatelské procedury

Toto téma obsahuje obecné pokyny, které vám pomohou naplánovat vaše východy, zejména související s obsluhováním chyb a neočekávaných událostí.

Selhání ukončení

Je-li funkce ukončení nestandardně ukončena po destruktivním nestandardním volání MQGET, ale před předáním zprávy do aplikace, obslužná rutina ukončení se může zotavit ze selhání a předat řízení aplikaci.

V takovém případě může dojít ke ztrátě zprávy. To se podobá tomu, co se stane, když aplikace selže bezprostředně po přijetí zprávy z fronty.

Volání MQGET může být dokončeno s funkcí MQCC_FAILED a MQRC_API_EXIT_ERROR.

Je-li funkce ukončení volání rozhraní API *před* ukončena nestandardním způsobem, obslužná rutina ukončení se může zotavit ze selhání a předat řízení aplikaci bez zpracování volání rozhraní API. V případě této události musí funkce uživatelské procedury obnovit všechny prostředky, které vlastní.

Pokud se používají zřetězené uživatelské procedury, *po* ukončení volání rozhraní API pro všechny *před* uživatelskou procedurou rozhraní API, které bylo úspěšně řízeno, může být motivované k řízení. Volání rozhraní API může selhat s chybou MQCC_FAILED a MQRC_API_EXIT_ERROR.

Příklad ošetření chyb pro funkce ukončení

Následující diagram zobrazuje body (e *N*) v jaké chybě může dojít k chybám. Je to jen příklad, jak ukázat, jak se chování se chová a měly by být čteny společně s následující tabulkou. V tomto příkladu jsou dvě ukončovací funkce vyvolány jak před, tak po každém volání rozhraní API, aby se zobrazoval 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
```

Následující tabulka obsahuje seznam akcí, které mají být provedeny v každém bodě chyby. Byla pokryta pouze část chybových bodů, protože se zde uvedená pravidla mohou vztahovat na všechny ostatní. Jedná se o akce, které určují zamýšlené chování v jednotlivých případech.

Tabulka 838. Výstupní chyby rozhraní API a odpovídající akce, které je třeba provést

Err Pt	Popis	Akce
e1	Chyba při nastavování nastavení prostředí.	1. Anulovat nastavení prostředí podle potřeby 2. Jednotka bez ukončovacích funkcí 3. Selhání MQCONN s MQCC_FAILED, MQRC_API_EXIT_LOAD_ERROR
e2	Funkce MQ_INIT_EXIT je dokončena s: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Vyčistit prostředí 2. Selhání MQCONN s MQCC_FAILED, MQRC_API_EXIT_INIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Vyčistit prostředí
e3	*Před* dokončením funkce MQ_CONNX_EXIT 1 postupujte takto: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Funkce Drive MQ_TERM_EXIT 2. Vyčistit prostředí 3. Nezdařilo se volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka MQ_TERM_EXIT, je-li požadována 3. Vyčistit prostředí, je-li požadováno
e4	*Před* dokončením funkce MQ_CONNX_EXIT 2 je: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_CONNX_EXIT 1 2. Funkce Drive MQ_TERM_EXIT 3. Vyčistit prostředí 4. Nezdařilo se volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka *po* funkci MQ_CONNX_EXIT 1, pokud není ukončení potlačeno 3. Jednotka MQ_TERM_EXIT, je-li požadována 4. Vyčistit prostředí, je-li požadováno

Tabulka 838. Výstupní chyby rozhraní API a odpovídající akce, které je třeba provést (pokračování)

Err Pt	Popis	Akce
e5	Volání MQCONN selhává.	1. Průchod MQCONN CompCode a důvod 2. Jednotka *po* funkci MQ_CONNX_EXIT 2, je-li *před* MQ_CONNX_EXIT 2 úspěšná a uživatelská procedura není potlačena 3. Jednotka *po* funkci MQ_CONNX_EXIT 1, je-li hodnota *před* MQ_CONNX_EXIT 1 úspěšná a uživatelská procedura není potlačena 4. Funkce Drive MQ_TERM_EXIT 5. Vyčistit prostředí
e6	*Po* dokončení funkce MQ_CONNX_EXIT 2 s: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_CONNX_EXIT 1 2. Úplné volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka *po* funkci MQ_CONNX_EXIT 1, je-li požadována
e7	*Po* dokončení funkce MQ_CONNX_EXIT 1 s: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED bylo dokončeno volání MQCONN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * se chová jako hodnoty MQXCC_ * a MQXR2_ * [1]
e8	*Před* funkcí MQ_OPEN_EXIT 1 je dokončena: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED bylo dokončeno volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * se chová jako hodnoty MQXCC_ * a MQXR2_ * [1]
e9	*Před* dokončením funkce MQ_OPEN_EXIT 2 postupujte takto: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_OPEN_EXIT 1 2. Dokončení volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * se chová jako hodnoty MQXCC_ * a MQXR2_ * [1]
e10	Volání MQOPEN selhává	1. Předat MQOPEN CompCode a příčinu 2. Jednotka *po* funkci MQ_OPEN_EXIT 2, není-li tato uživatelská procedura potlačena 3. Jednotka *po* funkci MQ_OPEN_EXIT 1, pokud není potlačena uživatelská procedura a nejsou-li potlačeny zřetězené uživatelské procedury

Tabulka 838. Výstupní chyby rozhraní API a odpovídající akce, které je třeba provést (pokračování)		
Err Pt	**Popis**	**Akce**
e11	*Po* dokončení funkce MQ_OPEN_EXIT 2 s: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_OPEN_EXIT 1 2. Dokončení volání MQOPEN s MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Jednotka *po* funkci MQ_OPEN_EXIT 1, pokud není ukončení potlačeno
e25	*Po* dokončení funkce MQ_DISC_EXIT 2 s: • SELHÁNÍ MQXCC_FAILED • MQXCC_*	• Pro MQXCC_FAILED: 1. Jednotka *po* funkci MQ_DISC_EXIT 1 2. Funkce Drive MQ_TERM_EXIT 3. Vyčistit prostředí pro provádění uživatelské procedury 4. Úplné volání MQDISC s funkcí MQCC_FAILED, MQRC_API_EXIT_ERROR • Pro MQXCC_ * 1. Jednat jako hodnoty MQXCC_ * a MQXR2_ * [1] 2. Funkce Drive MQ_TERM_EXIT 3. Vyčistit prostředí pro provádění uživatelské procedury

Poznámka:

1. Hodnoty položek MQXCC_ * a MQXR2_* a příslušné akce jsou definovány v tématu Jak správce front zpracovává výstupní funkce.

Pole ExitResponse byla nesprávně nastavena.

Toto téma poskytuje informace o tom, co by se stalo, když je pole ExitResponse nastaveno na cokoliv, ale na podporované hodnoty.

Je-li pole ExitResponse nastaveno na jinou hodnotu než jednu z podporovaných hodnot, platí následující akce:

- Pro funkci uživatelské procedury rozhraní API MQCONN nebo MQDISC pro *before* :
  - Hodnota ExitResponse2 je ignorována.
  - Žádné další funkce ukončení *před* v řetězci uživatelských procedur (je-li nějaké) jsou vyvolány; samotné volání API se nevydá.
  - Pro všechny *dřívější* uživatelské procedury, které byly úspěšně volány, jsou uživatelské procedury *po* volány v opačném pořadí.
  - Je-li zaregistrováno, funkce ukončení ukončení pro tyto funkce *před* MQCONN nebo MQDISC v řetězci, které byly úspěšně vyvolány, jsou řízeny k vyčištění po těchto ukončovacích funkcích.
  - Volání MQCONN nebo MQDISC selže s chybou MQRC_API_EXIT_ERROR.
- Pro uživatelskou proceduru rozhraní API produktu *před* IBM MQ jinou než MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Žádné další funkce *před* nebo *po* nebudou vyvolány funkce převodu dat v řetězci uživatelské procedury (pokud existují).

  - Pro všechny *dřívější* uživatelské procedury, které byly úspěšně volány, jsou uživatelské procedury *po* volány v opačném pořadí.
  - Samotné volání rozhraní API produktu IBM MQ není vydáno.
  - Volání rozhraní API produktu IBM MQ selhává s chybou MQRC_API_EXIT_ERROR.
- Pro funkci uživatelské procedury rozhraní API MQCONN nebo MQDISC pro *after* :
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající uživatelské funkce, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.
  - Je-li zaregistrováno, funkce ukončení ukončení pro tyto funkce *before* nebo *after* MQCONN nebo MQDISC v řetězci, které byly úspěšně vyvolány, jsou řízeny k vyčištění po ukončení.
  - CompCode závažnějšího z funkcí MQCC_WARNING a CompCode vrácený procedurou je vrácen do aplikace.
  - Příčinu MQRC_API_EXIT_ERROR je vrácen do aplikace.
  - Volání rozhraní API produktu IBM MQ bylo úspěšně vydáno.
- Pro funkci ukončení volání *po* ukončení volání rozhraní API produktu IBM MQ jinou než MQCONN nebo MQDISC:
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající uživatelské funkce, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.
  - CompCode závažnějšího z funkcí MQCC_WARNING a CompCode vrácený procedurou je vrácen do aplikace.
  - Příčinu MQRC_API_EXIT_ERROR je vrácen do aplikace.
  - Volání rozhraní API produktu IBM MQ bylo úspěšně vydáno.
- Pro *před* převodem dat na získání uživatelské procedury:
  - Hodnota ExitResponse2 je ignorována.
  - Zbývající uživatelské funkce, které byly úspěšně volány před voláním rozhraní API, jsou volány v opačném pořadí.
  - Zpráva se nekonvertuje a do aplikace se vrátí nekonvertované zprávy.
  - CompCode závažnějšího z funkcí MQCC_WARNING a CompCode vrácený procedurou je vrácen do aplikace.
  - Příčinu MQRC_API_EXIT_ERROR je vrácen do aplikace.
  - Volání rozhraní API produktu IBM MQ bylo úspěšně vydáno.

  Poznámka: Protože je chyba při ukončení, je lepší vrátit MQRC_API_EXIT_ERROR než vrátit MQRC_NOT_CONVERTED.

Pokud funkce uživatelské procedury nastaví pole ExitResponse2 na jinou hodnotu než jednu z podporovaných hodnot, bude místo toho použita hodnota MQXR2_DEFAULT_CONTINUATION .

Referenční informace o rozhraní instalovatelných služeb

Tato kolekce témat obsahuje referenční informace pro instalovatelné služby.

Funkce a datové typy jsou vypsány v abecedním pořadí v rámci skupiny pro každý typ služby.

Související úlohy

Rozšíření zařízení správce front

ALW Konfigurace instalovatelných služeb

Související odkazy

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 IBMi

Tyto informace vám pomohou pochopit referenční informace o instalovatelných službách pro produkt IBM i.

Způsob zobrazení funkcí

Jak jsou dokumentovány funkce instalovatelné služby.

Pro každou funkci existuje popis, včetně identifikátoru funkce (pro MQZEP).

Parametry *parameters* jsou uvedeny v pořadí, v jakém se musí vyskytnout. Všechny musí být přítomné.

Každý název parametru je následován příslušným datovým typem. Jedná se o elementární datové typy popsané v publikaci "Elementární datové typy" na stránce 235.

Vyvolání jazyka C je také poskytnuto, po popisu parametrů.

MQZ_AUTHENTICATE_USER-Ověřit uživatele

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_5 a je vyvolána správcem front k ověření uživatele nebo k nastavení polí kontextu identity. Je vyvolán, když je vytvořen kontext uživatelské aplikace IBM MQ .

Kontext aplikace se zavádí během volání connect v místě, kde je inicializován kontext uživatele aplikace, a v každém okamžiku, kdy se změní kontext uživatele aplikace. Při každém navázání spojení se informace o uživatelském kontextu aplikace znovu získávají v poli *IdentityContext* .

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_AUTHENTICATE_USER.

Syntaxe

MQZ_AUTHENTICATE_USER (*QMgrName* , *SecurityParms* , *ApplicationContext* , *IdentityContext* , *CorrelationPtr* , *ComponentData* , *Pokračování* , *CompCode* , *Důvod*)

Parametry

QMgrName

Typ: MQCHAR48 -Vstup

Název správce front. Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

SecurityParms

Typ: MQCSP-vstup

Parametry zabezpečení. Data týkající se ID uživatele, hesla a typu ověřování. Je-li atribut AuthenticationType struktury MQCSP zadán jako MQCSP_AUTH_USER_ID_AND_PWD, porovnání ID uživatele a hesla se porovná s ekvivalentnými poli v parametru IdentityContext (MQZIC), aby bylo možné určit, zda se shodují. Další informace viz téma "MQCSP-parametry zabezpečení" na stránce 336.

Během volání MQI MQCONN tento parametr obsahuje hodnotu Null nebo výchozí hodnoty.

ApplicationContext

Typ: MQZAC-vstup

Kontext aplikace. Data týkající se volající aplikace. Podrobné informace naleznete v tématu MQZAC-Aplikační kontext .

Při každém volání MQCONN nebo MQCONNX MQI se znovu získá informace o kontextu uživatele v rámci struktury MQZAC.

IdentityContext
Typ: MQZIC-input/output

Kontext identity. Při vstupu do funkce authenticate user tento kontext identifikuje aktuální kontext identity. Funkce authenticate user může tuto změnu změnit, v tom okamžiku správce front přijme nový kontext identity. Další informace o struktuře MQZIC naleznete v tématu Kontext MQZIC-Identity .

CorrelationPtr
Typ: MQPTR-výstup

Ukazatel korelace. Určuje adresu jakýchkoli korelačních dat. Tento ukazatel je následně předán dalším voláním OAM.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Příznak pokračování. Můžete určit následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

Vyvolání jazyka C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                   IdentityContext, &CorrelationPtr, ComponentData,
                   &Continuation, &CompCode, &Reason);
```

Deklarujte parametry předávané do služby 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-Oprávnění

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_1 a je spuštěna správcem front za účelem ověření, zda má entita oprávnění k provedení určité 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli 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, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

Není nezbytně nutné, aby tato entita byla známa podkladové službě zabezpečení. Není-li známo, použijí se pro kontrolu autorizace speciální skupiny **nobody** (ke které jsou všechny entity považovány za náležící). Prázdný název je platný a lze jej použít tímto způsobem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený hodnotou EntityName. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Ředitelé.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, ke kterému je přístup požadován. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění
Typ: MQLONG-vstup

Oprávnění ke kontrole. Je-li zkontrolováno jedno ověření, toto pole se rovná odpovídající operaci autorizace (MQZAO_ * konstanta). Pokud je ověřováno více než jedno ověření, je to bitové OR z odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující autorizace:

MQZAO_PŘIPOJENÍ
Schopnost použít volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadání volby MQGMO_BROWSE_FIRST, MQGMOROWS_MSG_UNDER_CURSOR nebo MQGMOROWSE_NEXT, které mají být zadány při volání MQGET.

MQZAO_VSTUP
Řediteli. Schopnost použít volání MQGET se vstupní volbou.

To umožňuje určení volby MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF, které mají být zadány při volání MQOPEN.

MQZAO_VÝSTUP
Schopnost použít volání MQPUT.

To umožňuje, aby byla volba MQOO_OUTPUT zadána v rámci volání MQOPEN.

MQZAO_DOTÁZAT SE
Schopnost použít volání MQINQ.

To umožňuje, aby byla volba MQOO_INQUIRE uvedená v rámci volání MQOPEN.

MQZAO_SADA
Schopnost použít volání MQSET.

To umožňuje, aby byla volba MQOO_SET zadána při volání MQOPEN.

KONTEXT MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje určení volby MQOO_PASS_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_IDENTITY_CONTEXT, které mají být zadány v rámci volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje určení volby MQOO_PASS_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

KONTEXT MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje určení volby MQOO_SET_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_IDENTITY_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

FUNKCE MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje určení volby MQOO_SET_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost použít alternativní oprávnění uživatele.

To umožňuje zadání volby MQOO_ALTERNATE_USER_AUTHORITY v rámci volání MQOPEN a volby MQPMO_ALTERNATE_USER_AUTHORITY, které mají být zadány ve volání MQPUT1 .

MQZA_ALL_MQI
Všechny autorizace MQI.

To povolí všechny autorizace.

Na administraci správce front se vztahují následující autorizace:

VYTVOŘIT_VYTVOŘIT_MQZAO_
Schopnost vytvořit objekty určitého typu.

MQZAO_DELETE
Schopnost odstranit uvedený objekt.

MQZAO_ZOBRAZENÍ
Schopnost zobrazit atributy zadaného objektu.

ZMĚNA MQZAO_CHANGE
Schopnost změnit atributy zadaného objektu.

MQZAO_CLEAR
Schopnost vymazat všechny zprávy z uvedené fronty.

MQZAO_AUTORIZOVAT
Schopnost autorizovat jiné uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spustit nebo zastavit listener, službu nebo objekt kanálu jiného typu než klienta a schopnost testovat spojení s objektem kanálu, který není typu klienta.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu u objektu neklientského kanálu.

MQZAODE_ALL_ADMIN
Schopnost nastavit kontext identity.

Všechny autorizace administrace, jiné než MQZAO_CREATE.

Pro použití rozhraní MQI a pro administraci správce front platí následující autorizace:

MQZAO_VŠE
Všechny autorizace, jiné než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

Pokud volání komponenty selže (to znamená, že *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují nějaké.

Je-li volání úspěšné (to znamená, že *CompCode* vrátí MQCC_OK), nevolají žádné další komponenty bez ohledu na nastavení parametru *Continuation* .

Pokud se volání nezdaří a parametr *Continuation* je MQZCI_STOP, pak se nevolají žádné další komponenty a vrátí se chyba správci front. Komponenty nemají žádné informace o předchozích voláních, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 -Kontrola oprávnění (rozšířená)

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front za účelem ověření, zda má entita oprávnění k provedení určité 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 argumentem **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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData

Typ: MQZED-vstup

Data entity. Data týkající se entity s autorizací k objektu, který má být zkontrolován. Podrobnosti viz “MQZED-deskriptor entity” na stránce 1664.

Není nezbytně nutné, aby tato entita byla známa podkladové službě zabezpečení. Není-li známo, použijí se pro kontrolu autorizace speciální skupiny **nobody** (ke které jsou všechny entity považovány za náležící). Prázdný název je platný a lze jej použít tímto způsobem.

EntityType

Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, ke kterému je přístup požadován. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Oprávnění ke kontrole. Je-li zkontrolováno jedno ověření, toto pole se rovná odpovídající operaci autorizace (MQZAO_ * konstanta). Pokud je ověřováno více než jedno ověření, je to bitové OR z odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující autorizace:

MQZAO_PŘIPOJENÍ
Schopnost použít volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadání volby MQGMO_BROWSE_FIRST, MQGMOROWS_MSG_UNDER_CURSOR nebo MQGMOROWSE_NEXT, které mají být zadány při volání MQGET.

MQZAO_VSTUP
Řediteli. Schopnost použít volání MQGET se vstupní volbou.

To umožňuje určení volby MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF, které mají být zadány při volání MQOPEN.

MQZAO_VÝSTUP
Schopnost použít volání MQPUT.

To umožňuje, aby byla volba MQOO_OUTPUT zadána v rámci volání MQOPEN.

MQZAO_DOTÁZAT SE
Schopnost použít volání MQINQ.

To umožňuje, aby byla volba MQOO_INQUIRE uvedená v rámci volání MQOPEN.

MQZAO_SADA
Schopnost použít volání MQSET.

To umožňuje, aby byla volba MQOO_SET zadána při volání MQOPEN.

KONTEXT MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje určení volby MQOO_PASS_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_IDENTITY_CONTEXT, které mají být zadány v rámci volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje určení volby MQOO_PASS_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

KONTEXT MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje určení volby MQOO_SET_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_IDENTITY_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

FUNKCE MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje určení volby MQOO_SET_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost použít alternativní oprávnění uživatele.

To umožňuje zadání volby MQOO_ALTERNATE_USER_AUTHORITY v rámci volání MQOPEN a volby MQPMO_ALTERNATE_USER_AUTHORITY, které mají být zadány ve volání MQPUT1 .

MQZA_ALL_MQI
Všechny autorizace MQI.

To povolí všechny autorizace.

Na administraci správce front se vztahují následující autorizace:

VYTVOŘIT_VYTVOŘIT_MQZAO_
Schopnost vytvořit objekty určitého typu.

MQZAO_DELETE
Schopnost odstranit uvedený objekt.

MQZAO_ZOBRAZENÍ
Schopnost zobrazit atributy zadaného objektu.

ZMĚNA MQZAO_CHANGE
Schopnost změnit atributy zadaného objektu.

MQZAO_CLEAR
Schopnost vymazat všechny zprávy z uvedené fronty.

MQZAO_AUTORIZOVAT
Schopnost autorizovat jiné uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spustit nebo zastavit listener, službu nebo objekt kanálu jiného typu než klienta a schopnost testovat spojení s objektem kanálu, který není typu klienta.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu u objektu neklientského kanálu.

MQZAODE_ALL_ADMIN
Schopnost nastavit kontext identity.

Všechny autorizace administrace, jiné než MQZAO_CREATE.

Pro použití rozhraní MQI a pro administraci správce front platí následující autorizace:

MQZAO_VŠE
Všechny autorizace, jiné než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 pomocí komponenty služby autorizace MQZAS_VERSION_6 a je vyvolána správcem front za účelem určení, 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, která má být zkontrolována. Další informace viz “MQZED-deskriptor entity” na stránce 1664.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený hodnotou EntityData. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ComponentData
Typ: MQBYTExComponentDataLength -vstupní/výstupní

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

Pokud volání komponenty selže (to znamená, že *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují nějaké.

Je-li volání úspěšné (to znamená, že *CompCode* vrátí MQCC_OK), nevolají žádné další komponenty bez ohledu na nastavení parametru *Continuation* .

Pokud se volání nezdaří a parametr *Continuation* je MQZCI_STOP, pak se nevolají žádné další komponenty a vrátí se chyba správci front. Komponenty nemají žádné informace o předchozích voláních, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584, X'A18') Tento uživatel není privilegovaným ID uživatele.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Kopírovat všechny

Tato funkce je poskytována komponentou autorizační služby. Správce front je spuštěn pro kopírování všech autorizací, které jsou aktuálně platné pro referenční objekt k jinému 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

Název RefObject
Typ: MQCHAR48 -Vstup

Název referenčního objektu. Název referenčního objektu, autorizace, které mají být kopírovány. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který mají být nastaveny přístupy. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený pomocí *RefObjectName* a *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

ComponentData
Typ: MQBYTExComponentDataLength -vstupní/výstupní

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

OBJEKT MQRC_UNKNOWN_REF_OBJECT
(2294, X'8F6') Referenční objekt není znám.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-

Tato funkce je poskytována komponentou autorizační služba a je spuštěna správcem front k odstranění všech autorizací přidružených k uvedenému 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který se mají odstranit přístupy. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Vyčíslení dat oprávnění

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_4 a je správcem front opakovaně spouštěn, aby načetl všechna data oprávnění, která odpovídají kritériím výběru uvedeným v 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

StartEnumeration
Typ: MQLONG-vstup

Příznak označující, zda volání může začít s výčtem. Označuje, zda může volání zahájit výčet dat oprávnění, nebo může pokračovat ve výčtu dat oprávnění spuštěných předchozím voláním MQZ_ENUMERATE_AUTHORITY_DATA. Hodnota je jedna z následujících hodnot:

SPUŠTĚNÍ MQZSE_START
Začátek výčtu. Volání je spuštěno s touto hodnotou pro spuštění výčtu dat oprávnění. Argument **Filter** udává kritéria výběru, která se mají použít při výběru dat oprávnění vrácených tímto a po sobě jdoucími voláními.

MQZ_CONTINUE
Pokračujte ve výčtu. Volání se spustí s touto hodnotou, aby bylo možné pokračovat ve výčtu dat oprávnění. Parametr **Filter** je v tomto případě ignorován a lze jej zadat jako ukazatel Null (výběrová kritéria jsou určována parametrem **Filter** zadaným voláním, který byl nastaven parametrem *StartEnumeration* na hodnotu MQZSE_START).

Filtr
Typ: MQZAD-vstup

Filtrovat. Je-li *StartEnumeration* MQZSE_START, *Filter* uvádí kritéria výběru, která se mají použít k výběru dat oprávnění, která se mají vrátit. Je-li *Filter* ukazatel Null, nejsou použita žádná kritéria výběru, to znamená, že jsou vrácena všechna data oprávnění. Podrobnosti o kritériích výběru, která lze použít, viz “MQZAD-data oprávnění” na stránce 1661 .

Je-li *StartEnumeration* MQZSE_CONTINUE, *Filter* je ignorován a lze jej zadat jako ukazatel null.

Délka AuthorityBufferDélka
Typ: MQLONG-vstup

Délka *AuthorityBuffer*. Toto je délka v bajtech parametru **AuthorityBuffer** . Vyrovnávací paměť oprávnění musí být dostatečně velká, aby pojmula data, která se mají vrátit.

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 a nejdelší název entity a nejdelší definovaný název domény.

Poznámka: Poznámka: Tento parametr je definován jako MQZAD, protože MQZAD se vždy vyskytuje na začátku vyrovnávací paměti. Je-li však vyrovnávací paměť deklarována jako vlastnost MQZAD, bude vyrovnávací paměť příliš malá-musí být větší než hodnota MQZAD, aby mohla být pojato názvy MQZAD, MQZED, plus entity a názvy domén.

Délka AuthorityData
Typ: MQLONG-výstup

Délka dat vrácených v *AuthorityBuffer*. Je-li vyrovnávací paměť oprávnění příliš malá, je hodnota *AuthorityDataLength* nastavena na délku požadované vyrovnávací paměti a volání vrátí kód dokončení MQCC_FAILED a kód příčiny MQRC_BUFFER_LENGTH_ERROR.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_ENUMERATE_AUTHORITY_DATA má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_NO_DATA_AVAILABLE
(2379, X'94B') Nejsou k dispozici žádná data.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 pomocí komponenty služby autorizace MQZAS_VERSION_5 a je spuštěna správcem front k uvolnění přidruženého přiděleného prostředku.

Je spuštěn, když byla dokončena aplikace pod všemi kontexty uživatele, 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

FreeParms
Typ: MQZFP-input

Volné parametry. Struktura obsahující data související s prostředkem, který má být uvolněn. Podrobnosti viz "MQZFP-volné parametry" na stránce 1666.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Příznak pokračování. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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_GETAUTHORITY-Získání oprávnění

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_1 a je spuštěna správcem front k načtení oprávnění, které má entita pro přístup k uvedenému objektu, včetně (je-li entita hlavní) oprávnění vlastněná skupinami, ve kterých 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli 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, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, ke kterému se má získat přístup. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_GET_AUTHORITY má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 -Získání oprávnění (rozšířené)

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front k načtení autority, kterou má entita pro přístup k uvedenému 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 argumentem **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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, pro kterou má být získána autorizace k objektu. Podrobnosti viz “MQZED-deskriptor entity” na stránce 1664.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který má být získáno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 autorizační služby MQZAS_VERSION_1 a je spuštěna správcem front k načtení oprávnění, které má entita pro přístup k uvedenému objektu, včetně (je-li entita hlavní) oprávnění vlastněná skupinami, ve kterých je činitel členem. Oprávnění z generických profilů jsou zahrnuta do vrácené sady oprávnění.

V systému AIX and Linuxpro vestavěného správce oprávnění objektu IBM MQ (OAM) je navrácené oprávnění, které je vlastněny 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityName
Typ: MQCHAR12 -Vstup

Název entity. Název entity, pro kterou má být získán přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který má být získáno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_GET_AUTHORITY má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 -Získání explicitního oprávnění (rozšířené)

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front, aby načetl oprávnění, které má pojmenovaná skupina pro přístup k uvedenému objektu (ale bez dalšího oprávnění skupiny **nikdo**), nebo oprávnění, které má primární skupina uvedeného činitele pro přístup k uvedenému 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 argumentem **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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, jejíž oprávnění k objektu má být načteno. Podrobnosti viz “MQZED-deskriptor entity” na stránce 1664.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který má být získáno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina

Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 autorizační služby

Tato funkce je poskytována komponentou autorizační služba a je spuštěna správcem front během konfigurace komponenty. Očekává se, že zavolá 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 manipulátor představuje inicializaci konkrétní komponenty. Tuto komponentu je třeba použít při volání správce front s funkcí MQZEP.

Volby
Typ: MQLONG-vstup

Volby inicializace. Musí se jednat o jednu 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

Délka ComponentData
Typ: MQLONG-vstup

Délka dat komponenty. Délka (v bajtech) oblasti *ComponentData* . Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Před voláním primární inicializační funkce komponenty je inicializováno na všechny nuly. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Verze
Typ: MQLONG-input/output

Číslo verze. Při vstupu do inicializační funkce toto identifikuje nejvyšší číslo verze, které správce front podporuje. Funkce inicializace musí v případě potřeby tuto verzi změnit na verzi rozhraní, které podporuje. Pokud při vrácení správce front nepodporuje verzi vrácenou komponentou, volá komponentu MQZ_TERM_AUTHORITY a nevyužívá další použití této komponenty.

Jsou podporovány 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 se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

INICIALIZACE MQRC_INITIALIZATION_SELHALA
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Dotaz na službu autorizace

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_5 a je spuštěna správcem front za účelem dotazování na podporovanou funkčnost.

Je-li použito více komponent služeb, jsou komponenty služeb volány v opačném pořadí, než jsou instalovány v pořadí, ve kterém byly instalovány.

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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

SelectorCount
Typ: MQLONG-vstup

Počet selektorů. Počet selektorů dodávaných s parametrem **Selectors** .

Hodnota musí být v rozsahu 0 až 256.

Selektory.
Typ: MQLONGxSelectorCount-vstup

Pole selektorů. Každý selektor identifikuje požadovaný atribut a musí být jeden z následujících:

- MQIACF_INTERFACE_VERSION (celé číslo)
- MQIACF_USER_ID_SUPPORT (celé číslo)
- MQCACF_SERVICE_COMPONENT (znak)

Selektory mohou být zadány v libovolném pořadí. Počet selektorů v poli je indikován parametrem **SelectorCount** .

Celočíselné atributy určené selektory se vracejí do parametru **IntAttrs** ve stejném pořadí, v jakém se objevují v produktu *Selectors*.

Atributy znaků určené selektory jsou vráceny v parametru **CharAttrs** ve stejném pořadí, v jakém se objevují ve *Selectors*.

IntAttrCount
Typ: MQLONG-vstup

Počet celočíselných atributů dodaných v parametru IntAttrs .

Hodnota musí být v rozsahu 0 až 256.

IntAttrs
Typ: MQLONG x IntAttrCount-output

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. Nejsou-li požadovány žádné znakové atributy, nula je platná hodnota.

CharAttrs
Typ: MQLONG x CharAttrCount-output

Vyrovnávací paměť pro atributy znaků. Vyrovnávací paměť obsahující atributy znaků, zřetězená dohromady. Atributy znaku se vrací ve stejném pořadí jako odpovídající selektory znaku v poli *Selectors* .

Délka vyrovnávací paměti je dána parametrem Počet 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 indikován parametrem **SelectorCount** . Každá hodnota v poli se vztahuje k selektoru z odpovídající pozice v poli Selektory. Každá hodnota je jedna z následujících možností:

FUNKCE MQZSL_RETURNED
Byl vrácen atribut požadovaný příslušným selektorem v parametru **Selectors** .

MQZSL_NOT_RETURNED
Atribut požadovaný odpovídajícím selektorem v parametru **Selectors** nebyl vrácen.

Pole je inicializováno se všemi hodnotami jako *MQZSL_NOT_RETURNED*. Když komponenta autorizační služby vrátí atribut, nastaví příslušnou hodnotu v poli na *MQZSL_NOT_RETURNED*. To umožňuje ostatním komponentám autorizační služby, ke kterým je vytvořeno dotazová volání, za účelem zjištění, které atributy již byly vráceny.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Má-li parametr *CompCode* hodnotu MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
Nedostatek prostoru pro atributy znaků.

POČ_DO_LOKÁLNÍ_FRONTY_MQRC_INT_TOO_SMALL
Nedostatek prostoru pro celočíselné atributy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA MQRC_SELECTOR_COUNT_ERROR
Počet selektorů není platný.

CHYBA MQRC_SELECTOR_ERROR
Selektor atributu není platný.

MQRC_SELECTOR_LIMIT_EXCEEDED
Je zadáno příliš mnoho selektorů.

CHYBA_MQRC_INT_ATTR_COUNT_ERROR
Počet celočíselných atributů je neplatný.

CHYBA POLE MQRC_INT_ATTRS_ARRAY_ERROR
Pole celočíselné atributy není platné.

MQRC_CHAR_ATTR_LENGTH_ERROR
Počet znakových atributů je neplatný.

CHYBA MQRC_CHAR_ATTRS_ERROR
Řetězec znaků znaků je neplatný.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Aktualizovat všechny autorizace

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_3 a je vyvolána správcem front k aktualizaci seznamu oprávnění interně držených 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Má-li parametr *CompCode* hodnotu MQCC_WARNING:

CHYBA SLUŽBY MQRC_SERVICE_
(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-Nastavení oprávnění

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_1 a je spuštěna správcem front, aby nastavil oprávnění, které má entita pro přístup k uvedenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechny existující oprávnění. Chcete-li zachovat existující oprávnění, je třeba je nastavit znovu s touto funkcí.

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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityName
Typ: MQCHAR12 -Vstup

Název entity. Název entity, pro kterou má být získán přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityName*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, ke kterému je přístup požadován. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Je-li nastaveno jedno oprávnění, je toto pole rovno odpovídající operaci autorizace (MQZAO_ * konstanta). Je-li nastavováno více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentDatarname>
Typ: MQBYTExComponentDataLength -vstupní/výstupní

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_GET_AUTHORITY má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 -Nastavení oprávnění (rozšířené)

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_2 a je spuštěna správcem front pro nastavení oprávnění, které má entita pro přístup k uvedenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechny existující oprávnění. Chcete-li zachovat existující oprávnění, je třeba je nastavit znovu s touto funkcí.

MQZ_SET_AUTHORITY_2 je jako MQZ_SET_AUTHORITY, ale s parametrem **EntityName** nahrazeným argumentem **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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, jejíž oprávnění k objektu má být nastaveno. Podrobnosti viz “MQZED-deskriptor entity” na stránce 1664.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený parametrem *EntityData*. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName
Typ: MQCHAR48 -Vstup

Název objektu. Název objektu, pro který má být nastaveno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je stejný jako *QMgrName*.

ObjectType
Typ: MQLONG-vstup

Typ objektu. Typ entity určený parametrem *ObjectName*. Musí se jednat o jednu z následujících hodnot:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

MQOT_TOPIC
.

Oprávnění
Typ: MQLONG-vstup

Orgán účetní jednotky. Je-li nastaveno jedno oprávnění, je toto pole rovno odpovídající operaci autorizace (MQZAO_ * konstanta). Je-li nastavováno více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Ukončení autorizační služby

Tato funkce je poskytována komponentou autorizační služby a je spuštěna správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést jakékoli 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 komponentu, která se ukončuje. Tuto komponentu je třeba použít při volání správce front s funkcí MQZEP.

Volby
Typ: MQLONG-vstup

Volby ukončení. Musí se jednat o jednu z následujících hodnot:

MQZTO_PRIMÁRNÍ
Primární ukončení.

MQZ_SEKUNDÁRNÍ
Sekundární ukončení.

QMgrName
Typ: MQCHAR48 -Vstup

Název správce front. Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_AUTHORITY.

Po dokončení volání MQZ_TERM_AUTHORITY zahodí správce front tato data.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

SELHÁNÍ MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení se nezdařilo z nedefinované příčiny.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Odstranit název

Tato funkce je poskytována komponentou služby názvů a je spuštěna správcem front k 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

QName
Typ: MQCHAR48 -Vstup

Název fronty. Název fronty, pro kterou má být položka odstraněna. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předávána správcem front v parametru Length ComponentDatav rámci volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Musí se jednat o jednu z následujících hodnot:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

MQZCI_STOP
Nepokračovat s další komponentou.

Pro příkaz **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 se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Varování (částečné dokončení).

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Má-li parametr *CompCode* hodnotu MQCC_WARNING:

NÁZEV MQRC_UNKNOWN_NAME
(2288, X'8F0') Název fronty nebyl nalezen.

Poznámka: Možná nebude možné vrátit tento kód, pokud základní služba odpoví s úspěchem pro tento případ.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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, že zavolá 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 manipulátor představuje inicializaci konkrétní komponenty. Tuto komponentu je třeba použít při volání správce front s funkcí MQZEP.

Volby
Typ: MQLONG-vstup

Volby inicializace. Musí se jednat o jednu 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

Délka ComponentData
Typ: MQLONG-vstup

Délka dat komponenty. Délka (v bajtech) oblasti *ComponentData* . Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Před voláním primární inicializační funkce komponenty je inicializováno na všechny nuly. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Verze
Typ: MQLONG-input/output

Číslo verze. Při vstupu do inicializační funkce toto identifikuje nejvyšší číslo verze, které správce front podporuje. Funkce inicializace musí v případě potřeby tuto verzi změnit na verzi rozhraní, které podporuje. Pokud při návratu správce front nepodporuje verzi vrácenou komponentou, volá funkci MQZ_TERM_NAME a nevyužívá další použití této komponenty.

Jsou podporovány následující hodnoty:

MQZAS_VERSION_1
Verze 1.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

INICIALIZACE MQRC_INITIALIZATION_SELHALA
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 prostřednictvím komponenty služby názvů a je spuštěna správcem front za účelem vložení položky pro určenou frontu obsahující název správce front, který je vlastníkem fronty. Je-li fronta již ve službě definována, volání selže.

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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

QName
Typ: MQCHAR48 -Vstup

Název fronty. Název fronty, pro kterou má být vložena položka. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název ResolvedQMgr
Typ: MQCHAR48 -Vstup

Vyřešený název správce front. Název správce front, do kterého je fronta rozpoznána. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

ComponentData
Typ: MQBYTE ×ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-input/output

Indikátor pokračování nastavený komponentou. U objektu MQZ_INSERT_NAME se správce front nepokusí spustit jinou komponentu, ať už je vracena v rámci parametru **Continuation** .

Jsou podporovány následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

MQRC_Q_ALREADY_EXISTS
(2290, X'8F2') Objekt fronty již existuje.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 vyhledání

Tato funkce je poskytována prostřednictvím komponenty 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

QName
Typ: MQCHAR48 -Vstup

Název fronty. Název fronty, pro kterou má být položka rozlišena. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

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 směrem doprava vyplněn mezerami až do celé délky parametru; jméno nesmí být ukončeno znakem null nebo obsahovat úvodní nebo vložené mezery.

ComponentData
Typ: MQBYTExComponentDataLength -vstupní/výstupní

Data komponent. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Pro MQZ_LOOKUP_NAME určuje správce front, zda má být spuštěna jiná komponenta služby názvů, takto:

- Pokud je *CompCode* MQCC_OK, nejsou spuštěny žádné další komponenty, jakákoli hodnota se vrátí v *Pokračování*.
- Pokud *CompCode* není MQCC_OK, je spuštěna další komponenta, pokud *Continuation* není MQZCI_STOP.

Jsou podporovány následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

MQRC_UNKNOWN_Q_NAME
(2288, X'8F0') Název fronty nebyl nalezen.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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-Ukončení služby názvů

Tato funkce je poskytována prostřednictvím komponenty služby názvů a je spuštěna správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést jakékoli 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 komponentu, která se ukončuje. Používá ji komponenta při volání správce front s funkcí MQZEP.

Volby
Typ: MQLONG-vstup

Volby ukončení. Musí se jednat o jednu z následujících hodnot:

MQZTO_PRIMÁRNÍ
Primární ukončení.

MQZ_SEKUNDÁRNÍ
Sekundární ukončení.

QMgrName
Typ: MQCHAR48 -Vstup

Název správce front. Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

ComponentData
Typ: MQBYTE x ComponentDataDélka-vstup/výstup

Data komponent. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z těchto funkcí komponent.

Data komponent jsou ve sdílené paměti přístupná pro všechny procesy.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_NAME.

Po dokončení volání MQZ_TERM_NAME správce front vyřadí tato data.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

SELHÁNÍ MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení se nezdařilo z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 pro volání MQZ_AUTHENTICATE_USER pro parametr *ApplicationContext* . Tento parametr uvádí data související s volající aplikací.

Tabulka 1 shrnuje pole ve struktuře.

Tabulka 839. Pole v MQZAC

Pole	Popis
StrucId	Identifikátor struktury
verze	Číslo verze struktury
ProcessId	Identifikátor procesu
ThreadId	Identifikátor podprocesu

Tabulka 839. Pole v MQZAC (pokračování)	
Pole	**Popis**
ApplName	Název aplikace
UserID	Identifikátor uživatele
EffectiveUserID	Efektivní identifikátor uživatele
Prostředí	Prostředí
CallerType	Typ volajícího
AuthenticationType	Typ ověřování
BindType	Typ vazby

Pole

StrucId
Typ: MQCHAR4 -Vstup

Identifikátor struktury. Hodnota je následující:

ID_STRUKTURY MQZAC_STRUCT
Identifikátor struktury kontextu aplikace.

Pro programovací jazyk C je také definován konstantní MQZAC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAC_STRUC_ID, ale je to pole znaků namísto 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 AIX and Linux toto pole uvádí skutečné ID uživatele aplikace. V poli Windows toto pole uvádí ID uživatele aplikace.

ID EffectiveUser
Typ: MQCHAR12 -Vstup

Efektivní identifikátor uživatele. V AIX and Linux toto pole uvádí efektivní ID uživatele aplikace. V systému Windows je toto pole prázdné.

Prostředí
Typ: MQLONG-vstup

Prostředí. Toto pole uvádí prostředí, ze kterého bylo volání provedeno. Pole je jedna z následujících hodnot:

MQXE_PŘÍKAZOVÝ_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 je jedna z následujících hodnot:

MQXACT_EXTERNAL
Volání je externí pro správce front.

MQXACT_INTERNAL
Volání je interní pro správce front.

AuthenticationType
Typ: MQLONG-vstup

Typ ověření. Toto pole uvádí typ ověření, které se provádí. Pole je jedna z následujících hodnot:

POČÁTEČNÍ_KONTEXT MQZATR_CONTEXT
Volání ověření je způsobeno inicializací kontextu uživatele. Tato hodnota se používá během volání MQCONN nebo MQCONNX.

KONTEXT MQZAT_CHANGE_CONTEXT
Volání ověření je způsobeno změnou kontextu uživatele. Tato hodnota se použije, když agent MCA změní kontext uživatele. Nadřízené téma: MQZAC-

BindType
Typ: MQLONG-vstup

Typ vazby. Toto pole uvádí typ vazby, která se má použít. Pole je jedna z následujících hodnot:

VAZBA MQCNO_FASTPATH_BINDING
Vazba zrychleného přístupu.

CQCNO_SHARED_BINDING
Sdílená vazba.

VAZBA MQCNO_ISOLATED_BINDING
Samostatná vazba.

Deklarace C

Deklarujte pole struktury následujícím způsobem:

```
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 v rámci 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-Vyčíslení dat oprávnění” na stránce 1623 :

- Objekt MQZAD se používá pro parametr **Filter** , který je vstupem pro volání. Tento parametr uvádí kritéria výběru, která mají být použita pro výběr dat oprávnění vrácených voláním.
- Objekt MQZAD se také používá pro parametr **AuthorityBuffer** , který je výstupem z volání. Tento parametr určuje autorizace pro jednu kombinaci názvu profilu, typu objektu a entity.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 840. 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:

ID_KONSTRUKCE_MQZAD_OBJEKTU
Identifikátor pro datovou strukturu oprávnění.

Pro programovací jazyk C je také definována konstanta MQZAD_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAD_STRUC_ID, ale je to pole znaků namísto 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 uvádí číslo verze aktuální verze:

V_AKTUÁLNÍ_VERZE MQZAD_
Aktuální verze datové struktury oprávnění.

ProfileName
Typ: MQCHAR48 -Vstup

Název profilu.

U parametru **Filter** je toto pole název profilu, pro který jsou vyžadována data oprávnění. Je-li název zcela prázdný až do konce pole nebo prvního znaku null, vrátí se data oprávnění pro všechny názvy profilů.

U parametru **AuthorityBuffer** je toto pole názvem profilu, který odpovídá zadaným kritériím výběru.

ObjectType
Typ: MQLONG-vstup

Typ objektu.

U parametru **Filter** je toto pole typ objektu, pro který jsou vyžadována data oprávnění. Je-li hodnota MQOT_ALL, je vrácena data oprávnění pro všechny typy objektů.

U parametru **AuthorityBuffer** je toto pole typ objektu, na který se použije profil identifikovaný parametrem **ProfileName** .

Hodnota je jedna z následujících možností; pro parametr **Filter** je hodnota MQOT_ALL také platná:

MQOT_AUTH_INFO
Ověřovací informace

MQOT_CHANNEL
Kanál

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta

MQOT_LISTENER
Modul listener

MQO_NAMELIST
Seznam názvů

PROCES MQOT_PROCESS
Definice procesu

MQOT_Q
Fronta

MQOT_Q_MGR
Správce front

SLUŽBA MQOT_SERVICE
Služba

Oprávnění
Typ: MQLONG-vstup

Oprávnění.

U parametru **Filter** je toto pole ignorováno.

U parametru **AuthorityBuffer** toto pole představuje oprávnění, která má entita k objektům identifikovaným pomocí **ProfileName** a **ObjectType**. Má-li entita pouze jedno oprávnění, je pole rovno odpovídající hodnotě autorizace (MQZAO_ * konstanta). Má-li entita více než jedno oprávnění, je toto pole bitové OR z odpovídajících konstant MQZAO_ *.

EntityDataPtr
Typ: PMQZED-vstup

Adresa struktury MQZED, která identifikuje entitu.

V případě parametru **Filter** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, pro kterou jsou vyžadována data oprávnění. Je-li **EntityDataPtr** ukazatel null, jsou vrácena data oprávnění pro všechny entity.

U parametru **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 vyžadována data oprávnění. Je-li hodnota MQZAET_NONE, vrátí se data oprávnění pro všechny typy entit.

U parametru **AuthorityBuffer** toto pole určuje typ entity určený strukturou MQZED, na kterou je odkazováno pomocí parametru **EntityDataPtr** .

Hodnota je jedna z následujících možností; pro parametr **Filter** je hodnota MQZAET_NONE také platná:

ČINITEL MQZAET_PRINCIPAL
Hlavní

SKUPINA 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, jak je uvedeno v poli **ProfileName** .

Kromě toho musí být zadán také jeden z následujících:

MQAUTHOPT_ENTITY_SET
Zobrazit všechny profily, které se používají k výpočtu kumulativního oprávnění, které má entita k objektu určenému argumentem **ProfileName** . Argument **ProfileName** nesmí obsahovat žádné zástupné znaky.

- Je-li uvedená entita činitelem, zobrazí se pro každého člena sady {entity, groups} nejvhodnější profil, který se vztahuje na daný objekt.
- Je-li uvedená entita skupina, zobrazí se nejvhodnější profil ze skupiny, která se vztahuje na objekt.
- Je-li zadána tato hodnota, musí být hodnoty **ProfileName**, **ObjectType**, **EntityType**a názvu entity zadané ve struktuře **EntityDataPtr** MQZED všechny neprázdné.

Pokud jste zadali parametr MQAUTHOPT_NAME_ALL_MATCHING, můžete také zadat následující hodnotu:

MQAUTHOPT_ENTITY_EXPLICIT
Zobrazí profily, které mají přesně stejný název entity, jako je název entity určený ve struktuře **EntityDataPtr** MQZED.

Deklarace C

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
  MQCHAR4   StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQCHAR48  ProfileName;    /* Profile name */
  MQLONG    ObjectType;     /* Object type */
  MQLONG    Authority;      /* Authority */
  PMQZED    EntityDataPtr;  /* Address of MQZED structure identifying an
                               entity */
```

```
  MQLONG    EntityType;     /* Entity type */
  MQAUTHOPT Options;        /* Options */
};
```

MQZED-deskriptor entity

Struktura MQZED se používá v mnoha voláních autorizační služby k určení entity, pro kterou se má ověřit autorizace.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 841. Pole v MQZED	
Pole	**Popis**
StrucId	Identifikátor struktury
verze	Verze
EntityName Ptr	Název entity
EntityDomainPtr	Ukazatel na doménu entity
SecurityId	Identifikátor zabezpečení
CorrelationPtr	Ukazatel korelace

Pole

StrucId
Typ: MQCHAR4 -Vstup

Identifikátor struktury. Hodnota je následující:

ID_STRUKTURY MQZED_STRUCT
Identifikátor struktury deskriptoru entity.

Pro programovací jazyk C je také definována konstanta MQZED_STRUC_ID_ARRAY; hodnota 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
Struktura deskriptoru entity Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZED_VERSION
Aktuální verze struktury deskriptoru entity.

EntityNamePtr
Typ: PMQCHAR-vstup

Název profilu.

Adresa názvu entity. Jedná se o ukazatel na název entity, jejíž autorizaci má být zkontrolována.

EntityDomainPtr
Typ: PMQCHAR-vstup

Adresa názvu domény entity. Jedná se o ukazatel na název domény obsahující definici entity, jejíž autorizaci má být zkontrolována.

SecurityId
Typ: MQBYTE40 -Vstup

Oprávnění.

Identifikátor zabezpečení. Jedná se o identifikátor zabezpečení, jehož autorizaci má být zkontrolována.

CorrelationPtr
Typ: MQPTR-vstup

Ukazatel korelace. To usnadňuje předávání korelačních dat mezi funkcí authenticate user a dalšími vhodnými funkcemi OAM.

Deklarace C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  PMQCHAR   EntityNamePtr;    /* Address of entity name */
  PMQCHAR   EntityDomainPtr;  /* Address of entity domain name */
  MQBYTE40  SecurityId;       /* Security identifier */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
```

MQZEP-Přidání vstupního bodu komponenty

Komponenta služby spouští tuto funkci během inicializace, aby přidal vstupní bod do vektoru vstupního bodu pro tuto komponentu služby.

Syntaxe
MQZEP (*Hconfig* , *Funkce* , *EntryPoint* , *CompCode* , *Reason*)

Parametry

Hconfig
Typ: MQHCONFIG-vstup

Popisovač konfigurace. Tento popisovač představuje komponentu, která se konfiguruje pro tuto konkrétní instalovatelnou službu. Musí být stejný jako komponenta předávaná správci front v rámci inicializace komponenty danou funkcí konfigurace komponenty.

Funkce
Typ: MQLONG-vstup

Identifikátor funkce. Platné hodnoty pro toto jsou definovány pro každou instalovatelnou službu.

Je-li MQZEP voláno více než jednou pro stejnou funkci, naposledy použité volání poskytuje vstupní bod, který se použije.

EntryPoint
Typ: PMQFUNC-vstup

Vstupní bod funkce. Jedná se o adresu vstupního bodu, který komponenta poskytuje k provedení funkce.

Hodnota NULL je platná a označuje, že funkce není poskytována touto komponentou. Pro vstupní body, které nejsou definovány pomocí MQZEP, se předpokládá hodnota NULL.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA FUNKCE MQRC_FUNCTION_ERROR
(2281, X'8E9') Identifikátor funkce není platný.

CHYBA MQRC_HCONFIG_ERROR
(2280, X'8E8') Popisovač konfigurace není platný.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 v rámci volání MQZ_FREE_USER pro parametr *FreeParms* . Tento parametr uvádí data související s prostředkem, který má být uvolněn.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 842. Pole v MQZFP

Pole	Popis
StrucId	Identifikátor struktury
verze	Verze
Vyhrazené	Vyhrazené pole
CorrelationPtr	Ukazatel korelace

Pole

StrucId
Typ: MQCHAR4 -Vstup

Identifikátor struktury. Hodnota je následující:

MQZIC_STRUCTION_ID
Identifikátor struktury kontextu identity. Pro programovací jazyk C je také definována konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků namísto řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZFP_VERSION_1
Struktura parametrů volných parametrů Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZFP_CURRENT_VERSION
Aktuální verze struktury volných parametrů.

Vyhrazené
Typ: MQBYTE8 -Vstup

Rezervované pole. Počáteční hodnota je null.

CorrelationPtr
Typ: MQPTR-vstup

Ukazatel korelace. Adresa korelačních dat souvisejících s prostředkem, který má být uvolněn.

Deklarace C

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQBYTE8   Reserved;         /* Reserved field */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
};
```

MQZIC-Kontext identity

Struktura MQZIC se používá pro 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 z aplikace, která vložila zprávu.
- Aplikace mohou používat pole *DataApplIdentityData* pro jakékoli další informace, které chtějí zahrnout o uživateli (například zašifrované heslo).

Autorizované autorizované aplikace mohou nastavit kontext identity pomocí funkce MQZ_AUTHENTICATE_USER.

Identifikátor zabezpečení systému Windows (SID) je uložen v poli *AccountingToken* , je-li vytvořena zpráva pod IBM MQ for Windows. Identifikátor SID lze použít k doplnění pole *UserIdentifier* a k ustanovení pověření uživatele.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 843. 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_STRUCTION_ID
Identifikátor struktury kontextu identity. Pro programovací jazyk C je také definována konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků namísto řetězce.

Verze
Typ: MQLONG-vstup

Číslo verze struktury. Hodnota je následující:

MQZIC_VERSION_1
Struktura kontextu identity Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZIC_AKTUÁLNÍ_VERZE
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 její formát. Další informace o poli *UserIdentifier* viz “UserIdentifier (MQCHAR12)” na stránce 454.

AccountingToken
Typ: MQBYTE32 -Vstup

Token evidence. Toto je část kontextu identity zprávy. Volba *AccountingToken* umožňuje aplikaci způsobit, že bude práce hotova jako výsledek řádně nabité zprávy. Správce front považuje tyto informace za řetězec bitů a nekontroluje jeho obsah. Další informace o poli *AccountingToken* viz “AccountingToken (MQBYTE32)” na stránce 455.

ApplIdentityData
Typ: MQCHAR32 -Vstup

Data aplikace související s identitou. Toto je část kontextu identity zprávy. ApplIdentityData jsou informace, které jsou definovány sadou aplikací, které lze použít k poskytnutí dalších informací o původu zprávy. Například by mohly být nastaveny aplikacemi, které jsou spuštěny s odpovídajícím oprávněním uživatele, aby označovaly, zda jsou data identity důvěryhodná. Další informace o datovém poli ApplIdentitynaleznete v tématu “Data ApplIdentity(MQCHAR32)” na stránce 457.

Deklarace C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQCHAR12  UserIdentifier;   /* User identifier */
  MQBYTE32  AccountingToken;  /* Accounting token */
  MQCHAR32  ApplIdentityData; /* Application data relating to identity */
};
```

IBM i Referenční informace o rozhraní instalovatelných služeb v systému IBM i

Tyto informace vám pomohou pochopit referenční informace o instalovatelných službách pro produkt IBM i.

Pro každou funkci existuje popis, včetně identifikátoru funkce (pro MQZEP).

Parametry *parameters* jsou uvedeny v pořadí, v jakém se musí vyskytnout. Všechny musí být přítomné.

Každý název parametru je následován příslušným datovým typem v závorkách. Jedná se o elementární datové typy popsané v části “Elementární datové typy” na stránce 988.

Vyvolání jazyka C je také poskytnuto, po popisu parametrů.

Související odkazy

IBM i Instalovatelné služby a komponenty v systému IBM i

ALW Instalovatelné služby a komponenty pro produkt AIX, Linux, and Windows

“Referenční informace o rozhraní instalovatelných služeb” na stránce 1606
Tato kolekce témat obsahuje referenční informace pro instalovatelné služby.

IBM i MQZEP (Přidání vstupního bodu komponenty) v systému IBM i

Tato funkce je vyvolána komponentou služby během inicializace, aby bylo možné přidat vstupní bod 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 v inicializačním volání komponenty.

Funkce (MQLONG)-vstup
Identifikátor funkce.

Platné hodnoty pro toto jsou definovány pro každou instalovatelnou službu. Pokud je funkce MQZEP volána pro stejnou funkci více než jednou, poslední volání poskytuje vstupní bod, který se používá.

EntryPoint (PMQFUNC)-vstup
Vstupní bod funkce.

Jedná se o adresu vstupního bodu, který komponenta poskytuje k provedení funkce. Hodnota NULL je platná a označuje, že funkce není poskytována touto komponentou. Hodnota 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.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA FUNKCE MQRC_FUNCTION_ERROR
(2281, X'8E9') Identifikátor funkce není platný.

CHYBA MQRC_HCONFIG_ERROR
(2280, X'8E8') Popisovač konfigurace není platný.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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) v systému IBM i

Datový typ MQHCONFIG představuje konfigurační popisovač, tj. komponentu, která je konfigurována pro konkrétní instalovatelnou službu. Manipulátor konfigurace musí být zarovnán na jeho přirozené hranici.

Aplikace musí testovat proměnné tohoto typu pouze pro rovnost.

Deklarace C

```
typedef void MQPOINTER MQHCONFIG;
```

IBM i PMQFUNC (Ukazatel na funkci) v systému IBM i

Ukazatel na funkci.

Deklarace C

```
typedef void MQPOINTER PMQFUNC;
```

IBM i MQZ_AUTHENTICATE_USER (Ověřit uživatele) v systému IBM i

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_5 . Je volán správcem front za účelem ověření uživatele, nebo k nastavení polí kontextu identity.

Je vyvolána při vytvoření kontextu uživatelské aplikace IBM MQ . K tomu dojde při volání connect v místě, kde je inicializován kontext uživatele aplikace, a v každém okamžiku, kdy se změní kontext uživatele

aplikace. Při každém navázání spojení se informace o uživatelském kontextu aplikace znovu získávají 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem. Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

SecurityParms (MQCSP)-vstup
Parametry zabezpečení.

Data týkající se ID uživatele, hesla a typu ověřování.

Během volání MQI MQCONN tento parametr obsahuje hodnotu Null nebo výchozí hodnoty.

ApplicationContext (MQZAC)-vstup
Kontext aplikace.

Data týkající se volající aplikace. Podrobnosti viz “MQZAC (kontext aplikace) v systému IBM i” na stránce 1700. Při každém volání MQCONN nebo MQCONNX MQI se znovu získá informace o kontextu uživatele v rámci struktury MQZAC.

IdentityContext (MQZIC)-vstup/výstup
Kontext identity.

Při vstupu do funkce authenticate user tento kontext identifikuje aktuální kontext identity. Funkce authenticate user může tuto změnu změnit, v tom okamžiku správce front přijme nový kontext identity. Další informace o struktuře MQZIC viz “MQZIC (kontext identity) v systému IBM i” na stránce 1707 .

CorrelationPtr (MQPTR)-výstup
Ukazatel korelace.

Určuje adresu jakýchkoli korelačních dat. Tento ukazatel je poté předán dalším voláním OAM.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a zobrazí se při příštím vyvolání jedné z těchto funkcí komponent. Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Příznak pokračování.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 (Kontrola oprávnění) v systému IBM i

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front za účelem ověření, zda má entita oprávnění k provedení určité 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem. Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

EntityName (MQCHAR12)-Vstup
Název entity.

Název entity, jejíž autorizace k objektu má být zkontrolována. Maximální délka řetězce je 12 znaků; je-li kratší, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

Není nezbytně nutné, aby tato entita byla známa podkladové službě zabezpečení. Není-li známo, použijí se pro kontrolu autorizace speciální skupiny **nikdo** (ke které jsou všechny entity považovány za náležící). 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:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, ke kterému je přístup požadován. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění (MQLONG)-vstup
Oprávnění ke kontrole.

Je-li zkontrolováno jedno ověření, toto pole se rovná odpovídající operaci autorizace (MQZAO_ * konstanta). Pokud je ověřováno více než jedno ověření, je to bitové OR z odpovídajících konstant MQZAO_ *.

Pro použití volání MQI platí následující autorizace:

MQZAO_PŘIPOJENÍ
Schopnost použít volání MQCONN.

MQZAO_BROWSE
Schopnost použít volání MQGET s volbou procházení.

To umožňuje zadání volby MQGMO_BROWSE_FIRST, MQGMOROWS_MSG_UNDER_CURSOR nebo MQGMOROWSE_NEXT, které mají být zadány při volání MQGET.

MQZAO_VSTUP
Schopnost použít volání MQGET se vstupní volbou.

To umožňuje určení volby MQOO_INPUT_SHARED, MQOO_INPUT_EXCLUSIVE nebo MQOO_INPUT_AS_Q_DEF, které mají být zadány při volání MQOPEN.

MQZAO_VÝSTUP
Schopnost použít volání MQPUT.

To umožňuje, aby byla volba MQOO_OUTPUT zadána v rámci volání MQOPEN.

MQZAO_DOTÁZAT SE
Schopnost použít volání MQINQ.

To umožňuje, aby byla volba MQOO_INQUIRE uvedená v rámci volání MQOPEN.

MQZAO_SADA
Schopnost použít volání MQSET.

To umožňuje, aby byla volba MQOO_SET zadána při volání MQOPEN.

KONTEXT MQZAO_PASS_IDENTITY_CONTEXT
Schopnost předat kontext identity.

To umožňuje určení volby MQOO_PASS_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_IDENTITY_CONTEXT, které mají být zadány v rámci volání MQPUT a MQPUT1 .

MQZAO_PASS_ALL_CONTEXT
Schopnost předat celý kontext.

To umožňuje určení volby MQOO_PASS_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_PASS_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

KONTEXT MQZAO_SET_IDENTITY_CONTEXT
Schopnost nastavit kontext identity.

To umožňuje určení volby MQOO_SET_IDENTITY_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_IDENTITY_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

FUNKCE MQZAO_SET_ALL_CONTEXT
Schopnost nastavit celý kontext.

To umožňuje určení volby MQOO_SET_ALL_CONTEXT v rámci volání MQOPEN a volby MQPMO_SET_ALL_CONTEXT, které mají být určeny v rámci volání MQPUT a MQPUT1 .

MQZAO_ALTERNATE_USER_AUTHORITY
Schopnost použít alternativní oprávnění uživatele.

To umožňuje zadání volby MQOO_ALTERNATE_USER_AUTHORITY v rámci volání MQOPEN a volby MQPMO_ALTERNATE_USER_AUTHORITY, které mají být zadány ve volání MQPUT1 .

MQZA_ALL_MQI
Všechny autorizace MQI.

To umožňuje všechny již popsané autorizace.

Na administraci správce front se vztahují následující autorizace:

VYTVOŘIT_VYTVOŘIT_MQZAO_
Schopnost vytvořit objekty určitého typu.

MQZAO_DELETE
Schopnost odstranit uvedený objekt.

MQZAO_ZOBRAZENÍ
Schopnost zobrazit atributy zadaného objektu.

ZMĚNA MQZAO_CHANGE
Schopnost změnit atributy zadaného objektu.

MQZAO_CLEAR
Schopnost vymazat všechny zprávy z uvedené fronty.

MQZAO_AUTORIZOVAT
Schopnost autorizovat jiné uživatele pro uvedený objekt.

MQZAO_CONTROL
Schopnost spustit, zastavit nebo odeslat příkaz ping na objekt kanálu jiného typu než klienta.

MQZAO_CONTROL_EXTENDED
Schopnost resetovat pořadové číslo nebo vyřešit neověřenou zprávu u objektu neklientského kanálu.

MQZAODE_ALL_ADMIN
Všechny autorizace administrace, jiné než MQZAO_CREATE.

Pro použití rozhraní MQI a pro administraci správce front platí následující autorizace:

MQZAO_VŠE
Všechny autorizace, jiné než MQZAO_CREATE.

MQZAO_NONE
Žádná oprávnění.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 pomocí komponenty služby autorizace MQZAS_VERSION_6 a je vyvolána správcem front za účelem určení, 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

EntityData
Typ: MQZED-vstup

Data entity. Data týkající se entity, která má být zkontrolována. Další informace viz "MQZED-deskriptor entity" na stránce 1664.

EntityType
Typ: MQLONG-vstup

Typ entity. Typ entity určený hodnotou EntityData. Musí se jednat o jednu z následujících hodnot:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ComponentData
Typ: MQBYTExComponentDataLength -vstupní/výstupní

Data komponent. Tato data jsou uchovávána správcem front jménem této konkrétní komponenty. Všechny změny provedené touto komponentou jsou zachovány a jsou prezentovány při příštím volání některé z těchto funkcí komponent.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování
Typ: MQLONG-výstup

Indikátor pokračování nastavený komponentou. Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_CHECK_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

Pokud volání komponenty selže (to znamená, že *CompCode* vrátí MQCC_FAILED) a parametr *Continuation* je MQZCI_DEFAULT nebo MQZCI_CONTINUE, správce front bude i nadále volat další komponenty, pokud existují nějaké.

Je-li volání úspěšné (to znamená, že *CompCode* vrátí MQCC_OK), nevolají žádné další komponenty bez ohledu na nastavení parametru *Continuation* .

Pokud se volání nezdaří a parametr *Continuation* je MQZCI_STOP, pak se nevolají žádné další komponenty a vrátí se chyba správci front. Komponenty nemají žádné informace o předchozích voláních, takže parametr *Continuation* je před voláním vždy nastaven na hodnotu MQZCI_DEFAULT.

CompCode
Typ: MQLONG-výstup

Kód dokončení. Musí se jednat o jednu z následujících hodnot:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Příčina
Typ: MQLONG-výstup

Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

MQRC_NOT_PRIVILEGED
(2584, X'A18') Tento uživatel není privilegovaným ID uživatele.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Kódy dokončení a příčin 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 (Kopírování všech oprávnění) v systému IBM i

Tato funkce je poskytována komponentou autorizační služby. Je vyvolán správcem front pro kopírování všech autorizací, které jsou aktuálně platné pro referenční objekt, na jiný objekt.

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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

RefObjectNázev (MQCHAR48)-Vstup
Název referenčního objektu.

Název referenčního objektu, autorizace, které mají být kopírovány. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který mají být nastaveny přístupy. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

ObjectType (MQLONG)-vstup
Typ objektu.

Typ objektu zadaného 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_COPY_ALL_AUTHORITY má tento efekt stejný efekt jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

OBJEKT MQRC_UNKNOWN_REF_OBJECT
(2294, X'8F6') Referenční objekt není znám.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 (Delete authority) on IBM i

Tato funkce je poskytována komponentou autorizační služby a je vyvolána správcem front k odstranění všech autorizací přidružených k danému 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který se mají odstranit přístupy. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_DELETE_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 (Výčtová data oprávnění) v systému IBM i

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_4 a je vyvolána opakovaně správcem front k načtení všech dat oprávnění, která odpovídají kritériím výběru uvedeným v 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

StartEnumeration (MQLONG)-Vstup
Příznak označující, zda má být volání zahájeno výčtem.

Označuje, zda by mělo volání začít s výčtem dat oprávnění, nebo pokračovat ve výčtu dat oprávnění spuštěných předchozím voláním MQZ_ENUMERATE_AUTHORITY_DATA. Hodnota je jedna z následujících možností:

SPUŠTĚNÍ MQZSE_START
Začátek výčtu.

Volání je vyvoláno touto hodnotou pro spuštění výčtu dat oprávnění. Argument **Filter** udává kritéria výběru, která se mají použít při výběru dat oprávnění vrácených tímto a po sobě jdoucími voláními.

MQZ_CONTINUE
Pokračujte ve výčtu.

Volání je vyvoláno touto hodnotou, aby bylo možné pokračovat ve výčtu dat oprávnění. Parametr **Filter** je v tomto případě ignorován a lze jej zadat jako ukazatel Null (výběrová kritéria jsou určována parametrem **Filter** zadaným voláním, který byl nastaven parametrem *StartEnumeration* na hodnotu MQZSE_START).

Filtr (MQZAD)-vstup
Filtrovat.

Je-li *StartEnumeration* MQZSE_START, *Filter* uvádí kritéria výběru, která se mají použít k výběru dat oprávnění, která se mají vrátit. Je-li *Filter* ukazatel Null, nejsou použita žádná kritéria výběru, to znamená, že jsou vrácena všechna data oprávnění. Podrobnosti o kritériích výběru, která lze použít, viz “MQZAD (Data Authority) v systému IBM i” na stránce 1702 .

Je-li *StartEnumeration* MQZSE_CONTINUE, *Filter* je ignorován a lze jej zadat jako ukazatel null.

Délka AuthorityBufferLength (MQLONG)-input
Délka *AuthorityBuffer*.

Toto je délka v bajtech parametru **AuthorityBuffer** . Vyrovnávací paměť oprávnění musí být dostatečně velká, aby pojmula data, která se mají vrátit.

Výstup 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 a nejdelší název entity a nejdelší definovaný název domény.

Poznámka: Tento parametr je definován jako MQZAD, protože MQZAD se vždy vyskytuje na začátku vyrovnávací paměti. Je-li však vyrovnávací paměť ve skutečnosti deklarována jako vlastnost MQZAD, bude vyrovnávací paměť příliš malá-musí být větší než hodnota MQZAD, aby mohla být schopná zpracovat názvy MQZAD, MQZED, plus entity a domén.

Délka AuthorityDataLength (MQLONG)-výstup
Délka dat vrácených v *AuthorityBuffer*.

Toto je délka dat vrácených v produktu *AuthorityBuffer*. Je-li vyrovnávací paměť oprávnění příliš malá, je hodnota *AuthorityDataLength* nastavena na délku požadované vyrovnávací paměti a volání vrátí kód dokončení MQCC_FAILED a kód příčiny MQRC_BUFFER_LENGTH_ERROR.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_ENUMERATE_AUTHORITY_DATA má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

CHYBA_MQRC_BUFFER_LENGTH_ERROR
(2005, X'7D5') Parametr délky vyrovnávací paměti není platný.

MQRC_NO_DATA_AVAILABLE
(2379, X'94B') Nejsou k dispozici žádná data.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 pomocí komponenty služby autorizace MQZAS_VERSION_5 a je vyvolána správcem front k uvolnění přidruženého přiděleného prostředku. Je vyvolán, když byla dokončena aplikace pod všemi kontexty uživatele, například během volání MQDISC MQI.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_FREE_USER.

IBM i MQZ_GET_AUTHORITY (Získání oprávnění) v systému IBM i

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front k načtení autority, kterou má entita pro přístup k uvedenému 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

EntityName (MQCHAR12)-Vstup
Název entity.

Název entity, jejíž přístup k objektu má být načten. Maximální délka řetězce je 12 znaků; je-li kratší, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Je možné zadat následující hodnotu:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který má být získáno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění (MQLONG)-výstup
Orgán účetní jednotky.

Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro objekt MQZ_GET_AUTHORITY má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 (Získání explicitního oprávnění) v systému IBM i

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front k načtení autority, kterou má pojmenovaná skupina pro přístup k uvedenému objektu (ale

bez dalšího oprávnění skupiny **nikdo**), nebo oprávnění, které má primární skupina uvedeného činitele k přístupu k uvedenému 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 funkce MQZ_GET_EXPLICIT_AUTHORITY má následující parametry.

QMgrName (MQCHAR48)-Vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

EntityName (MQCHAR12)-Vstup
Název entity.

Název entity, ze které má být získán přístup k objektu. Maximální délka řetězce je 12 znaků; je-li kratší, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Je možné zadat následující hodnotu:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, pro který má být získáno oprávnění entity. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění (MQLONG)-výstup
Orgán účetní jednotky.

Má-li entita jedno oprávnění, toto pole se rovná odpovídající operaci autorizace (konstanta MQZAO_ *). Pokud má více než jedno oprávnění, je toto pole bitově operátorem OR odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_GET_EXPLICIT_AUTHORITY má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA MQRC_UNKNOWN_ENTITY
(2292, X'8F4') Entita neznámá pro službu.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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) v systému IBM i

Tato funkce je poskytována komponentou autorizační služby a je vyvolána správcem front během konfigurace komponenty. Očekává se, že zavolá 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 funkce MQZ_INIT_AUTHORITY má následující parametry.

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace.

Tento manipulátor představuje inicializaci konkrétní komponenty. Tuto komponentu je třeba použít při volání správce front s funkcí 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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

Délka ComponentData(MQLONG)-vstup
Délka dat komponenty.

Délka (v bajtech) oblasti *ComponentData* . Tato délka je definována v konfiguračních datech komponenty.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato volba je před voláním primární inicializační funkce komponenty inicializována na všechny nuly. Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty; všechny změny provedené kteroukoli z funkcí (včetně inicializační funkce) poskytované touto komponentou jsou zachovány a jsou prezentovány při příštím vyvolání jedné z funkcí této komponenty.

Verze (MQLONG)-vstupní/výstupní
Číslo verze.

Při vstupu do inicializační funkce toto identifikuje *nejvyšší* číslo verze, které správce front podporuje. Funkce inicializace musí v případě potřeby tuto verzi změnit na verzi rozhraní, které *ní* podporuje. Pokud při návratu správce front nepodporuje verzi vrácenou komponentou, volá funkci MQZ_TERM_AUTHORITY komponenty a dále ji nebude dále používat.

Jsou podporovány 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.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

INICIALIZACE MQRC_INITIALIZATION_SELHALA
(2286, X'8EE') Inicializace se nezdařila z nedefinované příčiny.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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 (Zjišťovat autorizační službu) v systému IBM i

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_5 a je vyvolána správcem front za účelem zpracování dotazů na podporovanou funkčnost. Je-li použito více komponent služeb, jsou komponenty služeb volány v opačném pořadí, než jsou instalovány v pořadí, ve kterém byly instalovány.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_INQUIRE.

Syntaxe

MQZ_DOTÁZAT SE
(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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném 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 povinný atribut a musí mít jeden z následujících typů:

- MQIACF_ * (celé číslo)

- MQCACF_ * (znak)

Selektory mohou být zadány v libovolném pořadí. Počet selektorů v poli je označen parametrem SelectorCount .

Celočíselné atributy určené selektory jsou vráceny v parametru IntAttrs ve stejném pořadí, v jakém se objevují v selektorech.

Atributy znaků určené selektory jsou vráceny v parametru CharAttrs ve stejném pořadí, ve kterém se objevují selektory.

Počet IntAttrCount (MQLONG)-input
Počet celočíselných atributů.

Počet celočíselných atributů dodaných v parametru IntAttrs .

Hodnota musí být v rozsahu 0 až 256.

IntAttrs (MQLONG x IntAttrCount)-výstup
Celočíselné atributy.

Pole celočíselných atributů. Celočíselné atributy jsou vráceny ve stejném pořadí jako odpovídající celočíselné selektory v poli Selektory.

Počet znaků CharAttr(MQLONG)-input
Délka vyrovnávací paměti atributů znaků.

Délka (v bajtech) parametru CharAttrs .

Hodnota musí být alespoň součtem délek požadovaných znakových atributů. Nejsou-li požadovány žádné znakové atributy, nula je platná hodnota.

CharAttrs (MQLONG x CharAttrCount)-výstup
Vyrovnávací paměť pro atributy znaků.

Vyrovnávací paměť obsahující atributy znaků, zřetězená dohromady. Atributy znaku se vrací ve stejném pořadí jako odpovídající selektory znaků v poli Selektory.

Délka vyrovnávací paměti je dána parametrem Počet CharAttr.

SelectorReturned (PočetMQLONGxSelectorCount)-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 možností:

FUNKCE MQZSL_RETURNED
Byl vrácen atribut požadovaný odpovídajícím selektorem v parametru Selektory.

MQZSL_NOT_RETURNED
Atribut požadovaný odpovídajícím selektorem v parametru Selektory nebyl vrácen.

Pole je inicializováno se všemi hodnotami jako *MQZSL_NOT_RETURNED*. Když komponenta autorizační služby vrátí atribut, nastaví příslušnou hodnotu v poli na *MQZSL_RETURNED*. To umožňuje ostatním komponentám autorizační služby, ke kterým je vytvořeno dotazová volání, za účelem zjištění, které atributy již byly vráceny.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Příznak pokračování.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na jiných komponentách.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

VAROVÁNÍ MQCC_WARNING
Částečné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_CHAR_ATTRS_TOO_SHORT
Nedostatek prostoru pro atributy znaků.

POČ_DO_LOKÁLNÍ_FRONTY_MQRC_INT_TOO_SMALL
Nedostatek prostoru pro celočíselné atributy.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_SELECTOR_COUNT_ERROR
Počet selektorů není platný.

CHYBA MQRC_SELECTOR_ERROR
Selektor atributu není platný.

MQRC_SELECTOR_LIMIT_EXCEEDED
Je zadáno příliš mnoho selektorů.

CHYBA_MQRC_INT_ATTR_COUNT_ERROR
Počet celočíselných atributů je neplatný.

CHYBA POLE MQRC_INT_ATTRS_ARRAY_ERROR
Pole celočíselné atributy není platné.

MQRC_CHAR_ATTR_LENGTH_ERROR
Počet znakových atributů je neplatný.

CHYBA MQRC_CHAR_ATTRS_ERROR
Řetězec znaků znaků je neplatný.

CHYBA SLUŽBY MQRC_SERVICE_
(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 (Obnovení všech autorizací) v systému IBM i

Tato funkce je poskytována komponentou autorizační služby MQZAS_VERSION_3 . Je vyvolán správcem front k aktualizaci seznamu oprávnění interně držených komponentou.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_REFRESH_CACHE (8L).

Syntaxe

MQZ_REFRESH_CACHE
(QMgrName, ComponentData, Continuation, CompCode, Reason)

Parametry

***QMgrName* (MQCHAR48)-vstup**
Název správce front.

Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

***ComponentData* (MQBYTE x *ComponentDataLength*) -vstup/výstup**
Data komponent.

Tato data jsou uchovávána správcem front v zastoupení této konkrétní komponenty. Jakékoli změny provedené kteroukoli z funkcí poskytovaných touto komponentou budou zachovány a zobrazí se při příštím volání funkce komponenty.

Délka této datové oblasti je předávána správcem front v parametru *ComponentDataLength* volání MQZ_INIT_AUTHORITY.

***Pokračování* (MQLONG)-výstup**
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro MQZ_REFRESH_CACHE má tento efekt stejný účinek jako MQZCI_CONTINUE.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

***CompCode* (MQLONG)-výstup**
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

***Příčina* (MQLONG)-výstup**
Kód příčiny kvalifikující *CompCode*.

Má-li parametr *CompCode* hodnotu MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li položka *CompCode* MQCC_FAILED:

CHYBA SLUŽBY MQRC_SERVICE_
(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 (Nastavení oprávnění) v systému IBM i

Tato funkce je poskytována pomocí komponenty služby autorizace MQZAS_VERSION_1 a je vyvolána správcem front k nastavení oprávnění, které má entita pro přístup k uvedenému objektu.

Identifikátor funkce pro tuto funkci (pro MQZEP) je MQZID_SET_AUTHORITY.

Poznámka: Tato funkce přepíše všechny existující oprávnění. Chcete-li zachovat existující oprávnění, je třeba je nastavit znovu s touto funkcí.

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 doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném 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, než je zprava vyplněno mezerami. Název není ukončen nulovým znakem.

EntityType (MQLONG)-vstup
Typ entity.

Typ entity určený parametrem *EntityName*. Je možné zadat následující hodnotu:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

ObjectName (MQCHAR48)-vstup
Název objektu.

Název objektu, ke kterému je přístup požadován. Maximální délka řetězce je 48 znaků; je-li kratší, než je vyplněna zprava mezerami. Název není ukončen nulovým znakem.

Je-li *ObjectType* MQOT_Q_MGR, tento název je 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.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění (MQLONG)-vstup
Oprávnění ke kontrole.

Je-li nastavena jedna autorizace, rovná se toto pole odpovídající operaci autorizace (konstanta MQZAO_ *)). Pokud je nastavována více než jedna autorizace, je to bitové OR z odpovídajících konstant MQZAO_ *.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** volání MQZ_INIT_AUTHORITY.

Pokračování (MQLONG)-výstup
Indikátor pokračování nastavený komponentou.

Mohou být uvedeny následující hodnoty:

VÝCHOZÍ HODNOTA MQZCI_DEFAULT
Pokračování závislé na správci front.

Pro vlastnost MQZ_SET_AUTHORITY má tento efekt stejný účinek jako MQZCI_STOP.

MQZCI_CONTINUE
Pokračujte s další komponentou.

MQZCI_STOP
Nepokračovat s další komponentou.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

CHYBA SLUŽBY MQRC_SERVICE_
(2289, X'8F1') Došlo k neočekávané chybě při přístupu ke službě.

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

ENTITA 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-Ukončení autorizační služby

Tato funkce je poskytována komponentou autorizační služby a je vyvolána správcem front, pokud již nevyžaduje služby této komponenty. Funkce musí provést jakékoli 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 funkce MQZ_TERM_AUTHORITY má následující parametry.

Hconfig (MQHCONFIG)-vstup
Popisovač konfigurace.

Tento popisovač představuje konkrétní komponentu, která se ukončuje.

Volby (MQLONG)-vstup
Volby ukončení.

Jedná se o jednu z následujících položek:

MQZTO_PRIMÁRNÍ
Primární ukončení.

MQZ_SEKUNDÁRNÍ
Sekundární ukončení.

QMgrName (MQCHAR48)-Vstup
Název správce front.

Název správce front, který volá komponentu. Tento název je doplněn mezerami na celou délku parametru; název není ukončen nulovým znakem.

Název správce front je předán komponentě pro informaci; rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobem.

ComponentData (MQBYTE x ComponentDataLength)-vstup/výstup
Data komponent.

Tato data jsou uchovávána správcem front jménem této konkrétní komponenty; všechny změny provedené kterýchkoli funkcí poskytovaných touto komponentou jsou zachovány a jsou prezentovány při příštím volání jedné z funkcí této komponenty.

Délka této datové oblasti je předána správcem front v parametru **ComponentDataLength** v rámci volání MQZ_INIT_AUTHORITY.

Po dokončení volání MQZ_TERM_AUTHORITY zahodí správce front tato data.

CompCode (MQLONG)-výstup
Kód dokončení.

Jedná se o jednu z následujících položek:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

Důvod (MQLONG)-výstup
Kód příčiny kvalifikující *CompCode*.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_FAILED:

MQRC_SERVICE_NOT_AVAILABLE
(2285, X'8ED') Služba Underlying není k dispozici.

SELHÁNÍ MQRC_TERMINATION_FAILED
(2287, X'8FF') Ukončení se nezdařilo z nedefinované příčiny.

Další informace o těchto kódech příčiny najdete v tématu Zprávy a kódy příčin.

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) v systému IBM i

Tento parametr uvádí data související s volající aplikací.

Struktura MQZAC se používá pro volání MQZ_AUTHENTICATE_USER pro parametr **ApplicationContext** .

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

ID_STRUKTURY MQZAC_STRUCT
Identifikátor struktury kontextu aplikace.

Pro programovací jazyk C je také definován konstantní MQZAC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAC_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG
Číslo verze struktury.

Hodnota je:

MQZAC_VERSION_1
Struktura kontextu aplikace Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZAC_AKTUÁLNÍ_VERZE
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.

U systémů IBM i se používá profil uživatele, pod kterým byla vytvořena aplikační úloha. (V systému IBM ise při výměně profilu s rozhraním QWTSETP API v úloze aplikace vrátí aktuální profil uživatele).

ID EffectiveUserID (MQCHAR12)
Efektivní identifikátor uživatele.

Pro systémy IBM i se používá aktuální profil uživatele úlohy aplikace.

Prostředí (MQLONG)
Prostředí.

Toto pole uvádí prostředí, ze kterého bylo volání provedeno.

Může mít jednu z následujících hodnot:

MQXE_PŘÍKAZOVÝ_SERVER
Příkazový server.

MQXE_MQSC
Interpret příkazu `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_EXTERNAL
Volání je externí pro správce front.

MQXACT_INTERNAL
Volání je interní pro správce front.

AuthenticationType (MQLONG)
Typ ověření.

Toto pole uvádí typ ověření, které se provádí.

Může mít jednu z následujících hodnot:

POČÁTEČNÍ_KONTEXT MQZATR_CONTEXT
Volání ověření je způsobeno inicializací kontextu uživatele. Tato hodnota se používá během volání MQCONN nebo MQCONNX .

KONTEXT MQZAT_CHANGE_CONTEXT
Volání ověření je způsobeno změnou kontextu uživatele. Tato hodnota se použije, když agent MCA změní kontext uživatele.

v

BindType (MQLONG
Typ vazby.

Toto pole uvádí typ vazby, která se má použít.

Může mít jednu z následujících hodnot:

VAZBA MQCNO_FASTPATH_BINDING
Vazba zrychleného přístupu.

CQCNO_SHARED_BINDING
Sdílená vazba.

VAZBA MQCNO_ISOLATED_BINDING
Samostatná vazba.

Deklarace C

```
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 Authority) v systému IBM i

Struktura MQZAD se používá v rámci volání MQZ_ENUMERATE_AUTHORITY_DATA pro dva parametry.

Další informace o parametrech **Filter** a **AuthorityBuffer** viz “MQZ_ENUMERATE_AUTHORITY_DATA (Výčtová data oprávnění) v systému IBM i” na stránce 1682 :

- Objekt MQZAD se používá pro parametr **Filter** , který je vstupem pro volání. Tento parametr uvádí kritéria výběru, která mají být použita pro výběr dat oprávnění vrácených voláním.
- Objekt MQZAD se také používá pro parametr **AuthorityBuffer** , který je výstupem z volání. Tento parametr určuje autorizace pro jednu kombinaci názvu profilu, typu objektu a entity.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

ID_KONSTRUKCE_MQZAD_OBJEKTU
Identifikátor pro datovou strukturu oprávnění.

Pro programovací jazyk C je také definována konstanta MQZAD_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZAD_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZAD_VERSION_1
Datová struktura oprávnění Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

V_AKTUÁLNÍ_VERZE MQZAD_
Aktuální verze datové struktury oprávnění.

Toto je vstupní pole pro službu.

ProfileName (MQCHAR48)
Název profilu.

U parametru **Filter** je toto pole názvem profilu, ze kterého jsou data oprávnění vyžadována. Je-li název zcela prázdný až do konce pole nebo prvního znaku null, vrátí se data oprávnění pro všechny názvy profilů.

U parametru **AuthorityBuffer** je toto pole názvem profilu, který odpovídá zadaným kritériím výběru.

ObjectType (MQLONG)
Typ objektu.

U parametru **Filter** je toto pole typ objektu, pro který jsou vyžadována data oprávnění. Je-li hodnota MQOT_ALL, je vrácena data oprávnění pro všechny typy objektů.

U parametru **AuthorityBuffer** je v tomto poli typ objektu, na který se vztahuje profil určený hodnotou **ProfileName** .

Hodnota je jedna z následujících možností; pro parametr **Filter** je hodnota MQOT_ALL také platná:

MQOT_AUTH_INFO
Ověřovací informace.

MQOT_CHANNEL
Kanál.

MQOT_CLNTCONN_CHANNEL
Kanál připojení klienta.

MQOT_LISTENER
Modul listener.

MQO_NAMELIST
Seznam jmen.

PROCES MQOT_PROCESS
Definice procesu.

MQOT_Q
Fronta.

MQOT_Q_MGR
Správce front.

SLUŽBA MQOT_SERVICE
Servis.

Oprávnění (MQLONG)
Oprávnění.

U parametru **Filter** je toto pole ignorováno.

U parametru **AuthorityBuffer** toto pole představuje oprávnění, která má entita k objektům identifikovaným pomocí **ProfileName** a **ObjectType**. Má-li entita pouze jedno oprávnění, je pole

rovno odpovídající hodnotě autorizace (MQZAO_ * konstanta). Má-li entita více než jedno oprávnění, je toto pole bitové OR z odpovídajících konstant MQZAO_ *.

EntityDataPtr (PMQZED)
Adresa struktury MQZED, která identifikuje entitu.

Pro parametr **Filter** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, z níž jsou data oprávnění vyžadována. Je-li **EntityDataPtr** ukazatel null, jsou vrácena data oprávnění pro všechny entity.

Pro parametr **AuthorityBuffer** toto pole ukazuje na strukturu MQZED, která identifikuje entitu, ze které pochází vrácená data oprávnění.

EntityType (MQLONG)
Typ entity.

Pro parametr **Filter** toto pole uvádí typ entity, pro který jsou vyžadována data oprávnění. Je-li hodnota MQZAET_NONE, vrátí se data oprávnění pro všechny typy entit.

U parametru **AuthorityBuffer** toto pole určuje typ entity identifikované strukturou MQZED, na kterou je odkazováno pomocí příkazu **EntityDataPtr**.

Hodnota je jedna z následujících možností; pro parametr **Filter** je hodnota MQZAET_NONE také platná:

ČINITEL MQZAET_PRINCIPAL
Řediteli.

SKUPINA MQZAET_GROUP
:NONE.

Volby (MQAUTHOPT)
Volby.

Toto pole uvádí volby, které dávají kontrolu nad zobrazenými profily.

Musí být uvedena jedna z následujících možností:

MQAUTHOPT_NAME_ALL_MATCHING
Zobrazí všechny profily

MQAUTHOPT_NAME_EXPLICIT
Zobrazí profily, které mají přesně stejný název, jak je uvedeno v poli **ProfileName** .

Kromě toho musí být zadán také jeden z následujících:

MQAUTHOPT_ENTITY_SET
Zobrazí všechny profily používané k výpočtu kumulativního oprávnění, které má entita k objektu specifiknému **ProfileName**. Pole **ProfileName** nesmí obsahovat žádné zástupné znaky.

- Je-li uvedená entita činitelem, zobrazí se pro každého člena sady {entity, groups} nejvhodnější profil, který se vztahuje na daný objekt.
- Je-li uvedená entita skupina, zobrazí se nejvhodnější profil ze skupiny, která se vztahuje na objekt.
- Je-li zadána tato hodnota, musí být hodnoty **ProfileName**, **ObjectType**, **EntityType**a názvu entity zadané ve struktuře **EntityDataPtr** MQZED všechny neprázdné.

Pokud jste uvedli *MQAUTHOPT_NAME_ALL_MATCHING*, můžete také uvést následující:

MQAUTHOPT_ENTITY_EXPLICIT
Zobrazí profily, které mají přesně stejný název entity, jako je název entity určený ve struktuře **EntityDataPtr** MQZED.

Deklarace C

```
typedef struct tagMQZAD MQZAD;
struct tagMQZAD {
```

```
   MQCHAR4   StrucId;        /* Structure identifier */
   MQLONG    Version;        /* Structure version number */
   MQCHAR48  ProfileName;    /* Profile name */
   MQLONG    ObjectType;     /* Object type */
   MQLONG    Authority;      /* Authority */
   PMQZED    EntityDataPtr;  /* Address of MQZED structure identifying an
                                entity */
   MQLONG    EntityType;     /* Entity type */
   MQAUTHOPT Options;        /* Options */
};
```

IBM i MQZED (popisovač entity) v systému IBM i

Struktura MQZED se používá v mnoha voláních autorizační služby k určení entity, pro kterou se má ověřit autorizace.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

ID_STRUKTURY MQZED_STRUCT
Identifikátor struktury deskriptoru entity.

Pro programovací jazyk C je také definována konstanta MQZED_STRUC_ID_ARRAY; hodnota 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
Struktura deskriptoru entity Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZED_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 autorizaci má být zkontrolována.

EntityDomainPtr (PMQCHAR)
Adresa názvu domény entity.

Jedná se o ukazatel na název domény obsahující definici entity, jejíž autorizaci má být zkontrolována.

SecurityId (MQBYTE40)
Identifikátor zabezpečení.

Jedná se o identifikátor zabezpečení, jehož autorizaci má být zkontrolována.

CorrelationPtr (MQPTR)
Ukazatel korelace.

To usnadňuje předávání korelačních dat mezi funkcí authenticate user a dalšími vhodnými funkcemi OAM.

Deklarace C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  PMQCHAR   EntityNamePtr;    /* Address of entity name */
  PMQCHAR   EntityDomainPtr;  /* Address of entity domain name */
  MQBYTE40  SecurityId;       /* Security identifier */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
```

IBM i MQZFP (volné parametry) v systému IBM i

Tento parametr uvádí data související s prostředkem, který má být uvolněn.

Struktura MQZFP se používá v rámci volání MQZ_FREE_USER pro parametr **FreeParms** .

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

ID_STRUKTURY MQZFP_STRUCT
Identifikátor pro strukturu volných parametrů.

Pro programovací jazyk C je také definována 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
Struktura parametrů volných parametrů Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZFP_CURRENT_VERSION
Aktuální verze struktury volných parametrů.

Toto je vstupní pole pro službu.

Rezervováno (MQBYTE8)
Rezervované pole.

Počáteční hodnota je null.

CorrelationPtr (MQPTR)
Ukazatel korelace.

Adresa korelačních dat souvisejících s prostředkem, který má být uvolněn.

Deklarace C

```
typedef struct tagMQZFP MQZFP;
struct tagMQZFP {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQBYTE8   Reserved;         /* Reserved field */
  MQPTR     CorrelationPtr;   /* Address of correlation data */
};
```

IBM i MQZIC (kontext identity) v systému IBM i

Struktura MQZIC se používá pro 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 z aplikace, která vložila zprávu.
- Aplikace mohou používat datové pole ApplIdentitypro jakékoli další informace, které chtějí zahrnout o uživateli (například zašifrované heslo).

Autorizované autorizované aplikace mohou nastavit kontext identity použitím funkce MQZ_AUTHENTICATE_USER.

Identifikátor zabezpečení systému Windows (SID) je uložen v poli AccountingToken , je-li vytvořena zpráva pod IBM MQ for Windows. Identifikátor SID lze použít k doplnění pole UserIdentifier a k ustanovení pověření uživatele.

Pole

StrucId (MQCHAR4)
Identifikátor struktury.

Hodnota je:

MQZIC_STRUCTION_ID
Identifikátor struktury kontextu identity.

Pro programovací jazyk C je také definována konstanta MQZIC_STRUC_ID_ARRAY; má stejnou hodnotu jako MQZIC_STRUC_ID, ale je to pole znaků namísto řetězce.

Toto je vstupní pole pro službu.

Verze (MQLONG)
Číslo verze struktury.

Hodnota je:

MQZIC_VERSION_1
Struktura kontextu identity Version-1 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQZIC_AKTUÁLNÍ_VERZE
Aktuální verze struktury kontextu identity.

Toto je vstupní pole pro službu.

UserIdentifier (MQCHAR12)
Identifikátor uživatele.

Tato část je součástí **kontextu identity** zprávy.

UserIdentifier uvádí identifikátor uživatele aplikace, která je původcem zprávy. Správce front považuje tyto informace za znaková data, ale nedefinuje její formát. Další informace o poli *UserIdentifier* viz “UserIdentifier (MQCHAR12)” na stránce 454.

AccountingToken (MQBYTE32)
Token evidence.

Tato část je součástí **kontextu identity** zprávy.

AccountingToken umožňuje aplikaci způsobit práci provedenou jako výsledek zprávy, která má být patřičně nabitá. Správce front považuje tyto informace za řetězec bitů a nekontroluje jeho obsah. Další informace o poli *AccountingToken* viz “AccountingToken (MQBYTE32)” na stránce 455.

Data ApplIdentity(MQCHAR32)
Data aplikace související s identitou.

Tato část je součástí **kontextu identity** zprávy.

ApplIdentityData jsou informace, které jsou definovány sadou aplikací, které lze použít k poskytnutí dalších informací o původu zprávy. Například by mohly být nastaveny aplikacemi, které jsou spuštěny s odpovídajícím oprávněním uživatele, aby označovaly, zda jsou data identity důvěryhodná. Další informace o poli *ApplIdentityData* viz “Data ApplIdentity(MQCHAR32)” na stránce 457.

Deklarace C

```
typedef struct tagMQZED MQZED;
struct tagMQZED {
  MQCHAR4   StrucId;          /* Structure identifier */
  MQLONG    Version;          /* Structure version number */
  MQCHAR12  UserIdentifier;   /* User identifier */
  MQBYTE32  AccountingToken;  /* Accounting token */
  MQCHAR32  ApplIdentityData; /* Application data relating to identity */
};
```

Třídy a rozhraní produktu IBM MQ .NET

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

Třída MQAsyncStatus.NET

MQAsyncStatus se používá k dotazům na stav předchozí aktivity MQI, například dotazy na úspěch předchozích asynchronních operací 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 1708
- “Konstruktory” na stránce 1709

Vlastnosti

Testuje se test produktu 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 pro volání MQI.

public static int PutWarningCount {get;}

Počet asynchronních volání pro volání MQI, která byla úspěšná s varováním.

public static int PutFailureCount {get;}

Počet nezdařených asynchronních volání MQI 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 objektu.

public static string ResolvedObjectName {get;}

Vyřešený název objektu.

public static string ResolvedObjectQMgrName {get;}

Vyřešený název správce front objektu.

Konstruktory

public MQAsyncStatus() throws MQException;

Metoda konstruktoru, konstruuje objekt s poli inicializovanými na nulu nebo prázdné, jak je to vhodné.

Třída MQAuthenticationInformationRecord.NET

Použijte `MQAuthenticationInformationRecord` k uvedení informací o ověřovateli, který má být použit 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 1710

- “Konstruktory” na stránce 1710

Vlastnosti

Testuje se test produktu `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 se použije pouze v případě, že je parametr **AuthInfoType** nastaven na hodnotu `CRLLDAP`.

public string LDAPUserName {get; set;}

Rozlišující jméno uživatele, který přistupuje k serveru LDAP. Nastavíte-li tuto vlastnost, jsou hodnoty `LDAPUserNameLength` a `LDAPUserNamePtr` automaticky nastaveny 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 je použita pouze v případě, že je volba `AuthInfoType` nastavena na hodnotu `OCSP` .

Toto pole rozlišuje velikost písmen. Musí začínat řetězcem http:// malými písmeny. Zbytek adresy URL může být citlivý na velikost písmen, v závislosti na implementaci serveru OCSP.

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 vytvořit její 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

Testuje se test produktu MQException při získávání vlastností.

public DateTime CreationDateTime {get;}

Datum a čas, kdy byla fronta nebo téma vytvořeny. Původně byl obsažen v produktu MQQueue, tato vlastnost byla přesunuta do základní třídy MQDestination .

Není zde žádná výchozí hodnota.

public int DestinationType {get;}

Celočíselná hodnota popisující typ použitého místa určení. Inicializováno z konstruktoru dílčích tříd, MQQueue nebo MQTopic, tato hodnota 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 MQException.

Získá zprávu z fronty, je-li cílem objekt MQQueue nebo z tématu, je-li cílem objekt produktu MQTopic , používá se výchozí instance produktu MQGetMessageOptions k provedení operace get.

Pokud dojde k selhání operace get, objekt MQMessage se nezmění. Pokud je úspěšný, jsou popisovač zprávy a části dat zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání do IBM MQ z konkrétního MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s čekáním, všechny ostatní podprocesy používající stejný MQQueueManager jsou blokovány od dalších volání IBM MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu IBM MQ současně, každý podproces musí vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá z polí v deskriptoru zpráv jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Reconnectable klient vrací kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení, pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby ovládající akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při konverzi z jednobajtových znakových kódů do dvoubajtových kódů. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez konverze.

Není-li parametr *getMessageOptions* zadán, bude použita volba zprávy MQGMO_NOWAIT.

Použijete-li volbu MQGMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Je-li zpráva ve frontě větší než tato velikost, nastane jedna ze dvou situací:

- Je-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven v objektu MQGetMessageOptions , zpráva je vyplněna co nejvíce informací o zprávě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED .
- Není-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven, zůstává zpráva ve frontě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED .

Není-li parametr *MaxMsgSize* zadán, bude načtena celá zpráva.

```
public void Put(MQMessage message);
public void Put(MQMessage message, MQPutMessageOptions putMessageOptions);
```

Vyvolá MQException.

Převede zprávu do fronty v případě, že cílem je objekt MQQueue , nebo publikuje zprávu do tématu, je-li cílem objekt MQTopic .

Úpravy objektu MQMessage po dokončení volání operace Put nemají vliv na skutečnou zprávu ve frontě IBM MQ nebo v tématu publikování.

Produkt Put aktualizuje vlastnosti MessageId a CorrelationId objektu MQMessage a nemaže data zprávy. Další volání Put nebo Get odkazují na aktualizované informace v objektu MQMessage . Například v následujícím úseku kódu bude první zpráva obsahovat a a druhý ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zpráv a zpráva, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zpráv bezprostředně po dokončení této metody jsou hodnotami, které byly vloženy do fronty nebo publikovány do tématu.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- MQRC_CALL_INTERRUPTED je-li připojení přerušeno při spuštění volání vložení na trvalou zprávu a opětovné navázání spojení je úspěšné.
- MQRC_NONE , je-li připojení úspěšné při spuštění volání vložení na netrvalou zprávu (viz Obnova aplikace).

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu MQPutMessageOptons .

Použijete-li volbu MQPMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Pokud chcete do fronty vložit jednu zprávu, použijte objekt MQQueueManager.Put pro zjednodušení a výkon. Pro tento objekt byste měli mít objekt MQQueue .

Konstruktory

MQDestination je abstraktní základní třída a nelze vytvořit její instanci. Přistupte k místům určení pomocí konstruktorů MQQueue a MQTopic , nebo pomocí MQQueueManager.AccessQueue a MQQueueManager.AccessTopic methods.

Třída MQEnvironment.NET

Použijte MQEnvironment k řízení, jak se volá konstruktor MQQueueManager a vyberte 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 1713
- "Vlastnosti" na stránce 1714
- "Konstruktory" na stránce 1715

Vlastnosti-pouze klient

Testuje se test produktu MQException při získávání vlastností.

public static int CertificateValPolicy {get; set;}

Nastavit, která zásada ověření platnosti certifikátu TLS se použije k ověření platnosti 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 Suite B. Platné jsou tyto hodnoty:

- MQC.MQ_SUITE_B_NONE -Jedná se o výchozí hodnotu.
- 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 instance produktu MQQueueManager v režimu klienta je třeba nastavit vlastnost kanálu.

public static int FipsRequired {get; set;}

Zadejte MQC.MQSSL_FIPS_YES , chcete-li použít pouze algoritmy certifikovaný FIPS, je-li šifrování prováděno v produktu IBM MQ. Standardní hodnota je MQC.MQSSL_FIPS_NO.

Je-li konfigurován kryptografický hardware, použijí se použité šifrovací moduly, které jsou poskytovány hardwarovým produktem. V závislosti na tom, který hardware se používá, nemusí být FIPS certifikován na konkrétní úroveň.

public static string Hostname {get; set;}

Název hostitele TCP/IP na počítači, na kterém je umístěn server IBM MQ . Není-li název hostitele nastaven a nejsou nastaveny žádné přepisující vlastnosti, použije se pro připojení k lokálnímu správci front režim vazeb serveru.

public static int Port {get; set;}

Port, ke kterému se chcete 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 hodnotu SSLCipherSpec na hodnotu sady CipherSpec nastavenou v kanálu SVRCONN, aby bylo možné povolit zabezpečení TLS pro připojení. Výchozí hodnota je Null a TLS není povoleno pro připojení.

public static string sslPeerName {get; set;}

Vzorek rozlišujícího názvu. Je-li nastavena volba sslCipherSpec , lze tuto proměnnou použít k ujištění, že je použit správný správce front. Je-li nastaveno na hodnotu null (výchozí), DN správce front se neprovede. Hodnota sslPeerName je ignorována, je-li parametr sslCipherSpec null.

Vlastnosti

Testuje se test produktu MQException při získávání vlastností.

public static ArrayList HdrCompList {get; set;}
Seznam komprese dat záhlaví

public static int KeyResetCount {get; set;}
Označuje počet nezašifrovaných 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 zprávy

public static string Password {get; set;}
Heslo, které se má ověřit. Heslo, na které se odkazuje struktura MQCSP , se naplní nastavením této vlastnosti Password.

public static string ReceiveExit {get; set;}
Uživatelská procedura příjmu vám umožňuje zkontrolovat a změnit data přijatá od správce front. Obvykle se používá s odpovídající uživatelskou procedurou odeslání ve správci front. Je-li volba ReceiveExit nastavena na hodnotu null, nebude volána žádná uživatelská procedura pro přijetí zprávy.

public static string ReceiveUserData {get; set;}
Uživatelská data přidružená k ukončení příjmu. Omezeno na 32 znaků.

public static string SecurityExit {get; set;}
Uživatelská procedura zabezpečení vám umožňuje přizpůsobit průběhy zabezpečení, které se vyskytnou při pokusu o připojení ke správci front. Je-li parametr SecurityExit nastaven na hodnotu null, není volána žádná uživatelská procedura pro zabezpečení zprávy.

public static string SecurityUserData {get; set;}
Uživatelská data přidružená k ukončení zabezpečení. Omezeno na 32 znaků.

public static string SendExit {get; set;}
Uživatelská procedura odeslání vám umožňuje zkontrolovat nebo změnit data odesílaná správci front. Obvykle se používá s odpovídající uživatelskou procedurou příjmu ve správci front. Je-li parametr SendExit nastaven na hodnotu null, nebude volána žádná uživatelská procedura pro odeslání zprávy.

public static string SendUserData {get; set;}
Uživatelská data přidružená k ukončení odeslání. Omezeno na 32 znaků.

public static string SharingConversations {get; set;}

Pole SharingConversations se používá pro připojení z aplikací produktu .NET , pokud tyto aplikace nepoužívají tabulku definic kanálů klienta (CCDT).

Volba SharingConversations určuje maximální počet konverzací, které lze sdílet na soketu přidruženém k tomuto připojení.

Hodnota 0 znamená, že kanál pracuje stejně jako předtím, než je IBM WebSphere MQ 7.0, pokud jde o sdílení konverzace, čtení napřed a prezenční signál.

Pole se předává v hašovací tabulce vlastností jako SHARING_CONVERSATIONS_PROPERTYpři vytváření instance správce front IBM MQ .

Pokud neuvedete SharingConversations, použije se výchozí hodnota 10.

public static string SSLCryptoHardware {get; set;}

Nastaví název řetězce parametru potřebného ke konfiguraci kryptografického hardwaru, který se nachází v systému. SSLCryptoHardware je ignorován, pokud sslCipherSpec má hodnotu null.

public static string SSLKeyRepository {get; set;}

Nastavte úplný název souboru úložiště klíčů.

Je-li parametr SSLKeyRepository nastaven na hodnotu null (výchozí), použije se k vyhledání úložiště klíčů certifikát prostředí MQSSLKEYR . SSLCryptoHardware je ignorován, pokud sslCipherSpec má hodnotu null.

Poznámka: Přípona .kdb je povinná část názvu souboru, ale není zahrnuta jako část hodnoty parametru. Adresář, který uvedete, musí existovat. Produkt IBM MQ vytvoří tento soubor při prvním přístupu k novému úložišti klíčů, pokud tento soubor již neexistuje.

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 uživatelské procedury API nebo zabezpečení.

Konstruktory

public MQEnvironment()

Třída MQException.NET

Použijte MQException k vyhledání dokončení a kód příčiny selhání funkce IBM MQ . Pokud dojde k chybě IBM MQ , dojde k vyvolání příkazu MQException .

Třída

```
System.Object
        │
        └─ System.Exception
                │
                └─ System.ApplicationException
                        │
                        └─ IBM.WMQ.MQException
```

```
public class IBM.WMQ.MQException extends System.ApplicationException;
```

- “Vlastnosti” na stránce 1715
- “Konstruktory” na stránce 1716

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;}
IBM MQ kód příčiny popisující chybu.

Konstruktory

public MQException(int *completionCode*, int *reasonCode*)

completionCode
Kód dokončení IBM MQ .

reasonCode
Kód dokončení IBM MQ .

Třída MQGetMessageOptions.NET

Použijte `MQGetMessageOptions` k uvedení, jak se zprávy načítají. Upravuje chování produktu `MQDestination.Get`.

Třída

```
System.Object
        │
        └─ IBM.WMQ.MQBase
                │
                └─ IBM.WMQ.MQBaseObject
                        │
                        └─ IBM.WMQ.MQGetMessageOptions
```

```
public class IBM.WMQ.MQGetMessageOptions extends IBM.WMQ.MQBaseObject;
```


- "Vlastnosti" na stránce 1716
- "Konstruktory" na stránce 1718

Vlastnosti

Poznámka: Chování některých voleb dostupných v této třídě závisí na prostředí, ve kterém jsou použity. Tyto prvky jsou označeny hvězdičkou *.

Testuje se test produktu `MQException` při získávání vlastností.

public int GroupStatus {get;}*
`GroupStatus` označuje, zda se načtená zpráva nachází ve skupině a zda je poslední ve skupině. Možné hodnoty jsou:

MQC.MQGS_LAST_MSG_IN_GROUP
Zpráva je poslední nebo jedinou zprávou ve skupině.

MQC.MQGS_MSG_IN_GROUP
Zpráva se nachází ve skupině, ale není poslední ve skupině.

MQC.MQGS_NOT_IN_GROUP
Zpráva se nenachází ve skupině.

public int MatchOptions {get; set;}*
`MatchOptions` určuje, jak je vybrána zpráva. Je možné nastavit následující volby shody:

MQC.MQMO_MATCH_CORREL_ID
ID korelace, které se má porovnat.

MQC.MQMO_MATCH_GROUP_ID
ID skupiny, které se má porovnat.

MQC.MQMO_MATCH_MSG_ID
ID zprávy, která se má porovnat.

MQC.MQMO_MATCH_MSG_SEQ_NUMBER
Porovnávat pořadové číslo zprávy.

MQC.MQMO_NONE
Nepožaduje se žádná shoda.

public int Options {get; set;}

Volby řídí akci produktu MQQueue.get. Může být uvedena jakákoli z následujících hodnot. Je-li požadována více než jedna volba, lze hodnoty přidat nebo zkombinovat s použitím bitového operátoru OR.

MQC.MQGMO_ACCEPT_TRUNCATED_MSG
Povolit oříznutí dat zprávy.

MQC.MQGMO_ALL_MSGS_AVAILABLE*
Načíst zprávy ze skupiny pouze tehdy, jsou-li k dispozici všechny zprávy ve skupině.

MQC.MQGMO_ALL_SEGMENTS_AVAILABLE*
Načíst segmenty logické zprávy pouze tehdy, jsou-li 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
Procházet od aktuální pozice ve frontě.

MQC.MQGMO_COMPLETE_MSG*
Načíst pouze úplné logické zprávy.

MQC.MQGMO_CONVERT
Vyžádejte data o aplikaci, která se mají převést, aby vyhovovala atributům CharacterSet a Encoding produktu MQMessage, než se data zkopírují do vyrovnávací paměti zpráv. Vzhledem k tomu, že převod dat je také použit při načítání dat z vyrovnávací paměti zpráv, aplikace tuto volbu nenastavujte.

Použití této volby může způsobit problémy při konverzi z jednobajtových znakových sad na dvoubajtová znaková sada. Místo toho proveďte převod s použitím metod readString, readLinea writeString po doručení zprávy.

MQC.MQGMO_FAIL_IF_QUIESCING
Selhat, pokud je správce front uváděn do klidového stavu.

MQC.MQGMO_LOCK*
Zamkni zprávu, která je procházena.

MQC.MQGMO_LOGICAL_ORDER*
Vrátit zprávy ve skupinách a segmentech logických zpráv v logickém pořadí.

Použijete-li volbu MQGMO_LOGICAL_ORDER v reconnectable client, vrátí se do aplikace kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

MQC.MQGMO_MARK_SKIP_BACKOUT*
Umožněte, aby byla jednotka práce vrácena, aniž by byla zpráva znovu vrácena do fronty.

MQC.MQGMO_MSG_UNDER_CURSOR
Získat zprávu pod kurzorem procházení.

MQC.MQGMO_NONE
Žádné další volby nebyly zadány; všechny volby předpokládají jejich výchozí hodnoty.

MQC.MQGMO_NO_PROPERTIES
Žádné vlastnosti zprávy, kromě vlastností obsažených v deskriptoru (či rozšíření) zprávy, které se načtou.

MQC.MQGMO_NO_SYNCPOINT
Získat zprávu bez řízení synchronizačního bodu.

MQC.MQGMO_NO_WAIT
Okamžitě se vraťte, pokud není k dispozici žádná vhodná zpráva.

MQC.MQGMO_PROPERTIES_AS_Q_DEF
Načtení vlastností zprávy, jak je definováno atributem `PropertyControl` produktu `MQQueue`. Přístup k vlastnostem zprávy v deskriptoru zprávy nebo rozšíření není ovlivněn atributem `PropertyControl` .

MQC.MQGMO_PROPERTIES_COMPATIBILITY
Načtěte vlastnosti zprávy s předponou `mcd`, `jms`, `usr`nebo `mqext`, v záhlaví `MQRFH2` . Ostatní vlastnosti zprávy, kromě vlastností obsažených v deskriptoru zpráv nebo rozšíření, jsou vyřazeny.

MQC.MQGMO_PROPERTIES_FORCE_MQRFH2
Načtení vlastností zprávy s výjimkou vlastností obsažených v deskriptoru zpráv nebo rozšíření v záhlaví `MQRFH2` . Použijte `MQC.MQGMO_PROPERTIES_FORCE_MQRFH2` v aplikacích, které čekají na načtení vlastností, ale nelze je změnit pro použití obslužných rutin zpráv.

MQC.MQGMO_PROPERTIES_IN_HANDLE
Načtěte vlastnosti zprávy pomocí volby `MsgHandle`.

MQC.MQGMO_SYNCPOINT
Získejte zprávu pod řízením synchronizačního bodu. Zpráva je označena jako nedostupná pro jiné aplikace, ale je vymazána z fronty pouze tehdy, když je potvrzena transakce. Zpráva je znovu zpřístupněna, pokud je jednotka práce zálohována.

MQC.MQGMO_SYNCPOINT_IF_PERSISTENT*
Získejte zprávu s řízením synchronizačního bodu, je-li zpráva trvalá.

MQC.MQGMO_UNLOCK*
Odemknout dříve zamčenou zprávu.

MQC.MQGMO_WAIT
Počkejte na doručení zprávy.

public string ResolvedQueueName {get;}
Správce front nastaví název `ResolvedQueueName` na lokální název fronty, ze které byla zpráva načtena. Hodnota `NázevResolvedQueue` se liší od názvu použitého k otevření fronty v případě, že byla otevřena fronta aliasů nebo fronta modelu.

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 uvádí, zda načtená zpráva je segment logické zprávy. Je-li zpráva segment, příznak označuje, zda se jedná o poslední segment. Možné hodnoty jsou:

MQC.MQSS_LAST_SEGMENT
Zpráva je poslední nebo jediný segment logické zprávy.

MQC.MQSS_NOT_A_SEGMENT
Zpráva není segment.

MQC.MQSS_SEGMENT
Zpráva je segment, ale nejedná se o poslední segment logické zprávy.

public int WaitInterval {get; set;}
Hodnota `WaitInterval` je maximální doba v milisekundách, po kterou volání `MQQueue.get` čeká na doručení vhodné zprávy. Použijte parametr `WaitInterval` s `MQC.MQGMO_WAIT`. Chcete-li čekat na neomezenou dobu pro zprávu, nastavte hodnotu `MQC.MQWI_UNLIMITED` .

Konstruktory

public MQGetMessageOptions()
Vytvořte nový objekt `MQGetMessageOptions` s volbami `Options` nastaveným na hodnotu `MQC.MQGMO_NO_WAIT`, `WaitInterval` nastaveným na nulu a `ResolvedQueueName` je prázdný.

Třída MQManagedObject.NET

Použijte MQManagedObject k zjištění a nastavení atributů MQDestination, MQProcess, MQQueueManagera MQSubscription. MQManagedObject je nadtřída těchto tříd.

Třídy

```
System.Object
        |
        └ IBM.WMQ.MQBase
                |
                └ IBM.WMQ.MQBaseObject
                        |
                        └ IBM.WMQ.MQManagedObject
                                |
                                └ IBM.WMQ.MQProcess
                                |
                                └ IBM.WMQ.MQQueueManager
                                |
                                └ IBM.WMQ.MQSubscription
                                |
                                └ IBM.WMQ.MQDestination
```

```
public class IBM.WMQ.MQManagedObject extends IBM.WMQ.MQBaseObject;
```


- "Vlastnosti" na stránce 1719
- "Metody" na stránce 1720
- "Konstruktory" na stránce 1721

Vlastnosti

Testuje se test produktu MQException při získávání vlastností.

public string AlternateUserId {get; set;}
Alternativní ID uživatele, pokud bylo nastaveno, nastaveno při otevření prostředku. AlternateUserID.set se ignoruje, když je vydán pro objekt, který je otevřený. Položka AlternateUserId není platná pro odběry.

public int CloseOptions {get; set;}

Nastavením tohoto atributu ovládíte způsob, jakým je prostředek uzavřen. Výchozí hodnota je MQC.MQCO_NONE. MQC.MQCO_NONE je jedinou přípustnou hodnotou pro všechny prostředky jiné než trvalé dynamické fronty, dočasné dynamické fronty, odběry a témata, k nimž přistupují objekty, které je vytvořily.

Pro fronty a témata jsou přípustné následující dodatečné hodnoty:

MQC.MQCO_DELETE
Pokud zde nejsou žádné zprávy, odstraňte frontu.

MQC.MQCO_DELETE_PURGE
Odstraňte frontu a odstraňte na ní zprávy.

MQC.MQCO_QUIESCE
Vyžádejte frontu, aby byla zavřena, aby byla zobrazena varovná zpráva, pokud nějaké zprávy zůstanou (což jim umožní získat zpět před konečným uzavřením).

Pro odběry jsou přípustné následující doplňkové hodnoty:

MQC.MQCO_KEEP_SUB
Odběr nebyl odstraněn. Tato volba je platná pouze tehdy, je-li původní odběr trvalý. MQC.MQCO_KEEP_SUB je výchozí hodnota pro trvalé téma.

MQC.MQCO_REMOVE_SUB
Odběr je odstraněn. MQC.MQCO_REMOVE_SUB je výchozí hodnota pro netrvalé nespravované téma.

MQC.MQCO_PURGE_SUB
Odběr je odstraněn. MQC.MQCO_PURGE_SUB is the default value for a non-durable, managed topic.

public MQQueueManager ConnectionReference {get;}
Správce front, do kterého patří tento prostředek.

public string MQDescription {get;}
Popis prostředku v držení správce 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 zadán v metodě přístupu, nebo název 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 OpenOptions.

Metody

public virtual void Close();
Vyvolá MQException.

Zavře objekt. Po volání Closenejsou povoleny žádné další operace proti tomuto prostředku. Chcete-li změnit chování metody Close , nastavte atribut closeOptions .

public string GetAttributeString(int *selector*, int *length*);
Vyvolá MQException.

Získá řetězec atributu.

selektor
Celé číslo označující, na který atribut se dotazujete.

délka
Celé číslo udávající délku požadovaného řetězce.

public void Inquire(int[] *selectors*, int[] *intAttrs*, byte[] *charAttrs*);
Vyvolá MQException.

Vrací pole celých čísel a sadu znakových řetězců, které obsahují atributy fronty, procesu nebo správce front. Atributy, které mají být dotazovány, jsou určeny v poli selektorů.

Poznámka: Mnoho z běžnějších atributů může být dotazováno 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 celočíselné hodnoty atributu. Hodnoty celočíselných atributů se vrací ve stejném pořadí jako celočíselné selektory atributů v poli selektorů.

charAttrs
Vyrovnávací paměť, ve které jsou vrácené znakové atributy, zřetězené. Atributy znaků se vrací 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 MQException.

Nastaví atributy definované ve vektoru selektorů. Atributy, které mají být nastaveny, jsou určeny v poli selektorů.

selektory
Celočíselné pole identifikující atributy s hodnotami, které mají být nastaveny.

intAttrs
Pole celočíselných hodnot atributů, které mají být nastaveny. Tyto hodnoty musejí být ve stejném pořadí jako celočíselné selektory atributů v poli selektorů.

charAttrs
Vyrovnávací paměť, ve které jsou zřetězeny znakové atributy, které mají být nastaveny. Tyto hodnoty musejí 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 `MQException`.

Nastaví řetězec atributu.

selektor
Celé číslo označující, který atribut se nastavuje.

hodnota
Řetězec, který se má nastavit jako hodnota atributu.

délka
Celé číslo udávající délku požadovaného řetězce.

Konstruktory

protected MQManagedObject()
Metoda konstruktoru. Tento objekt je abstraktní základní třída, která nemůže být převedena na instanci sama.

Třída MQMessage.NET

Použijte `MQMessage` pro přístup k popisovači 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 potom použijte metody `Read` a `Write` pro přenos dat mezi zprávou a dalšími objekty ve vaší aplikaci. Objekty `MQMessage` lze odesílat a přijímat pomocí metod `Put` a `Get` tříd `MQDestination`, `MQQueue` a `MQTopic` .

Získejte a nastavte vlastnosti deskriptoru zpráv pomocí vlastností `MQMessage`. Nastavte a získejte rozšířené vlastnosti zprávy pomocí metod `SetProperty` a `GetProperty` .

- “Vlastnosti” na stránce 1721
- “Metody zpráv Read a Write” na stránce 1727
- “Metody vyrovnávací paměti” na stránce 1730
- “Metody vlastností” na stránce 1730
- “Konstruktory” na stránce 1733

Vlastnosti

Testuje se test produktu `MQException` při získávání vlastností.

public string AccountingToken {get; set;}
Část kontextu identity zprávy; pomáhá aplikaci účtovat za práci provedenou jako výsledek této 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 MQQueue.Get jako součást jednotky práce. Výchozí hodnota je 0.

public int CharacterSet {get; set;}

Identifikátor kódované znakové sady znakových dat ve zprávě.

Nastavte CharacterSet , abyste označili znakovou sadu znakových dat ve zprávě. Získejte CharacterSet , abyste zjistili, v jaké znakové sadě byla použita ke kódování znakových dat ve zprávě.

Aplikace produktu .NET se vždy spouštějí v Unicode, zatímco v jiných prostředích jsou aplikace spouštěny ve stejné znakové sadě, pod kterou je spuštěn správce front.

Metody ReadString a ReadLine převádějí znaková data ve zprávě na Unicode.

Metoda WriteString se převádí z kódování Unicode do znakové sady kódované v CharacterSet. Je-li parametr CharacterSet nastaven na svou výchozí hodnotu, MQC.MQCCSI_Q_MGR, což je 0, žádná konverze se neprovádí a CharacterSet je nastaven na hodnotu 1200. Pokud nastavíte volbu CharacterSet na některou jinou hodnotu, produkt WriteString převede z kódování Unicode na alternativní hodnotu.

Poznámka: Jiné metody čtení a zápisu nepoužívají CharacterSet.

- ReadChar a WriteChar čtou a zapisují znak Unicode do a z vyrovnávací paměti zpráv bez konverze.
- ReadUTF a WriteUTF konvertují mezi řetězcem Unicode v aplikaci a řetězcem UTF-8 , s předponou o 2 bajtové délce, ve vyrovnávací paměti zpráv.
- Bajtové metody přenášejí bajty mezi aplikací a vyrovnávací pamětí zpráv beze změny.

public byte[] CorrelationId {get; set;}

- U 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 zpráv. Výchozí hodnota MQC.MQCI_NONEpomáhá jakémukoli identifikátoru korelace, aby se shodoval.
- Pro volání funkce MQQueue.Put je korelační identifikátor nastaven.

public int DataLength {get;}
Počet bajtů dat zprávy, které zbývají 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;}
Znázornění použité pro číselné hodnoty v datech zprávy aplikace. Kódování platí pro binární data, pakovaná desetinná čísla a data s plovoucí řádovou čárkou. Chování metod čtení a zápisu pro tyto numerické formáty je odpovídajícím způsobem změněno. Vytvořte hodnotu pro pole kódování přidáním jedné hodnoty z každé z těchto tří sekcí. Alternativně můžete vytvořit hodnotu zkombinující 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, která se používá v architektuře Intel .

2. Pakovaný desetinný

MQC.MQENC_DECIMAL_NORMAL
Pakované desetinné endian, použité produktem z/OS.

MQC.MQENC_DECIMAL_REVERSED
Paked-endian-desetinný.

3. pohyblivá řádová čárka

MQC.MQENC_FLOAT_IEEE_NORMAL
Big-endian IEEE floats.

MQC.MQENC_FLOAT_IEEE_REVERSED
Little-endian IEEE floats, as used Intel architecture.

MQC.MQENC_FLOAT_S390
Plovoucí body formátu z/OS .

Výchozí hodnota je:

```
MQC.MQENC_INTEGER_REVERSED |
MQC.MQENC_DECIMAL_REVERSED |
MQC.MQENC_FLOAT_IEEE_REVERSED
```

Výchozí nastavení způsobí, že `WriteInt` zapíše celočíselné hodnoty typu little-endian a `ReadInt` pro čtení celého čísla typu little-endian. Pokud namísto toho nastavíte příznak `MQC.MQENC_INTEGER_NORMAL` , příkaz `WriteInt` vypíše celočíselné hodnoty typu big-endian a `ReadInt` přečte celé číslo typu big-endian.

Poznámka: Při převodu z formátu IEEE s pohyblivou řádovou čárkou na formát systému 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 platnosti zprávy je vhodné, aby byl správce front vyřazen. Pokud zpráva uvádí jeden z parametrů `MQC.MQRO_EXPIRATION` , sestava se vygeneruje, když je zpráva vyřazena. Předvolená hodnota je `MQC.MQEI_UNLIMITED`, což znamená, že zpráva nikdy nevyprší.

public int Feedback {get; set;}
Použijte `Feedback` se zprávou typu `MQC.MQMT_REPORT` , abyste označili povahu sestavy. Následující kódy zpětné vazby jsou definovány systémem:

- `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 žádná zpětná vazba není poskytována.

public string Format {get; set;}
Jméno formátu použité odesílatelem zprávy pro označení povahy dat ve zprávě příjemci. Můžete použít své vlastní názvy formátů, ale názvy začínající písmeny MQ mají význam, který je definován správcem front. Vestavěné formáty správce front jsou:

MQC.MQFMT_ADMIN
Zpráva s požadavkem/odpovědí příkazového serveru.

MQC.MQFMT_COMMAND_1
Zpráva odpovědi příkazu typu 1.

MQC.MQFMT_COMMAND_2
Zadejte zprávu s odpovědí příkazu typu 2.

MQC.MQFMT_DEAD_LETTER_HEADER
Hlavička nedoručitelného dopisu.

MQC.MQFMT_EVENT
Zpráva o události.

MQC.MQFMT_NONE
Chybí název formátu.

MQC.MQFMT_PCF
Uživatelem definovaná zpráva ve formátu programovatelného příkazu.

MQC.MQFMT_STRING
Zpráva sestávající pouze 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 níž náleží fyzická zpráva. Výchozí hodnota je MQC.MQGI_NONE.

public int MessageFlags {get; set;}
Příznaky, které řídí segmentaci a stav zprávy.

public byte[] MessageId {get; set;}

U volání MQQueue.Get toto pole uvádí identifikátor zprávy, která se má načíst. Za normálních okolností správce front vrátí první zprávu s identifikátorem zprávy a identifikátorem korelace, které odpovídají polím deskriptoru zpráv. Povolit každému identifikátoru zprávy použít speciální hodnotu MQC.MQMI_NONE.

U volání MQQueue.Put toto pole uvádí identifikátor zprávy, který se má použít. Pokud je zadán MQC.MQMI_NONE , správce front vygeneruje jedinečný identifikátor zprávy, když je zpráva vložena. Hodnota této členské proměnné se aktualizuje po vložení, čímž se označí identifikátor zprávy, který byl použit. 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 v rámci skupiny.

public int MessageType {get; set;}
Označuje typ zprávy. V systému jsou momentálně definovány tyto hodnoty:

- MQC.MQMT_DATAGRAM
- MQC.MQMT_REPLY
- MQC.MQMT_REPORT
- MQC.MQMT_REQUEST

Lze také použít hodnoty definované aplikací, 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, relativní ukazatel 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 reconnectable client, vrátí se kód příčiny MQRC_NONE k aplikaci, když je připojení úspěšné.

- MQC.MQPER_PERSISTENT

  Nastavíte-li tuto volbu v reconnectable client, vrátí se kód příčiny MQRC_CALL_INTERRUPTED do aplikace po úspěšném připojení.

- MQC.MQPER_PERSISTENCE_AS_Q_DEF

Výchozí hodnota je MQC.MQPER_PERSISTENCE_AS_Q_DEF, která bere perzistenci zprávy z výchozího atributu perzistence cílové fronty.

public int Priority {get; set;}
Priorita zpráv. Speciální hodnotu MQC.MQPRI_PRIORITY_AS_Q_DEF může být také nastavena v odchozí zprávě. Priorita pro zprávu se pak vezme z atributu výchozí priority cílové fronty. Výchozí hodnota je MQC.MQPRI_PRIORITY_AS_Q_DEF.

public int PropertyValidation {get; set;}
Uvádí, zda ověření vlastností probíhá, když je nastavena vlastnost zprávy. 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. Hodnota PutApplicationType může být definovaná systémem nebo uživatelem definovaná hodnota. Následující hodnoty jsou definovány systémem:

- 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 znamená, že ve zprávě nejsou obsaženy žádné informace o kontextu.

public DateTime PutDateTime {get; set;}
Datum a čas, kdy byla zpráva vložena.

public string ReplyToQueueManagerName {get; set;}
Název správce front, který má odeslat zprávy s odpovědí nebo sestavami. Výchozí hodnota je ""a správce front poskytuje název ReplyToQueueManagerName.

public string ReplyToQueueName {get; set;}
Název fronty zpráv, do které aplikace, která vydala požadavek na získání pro zprávu, odešle zprávy MQC.MQMT_REPLY a MQC.MQMT_REPORT . Výchozí hodnota ReplyToQueueName je "".

public int Report {get; set;}
Volbu Sestava použijte k uvedení voleb o zprávách a zprávách odpovědí:

- Určuje, zda jsou vyžadovány sestavy.
- Zda mají být data zprávy aplikace zahrnuta do sestav.
- Jak nastavit identifikátory zprávy a korelace v sestavě nebo v odpovědi.

Je možné požadovat libovolnou kombinaci těchto čtyř typů sestavy:

- Určete libovolnou kombinaci těchto čtyř typů sestav. Vyberte libovolnou ze tří voleb pro každý typ sestavy, v závislosti na tom, zda mají být data zprávy aplikace zahrnuta do zprávy sestavy.
  1. Potvrdit při příchodu
     - 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 jako ** v seznamu nejsou správci front produktu z/OS podporovány. Nepoužívejte je v případě, že aplikace pravděpodobně přistupuje ke správci front produktu z/OS bez ohledu na platformu, na které je aplikace spuštěna.
- Uveďte jednu z následujících možností, chcete-li řídit, jak se vygeneruje ID zprávy pro zprávu nebo zprávu odpovědi:
  - MQC.MQRO_NEW_MSG_ID
  - MQC.MQRO_PASS_MSG_ID
- Uveďte jednu z následujících možností, chcete-li řídit, jak má být korelační ID sestavy nebo zprávy odpovědi nastaveno:
  - MQC.MQRO_COPY_MSG_ID_TO_CORREL_ID
  - MQC.MQRO_PASS_CORREL_ID
- Určete jednu z následujících možností k řízení dispozice původní zprávy v případě, že ji nelze doručit do cílové fronty:

- MQC.MQRO_DEAD_LETTER_Q
- MQC.MQRO_DISCARD_MSG **

- Nejsou-li 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 z následujících možností, abyste si vyžádali, že přijímající aplikace odešle kladnou akci nebo zprávu s hlášením o negativní akci.
  - MQC.MQRO_PAN
  - MQC.MQRO_NAN

public int TotalMessageLength {get;}
Celkový počet bajtů ve zprávě, jak je uloženo ve frontě zpráv, ze které byla tato zpráva přijata.

public string UserId {get; set;}
UserId je část kontextu identity zprávy. Správce front obvykle tuto hodnotu poskytuje. Pokud máte oprávnění nastavit kontext identity, můžete tuto hodnotu přepsat.

public int Version {get; set;}
Verze struktury MQMD v použití.

Metody zpráv Read a Write

Metody Read a Write provádějí stejné funkce jako členy tříd BinaryReader a BinaryWriter v oboru názvů .NET System.IO . Viz MSDN pro plnou syntaxi jazyka a příklady použití. Metody čtou nebo zapisují z aktuální pozice ve vyrovnávací paměti zpráv. Přesunou aktuální pozici vpřed o počet přečtených nebo zapsaných bajtů.

Poznámka: Pokud data zprávy obsahují záhlaví MQRFH nebo MQRFH2 , je nutné ke čtení dat použít metodu produktu ReadBytes .

- Všechny metody throw IOException.
- Metody ReadFully automaticky mění velikost cílového pole byte nebo sbyte tak, aby se přesně vešly do zprávy. Mění se také velikost pole s hodnotou null.
- Read metody throw EndOfStreamException.
- WriteDecimal metody throw MQException.
- Převod metod ReadString, ReadLine a WriteString mezi Unicode a znakovou sadou zprávy; viz CharacterSet .
- Metody Decimal čtou a zapisují pakovaná desetinná čísla zakódovaná buď ve formátu big-endian, MQC.MQENC_DECIMAL_NORMAL, nebo little-endian MQC.MQENC_DECIMAL_REVERSE , podle hodnoty Kódování. 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 Kódování.
- Metody Int čtou a zapisují celočíselné hodnoty ve tvaru big-endian, MQC.MQENC_INTEGER_NORMALnebo little-endian, MQC.MQENC_INTEGER_REVERSED, formát, v závislosti na hodnotě Kódování. Celá čísla jsou podepsaná kromě přidání 2bajtového celočíselného typu bez znaménka. Celočíselné velikosti a typy .NET a IBM MQ jsou následující:

2 byte
`short`, `Int2`, `ushort`, `UInt2`

4 bajty
`int`, `Int4`

8 byte
`long`, `Int8`

- `WriteObject` přenáší třídu objektu, hodnoty jeho nepřechodných a nestatických polí a pole jeho supertypů do vyrovnávací paměti zpráv.
- `ReadObject` vytvoří objekt ze třídy objektu, podpis třídy a hodnoty jeho nepřechodných a nestatických polí a polí jeho supertypů.

Tabulka 844. Metody čtení a zápisu zpráv

Typ cíle	Podpisy metody

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 844. Metody čtení a zápisu zpráv (pokračování)

Typ cíle	Podpisy metody
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 844. Metody čtení a zápisu zpráv (pokračování)	
Typ cíle	**Podpisy metody**

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 `IOException`.

Vyřadí všechna data ve vyrovnávací paměti zpráv a nastaví offset dat zpět na nulu.

`public void ResizeBuffer(int size)`

Vyvolá `IOException`.

Pokyn k objektu `MQMessage` o velikosti vyrovnávací paměti, která může být vyžadována pro následné operace get. Pokud zpráva momentálně obsahuje data zprávy a nová velikost je menší než aktuální velikost, data zprávy budou zkrácena.

`public void Seek(int pos)`

Vyvolá `IOException`, `ArgumentOutOfRangeException`, `ArgumentException`.

Přesouvá kurzor na absolutní pozici ve vyrovnávací paměti zprávy udané příkazem *pos*. Následná čtení a zápisy fungují na této pozici ve vyrovnávací paměti.

`public int SkipBytes(int i)`

Vyvolá `IOException`, `EndOfStreamException`.

Přesouvá n bajtů vpřed v vyrovnávací paměti zpráv a vrací n, počet vynechaných bajtů.

Bloky metody `SkipBytes` , dokud nedojde k jedné z následujících událostí:

- Všechny bajty jsou přeskočeny.
- Byl zjištěn konec vyrovnávací paměti zpráv.
- Je vyvolána výjimka

Metody vlastností

`public void DeleteProperty(string name);`

Vyvolá `MQException`.

Odstraní vlastnost se zadaným názvem ze zprávy.

název
Název vlastnosti, která má být odstraněna.

public System.Collections.IEnumerator GetPropertyNames(string *name*)

Vyvolá MQException.

Vrací IEnumerator všech názvů vlastností odpovídajících zadanému názvu. Znak procenta '%' může být použit na konci názvu jako zástupný znak pro filtrování vlastností zprávy, shody na nule nebo více znaků, včetně období.

název
Název vlastnosti, se kterou se má shodovat.

Metody SetProperty a GetProperty

Všechny metody SetProperty a GetProperty generují MQException.

Metoda SetProperty třídy MQMessage .NET přidá novou vlastnost, pokud již vlastnost neexistuje. Pokud však tato vlastnost již existuje, je zadaná hodnota vlastnosti přidána na konec seznamu. Je-li více hodnot nastaveno na název vlastnosti pomocí SetProperty, volání GetProperty pro tento název vrátí tyto hodnoty sekvenčně v pořadí, ve kterém byly tyto hodnoty nastaveny.

Chování je stejné pro všechny zadané metody Set*Property a Get*Property , jako například GetLongProperty, SetLongProperty, GetBooleanProperty, SetBooleanProperty, GetStringPropertya SetStringProperty.

Tabulka 845. Metody SetProperty a GetProperty

Typ	Podpisy metody

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 845. Metody `SetProperty` a `GetProperty` (pokračování)

Typ	Podpisy metody

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 845. Metody SetProperty a GetProperty (pokračování)	
Typ	**Podpisy metody**

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 s prázdnou vyrovnávací pamětí zpráv.

Třída MQProcess.NET

Použijte MQProcess pro dotaz 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 1733
- “Konstruktory” na stránce 1734

Vlastnosti

Testuje se test produktu MQException při získávání vlastností.

public string ApplicationId {get;}

Získá znakový řetězec, který identifikuje aplikaci, která má být spuštěna. ApplicationId používá aplikace monitoru spouštěčů. Položka ApplicationId se odešle do inicializační fronty jako část zprávy spouštěče.

Výchozí hodnota je null.

public int ApplicationType {get;}

Označuje typ procesu, který má být spuštěn aplikací monitoru spouštěčů. Jsou definovány standardní typy, ale ostatní 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;}

Získává informace o prostředí aplikace, která má být spuštěna.

Výchozí hodnota je null.

public string UserData {get;}

Získá informace, které uživatel poskytl o aplikaci, která má být spuštěna.

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 MQException.

Přístup k procesu produktu IBM MQ ve správci front *qMgr* se dotáže na atributy 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 bitového operátoru OR, jsou:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

QueueManagerName
Název správce front, ve kterém je proces definován. Pokud je správce front shodný s tím, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou null.

AlternateUserid

Pokud je zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY v argumentu **openOptions** , určuje parametr *alternateUserId* alternativní ID uživatele použité ke kontrole autorizace dané akce. Není-li parametr MQOO_ALTERNATE_USER_AUTHORITY zadán, může být hodnota *alternateUserId* prázdná nebo null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, pokud není 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 MQException.

Přístup k procesu produktu IBM MQ v tomto správci front za účelem dotazu na atributy procesu.

processName
Název procesu, který se má otevřít.

openOptions
Volby, které řídí otevření procesu. Platné volby, které lze přidat nebo kombinovat pomocí bitového operátoru OR, jsou:

- MQC.MQOO_FAIL_IF_QUIESCING
- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

QueueManagerName
Název správce front, ve kterém je proces definován. Pokud je správce front shodný s tím, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou null.

AlternateUserid

Pokud je zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY v argumentu **openOptions** , určuje parametr *alternateUserId* alternativní ID uživatele použité ke kontrole autorizace dané akce. Není-li parametr MQOO_ALTERNATE_USER_AUTHORITY zadán, může být hodnota *alternateUserId* prázdná nebo null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, pokud není zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY .

Třída MQPropertyDescriptor.NET

Použijte MQPropertyDescriptor jako parametr k metodám 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 1735
- "Konstruktory" na stránce 1737

Vlastnosti

Testuje se test produktu MQException při získávání vlastností.

public int Context {get; set;}

Kontext zprávy, do kterého patří vlastnost. 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, je při načtení zprávy uložena vlastnost přidružená k kontextu uživatele. Následná metoda Put , která odkazuje na uložený kontext, může předat vlastnost do nové zprávy.

public int CopyOptions {get; set;}

Volba CopyOptions popisuje typ zprávy, do níž lze vlastnost zkopírovat.

Obdrží-li správce front zprávu obsahující vlastnost definovanou IBM MQ , kterou správce front rozpozná jako nesprávnou, opraví hodnotu pole CopyOptions jako chybnou.

Je možné zadat libovolnou kombinaci následujících voleb. Zkombinujte volby přidáním hodnot nebo pomocí bitového toku OR.

MQC.MQCOPY_ALL
Vlastnost se zkopíruje do všech typů následujících zpráv.

MQC.MQCOPY_FORWARD
Vlastnost se okopíruje do zprávy, která se předává.

MQC.MQCOPY_PUBLISH
Vlastnost se okopíruje do zprávy přijaté odběratelem při publikování zprávy.

MQC.MQCOPY_REPLY
Vlastnost se zkopíruje do zprávy odpovědi.

MQC.MQCOPY_REPORT
Vlastnost je zkopírována do zprávy sestavy.

MQC.MQCOPY_DEFAULT
Hodnota neoznačila žádné další volby kopírování, které byly zadány. Mezi vlastností a následujícími zprávami neexistuje žádný vztah. MQC.MQCOPY_DEFAULT je vždy vrácen pro vlastnosti deskriptoru zpráv.

MQC.MQCOPY_NONE
Stejné jako MQC.MQCOPY_DEFAULT

public int Options { set; }
Volba Volby standardně zobrazuje hodnotu CMQC.MQPD_NONE. Není možné nastavit žádnou jinou hodnotu.

public int Support { get; set; }

Nastavte volbu Podpora , abyste určili úroveň podpory požadovanou pro vlastnosti zprávy definované produktem IBM MQ. Podpora všech ostatních vlastností je volitelná. Je možné zadat libovolnou nebo žádnou z následujících hodnot

MQC.MQPD_SUPPORT_OPTIONAL
Vlastnost je přijata správcem front, i když není podporována. Vlastnost může být vyřazena, aby byla zpráva přetékat 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 zprávu vložíte do správce front, který nepodporuje vlastnost definovaná uživatelem IBM MQ, dojde k selhání metody. 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 zprávu vložíte do lokální fronty ve správci front, který nepodporuje vlastnost definovaná uživatelem IBM MQ, dojde k selhání metody. Vrací kód dokončení MQC.MQCC_FAILED a kód příčiny MQC.MQRC_UNSUPPORTED_PROPERTY.

Pokud je zpráva vložena do vzdáleného správce front, nebude provedena žádná kontrola.

Konstruktory

PropertyDescriptor();
Vytvořte deskriptor vlastnosti.

Třída MQPutMessageOptions.NET

Použijte MQPutMessageOptions k uvedení, jak se zprávy odesílají. Upravuje chování produktu 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 1737 “Konstruktory” na stránce 1739

Vlastnosti

Testuje se test produktu 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 kterých se mají brát informace o kontextu.

Počáteční hodnota tohoto pole je null.

public int InvalidDestCount {get;} *

Obecně řečeno, použitý pro distribuční seznamy, InvalidDestCount uvádí počet zpráv, které nemohly být odeslány do front v rozdělovníku. Tento počet zahrnuje fronty, které se nepodařilo otevřít, a také fronty, které byly úspěšně otevřeny, ale pro kterou selhala operace vložení.

Produkt .NET nepodporuje distribuční seznamy, ale při otevírání jedné fronty je nastaven parametr InvalidDestCount .

public int KnownDestCount {get;} *

Obecně se používá pro distribuční seznamy, KnownDestCount 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 akce MQDestination.put a MQQueueManager.put. Je možné zadat libovolnou z následujících hodnot nebo žádnou z následujících hodnot. Je-li požadována více než jedna volba, lze hodnoty přidat nebo zkombinovat pomocí bitového operátoru OR.

MQC.MQPMO_ASYNC_RESPONSE
Tato volba způsobí asynchronní volání obslužného programu MQDestination.put s některými daty odezvy.

MQC.MQPMO_DEFAULT_CONTEXT
Přidružte výchozí kontext ke zprávě.

MQC.MQPMO_FAIL_IF_QUIESCING
Selhat, pokud je správce front uváděn do klidového stavu.

MQC.MQPMO_LOGICAL_ORDER *
Vložení logických zpráv a segmentů ve skupinách zpráv do jejich logického pořadí.

Použijete-li volbu MQPMO_LOGICAL_ORDER v reconnectable client, vrátí se do aplikace kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

MQC.MQPMO_NEW_CORREL_ID *
Vygenerujte nové ID korelace pro každou odeslanou zprávu.

MQC.MQPMO_NEW_MSG_ID *
Generujte nové ID zprávy pro každou odeslanou zprávu.

MQC.MQPMO_NONE
Nejsou uvedeny žádné volby. Nepoužívejte s jinými možnostmi.

MQC.MQPMO_NO_CONTEXT
K této zprávě nemá být přidružen žádný kontext.

MQC.MQPMO_NO_SYNCPOINT
Vložit zprávu bez řízení synchronizačního bodu. Není-li určena volba pro řízení synchronizačního bodu, předpokládá se výchozí hodnota bez bodu synchronizace.

MQC.MQPMO_PASS_ALL_CONTEXT
Předejte všechny kontext z ovladače vstupní fronty.

MQC.MQPMO_PASS_IDENTITY_CONTEXT
Předat kontext identity z ovladače vstupní fronty.

MQC.MQPMO_RESPONSE_AS_Q_DEF

U volání MQDestination.put tato volba přijímá typ odezvy vložení z atributu DEFPRESP ve frontě.

U volání MQQueueManager.put tato volba způsobí, že volání bude provedeno 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 témat.

MQC.MQPMO_RETAIN

Odeslaná publikování má být uchována správcem front. Je-li tato volba použita a publikování nelze zadržet, zpráva se nepublikuje a volání selže s MQC.MQRC_PUT_NOT_RETAINED.

Vyžádejte si kopii této publikace po jejím publikování voláním metody MQSubscription.RequestPublicationUpdate . Uložené publikování je odesláno aplikacím, které vytvářejí odběr, aniž by byla nastavena volba MQC.MQSO_NEW_PUBLICATIONS_ONLY . Po přijetí zkontrolujte vlastnost zprávy MQIsRetained , pokud byla přijata, a zjistěte, zda se jedná o zachované publikování.

Pokud odběratel požaduje zachovaná publikování, může použitý odběr obsahovat zástupný znak v řetězci tématu. Pokud ve stromu témat obsahuje více zachovaných publikování, které odpovídají odběru, jsou všechny odeslány.

MQC.MQPMO_SET_ALL_CONTEXT
Nastavte veškerý 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
Jakékoli informace vyplněné v polích ReplyToQueueName a ReplyToQueueManager v publikování nejsou předány odběratelům. Je-li tato volba použita v kombinaci s volbou sestavy, která vyžaduje parametr ReplyToQueueName, volání selže s MQC.MQRC_MISSING_REPLY_TO_Q.

MQC.MQPMO_SYNCPOINT
Vložit zprávu s ovládacím prvkem synchronizačního bodu. Zpráva není viditelná mimo pracovní jednotku, dokud se jednotka práce nepotvrdí. Je-li jednotka práce zálohována, zpráva se odstraní.

public int RecordFields {get; set;}*

Informace o distribučních seznamech. Distribuční seznamy nejsou v produktu .NETpodporované.

public string ResolvedQueueManagerName {get;}

Výstupní pole nastavené správcem front na název správce front, který vlastní frontu určenou názvem vzdálené fronty. Položka ResolvedQueueManagerName se může lišit od názvu správce front, z něhož byla fronta zpřístupněna, je-li fronta vzdálenou 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 string ResolvedQueueName {get;}

Výstupní pole, které je nastaveno správcem front, do názvu fronty, na které je zpráva umístěna. Název 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. Ohlásí 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

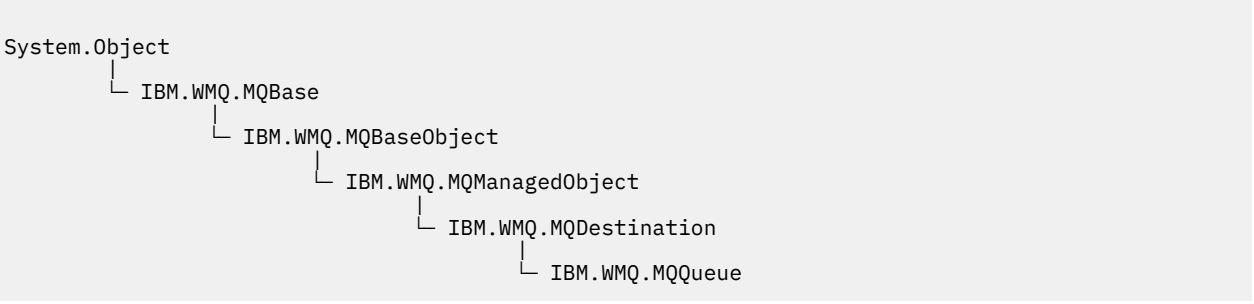
public MQPutMessageOptions();

Vytvořte nový objekt MQPutMessageOptions bez nastavení voleb a prázdnou hodnotu ResolvedQueueName a ResolvedQueueManagerName.

Třída MQQueue.NET

MQQueue slouží k odesílání a příjmu zpráv a k dotazování na atributy fronty IBM MQ . Vytvořte objekt MQQueue pomocí konstruktoru nebo metody MQQueueManager.AccessProcess .

Třída



```
public class IBM.WMQ.MQQueue extends IBM.WMQ.MQDestination;
```


- "Vlastnosti" na stránce 1740
- "Metody" na stránce 1742
- "Konstruktory" na stránce 1744

Vlastnosti

Testuje se test produktu `MQException` při získávání vlastností.

public int ClusterWorkLoadPriority {get;}
Uvádí prioritu fronty. Tento parametr je platný pouze pro lokální, vzdálené fronty a alias fronty.

public int ClusterWorkLoadRank {get;}
Uvádí očíslování pořadí fronty. Tento parametr je platný pouze pro lokální, vzdálené fronty a alias fronty.

public int ClusterWorkLoadUseQ {get;}
Určuje chování operace MQPUT v případě, že cílová fronta má lokální instanci a alespoň jednu vzdálenou instanci klastru. Tento parametr se nepoužije, má-li příkaz MQPUT pochází z kanálu klastru. Tento parametr je platný pouze pro lokální fronty.

public DateTime CreationDateTime {get;}
Datum a čas, kdy byla tato fronta vytvořena.

public int CurrentDepth {get;}
Získává počet zpráv, které jsou aktuálně ve frontě. Tato hodnota se inkrementuje během volání vložení a během odvolání k získání volání. Je to dekrementováno během operace bez procházení a během odvolání 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;}
Řídí, zda můžete vložit zprávy do této fronty nebo do tohoto tématu. Možné hodnoty jsou:

- `MQQA_PUT_INHIBITED`
- `MQQA_PUT_ALLOWED`

public int MaximumDepth {get;}
Maximální počet zpráv, které mohou být v dané frontě současně existovat. Pokus o vložení zprávy do fronty, která již obsahuje toto množství zpráv, se nezdaří 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 selže 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.
`PočetOpenInputCount` je celkový počet platných vstupních popisovačů známých lokálnímu správci front, nikoli pouze o obslužné rutiny vytvořené aplikací.

public int OpenOutputCount {get;}
Počet popisovačů, které jsou momentálně platné pro přidání zpráv do fronty.
`PočetOpenOutputCount` je celkový počet platných výstupních popisovačů známých pro lokálního správce front, nikoli pouze pro popisovače vytvořené aplikací.

public int QueueAccounting {get;}
Uvádí, zda můžete povolit shromažďování informací o účtování pro frontu.

public int QueueMonitoring {get;}
Určuje, 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;}
Určuje, zda se zprávy spouštěče zapisují do inicializační fronty, aby bylo možné spustit aplikaci 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, která dorazí do této fronty, způsobí, že se do inicializační fronty zapíše zpráva spouštěče. Maximální přípustná délka řetězce je dána `MQC.MQ_TRIGGER_DATA_LENGTH`.

public int TriggerDepth {get; set;}
Počet zpráv, které musí být ve frontě před zápisem zprávy spouštěče, když je typ spouštěče nastaven na `MQC.MQTT_DEPTH`.

public int TriggerMessagePriority {get; set;}
Priorita zpráv, pod kterou zprávy nepřispívají k generaci zpráv spouštěče. To znamená, že správce front tyto zprávy ignoruje při rozhodování, zda má být vygenerován spouštěč. Hodnota nula způsobí, že všechny zprávy přispívají k generaci zpráv spouštěče.

public int TriggerType {get; set;}
Podmínky, za kterých se zprávy spouštěče zapisují 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 MQException.

Získá zprávu z fronty.

Pokud dojde k selhání operace get, objekt MQMessage se nezmění. Pokud je úspěšný, jsou popisovač zprávy a části dat zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání do IBM MQ z konkrétního MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s čekáním, všechny ostatní podprocesy používající stejný MQQueueManager jsou blokovány od dalších volání IBM MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu IBM MQ současně, každý podproces musí vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá z polí v deskriptoru zpráv jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Reconnectable klient vrací kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení, pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby ovládající akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při konverzi z jednobajtových znakových kódů do dvoubajtových kódů. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez konverze.

Není-li parametr *getMessageOptions* zadán, bude použita volba zprávy MQGMO_NOWAIT.

Použijete-li volbu MQGMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Je-li zpráva ve frontě větší než tato velikost, nastane jedna ze dvou situací:

- Je-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven v objektu MQGetMessageOptions , zpráva je vyplněna co nejvíce informací o zprávě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED .
- Není-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven, zůstává zpráva ve frontě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED .

Není-li parametr *MaxMsgSize* zadán, bude načtena celá zpráva.

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Vyvolá MQException.

Přepne zprávu do fronty.

Úpravy objektu MQMessage po dokončení volání operace Put nemají vliv na skutečnou zprávu ve frontě IBM MQ nebo v tématu publikování.

Produkt Put aktualizuje vlastnosti MessageId a CorrelationId objektu MQMessage a nemaže data zprávy. Další volání Put nebo Get odkazují na aktualizované informace v objektu MQMessage . Například v následujícím úseku kódu bude první zpráva obsahovat a a druhý ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zpráv a zpráva, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zpráv bezprostředně po dokončení této metody jsou hodnotami, které byly vloženy do fronty nebo publikovány do tématu.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- MQRC_CALL_INTERRUPTED je-li připojení přerušeno při spuštění volání vložení na trvalou zprávu a opětovné navázání spojení je úspěšné.
- MQRC_NONE , je-li připojení úspěšné při spuštění volání vložení na netrvalou zprávu (viz Obnova aplikace).

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu MQPutMessageOptons .

Použijete-li volbu MQPMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Pokud chcete do fronty vložit jednu zprávu, použijte objekt MQQueueManager.Put pro zjednodušení a výkon. Pro tento objekt byste měli mít objekt MQQueue .

public void PutForwardMessage(MQMessage *message*);
public void PutForwardMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);

Vyvolá se MQException

Vložte zprávu předávaný do fronty, kde *message* je původní zpráva.

zpráva
Objekt MQMessage obsahující data deskriptoru zpráv a zpráva, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zpráv bezprostředně po dokončení této metody jsou hodnotami, které byly vloženy do fronty nebo publikovány do tématu.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- MQRC_CALL_INTERRUPTED je-li připojení přerušeno při spuštění volání vložení na trvalou zprávu a opětovné navázání spojení je úspěšné.
- MQRC_NONE , je-li připojení úspěšné při spuštění volání vložení na netrvalou zprávu (viz Obnova aplikace).

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu MQPutMessageOptons .

Použijete-li volbu MQPMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

public void PutReplyMessage(MQMessage *message*)
public void PutReplyMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Vyvolá MQException.

Vložte zprávu odpovědi do fronty, kde *message* je původní zpráva.

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá z polí v deskriptoru zpráv jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry `MessageId` a `CorrelationId` byly nastaveny podle potřeby.

Reconnectable klient vrací kód příčiny `MQRC_BACKED_OUT` po úspěšném opětovném připojení, pro zprávy přijaté pod `MQGM_SYNCPOINT`.

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu `MQPutMessageOptons` .

Použijete-li volbu `MQPMO_LOGICAL_ORDER` v reconnectable client, vrátí se kód příčiny `MQRC_RECONNECT_INCOMPATIBLE` .

public void PutReportMessage(MQMessage *message*)
public void PutReportMessage(MQMessage *message*, MQPutMessageOptions *putMessageOptions*)

Vyvolá `MQException`.

Vložte zprávu do fronty do fronty, kde *message* je původní zpráva.

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá z polí v deskriptoru zpráv jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry `MessageId` a `CorrelationId` byly nastaveny podle potřeby.

Reconnectable klient vrací kód příčiny `MQRC_BACKED_OUT` po úspěšném opětovném připojení, pro zprávy přijaté pod `MQGM_SYNCPOINT`.

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu `MQPutMessageOptons` .

Použijete-li volbu `MQPMO_LOGICAL_ORDER` v reconnectable client, 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 `MQException`.

Přistupuje k frontě v tomto správci front.

Můžete získat nebo procházet zprávy, vložit zprávy, dotázat se na atributy fronty nebo nastavit atributy fronty. Je-li uvedená fronta modelová fronta, vytvoří se dynamická lokální fronta. Dotažte se na atribut `name` výsledného objektu `MQQueue` , abyste zjistili název dynamické fronty.

queueName
Název fronty, která má být otevřena.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Validovat s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se k určitému místu určení.

MQC.MQOO_BIND_ON_OPEN
Svázat popisovač do cíle při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zprávy.

MQC.MQOO_FAIL_IF_QUIESCING
Selhat, pokud je správce front uváděn do klidového stavu.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřeno pro získání zpráv s použitím výchozí hodnoty definované frontou.

MQC.MQOO_INPUT_SHARED
Otevřeno pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřeno pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřeno pro dotaz-nezbytné, pokud chcete dotázat se na vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vkládání zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předávání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předávání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načítání zprávy.

MQC.MQOO_SET
Chcete-li nastavit vlastnosti, otevřete jej pro nastavení atributů.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavit veškerý kontext.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavit kontext identity.

QueueManagerName
Název správce front, ve kterém je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt `MQQueueManager` připojen.

Název dynamicQueue
dynamicQueueName je ignorován, pokud `queueName` neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* udává název dynamické fronty, která má být vytvořena. Prázdný název nebo název s hodnotou null není platný, pokud `queueName` uvádí název modelové fronty. Pokud je poslední neprázdný znak v názvu hvězdička, `*`, nahradí správce front hvězdičku řetězcem znaků. Znaky garantují, že název generovaný pro frontu je jedinečný v tomto správci front.

AlternateUserid
Je-li `MQC.MQOO_ALTERNATE_USER_AUTHORITY` zadán v parametru `openOptions` , *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole oprávnění pro otevření. Není-li parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` zadán, může být *alternateUserId* ponechán prázdný nebo má hodnotu null.

public MQQueue(MQQueueManager *queueManager*, string *queueName*, int *openOptions*, string *queueManagerName*, string *dynamicQueueName*, string *alternateUserId*);

Vyvolá `MQException`.

Přistupuje k frontě v systému `queueManager`.

Můžete získat nebo procházet zprávy, vložit zprávy, dotázat se na atributy fronty nebo nastavit atributy fronty. Je-li uvedená fronta modelová fronta, vytvoří se dynamická lokální fronta. Dotažte se na atribut name výsledného objektu MQQueue , abyste zjistili název dynamické fronty.

queueManager
Správce front pro přístup k frontě.

queueName
Název fronty, která má být otevřena.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Validovat s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se k určitému místu určení.

MQC.MQOO_BIND_ON_OPEN
Svázat popisovač do cíle při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zprávy.

MQC.MQOO_FAIL_IF_QUIESCING
Selhat, pokud je správce front uváděn do klidového stavu.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřeno pro získání zpráv s použitím výchozí hodnoty definované frontou.

MQC.MQOO_INPUT_SHARED
Otevřeno pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřeno pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřeno pro dotaz-nezbytné, pokud chcete dotázat se na vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vkládání zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předávání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předávání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načítání zprávy.

MQC.MQOO_SET
Chcete-li nastavit vlastnosti, otevřete jej pro nastavení atributů.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavit veškerý kontext.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavit kontext identity.

QueueManagerName
Název správce front, ve kterém je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt MQQueueManager připojen.

Název dynamicQueue
dynamicQueueName je ignorován, pokud queueName neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* udává název dynamické fronty, která má být vytvořena. Prázdný název nebo název s hodnotou null není platný, pokud queueName uvádí název modelové fronty. Pokud je

poslední neprázdný znak v názvu hvězdička, *, nahradí správce front hvězdičku řetězcem znaků. Znaky garantují, že název generovaný pro frontu je jedinečný v tomto správci front.

AlternateUserid
Je-li MQC.MQOO_ALTERNATE_USER_AUTHORITY zadán v parametru openOptions , *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole oprávnění pro otevření. Není-li parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY zadán, může být *alternateUserId* ponechán prázdný nebo má hodnotu null.

Třída MQQueueManager.NET

Pomocí produktu MQQueueManager se můžete připojit k objektům správce front a přistupovat k objektům správce front. Rovněž kontroluje transakce. Konstruktor produktu MQQueueManager vytvoří buď 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 1747
- “Metody” na stránce 1750
- “Konstruktory” na stránce 1756

Vlastnosti

Testuje se test produktu MQException při získávání vlastností.

public int AccountingConnOverride {get;}
Určuje, zda aplikace mohou přepsat nastavení hodnot evidence evidence a evidence front MQI .

public int AccountingInterval {get;}
Jak dlouho před zápisem intermediačních záznamů evidence (v sekundách).

public int ActivityRecording {get;}
Ovládá generování sestav aktivity.

public int AdoptNewMCACheck {get;}
Určuje, které prvky se kontrolují při určení, zda je agent MCA přijat při zjištění nového příchozího kanálu. Aby bylo možné přijmout, název MCA musí odpovídat názvu aktivního agenta MCA.

public int AdoptNewMCAInterval {get;}
Doba (v sekundách), po kterou nový kanál čeká na ukončení tohoto osiřelého kanálu.

public int AdoptNewMCAType {get;}
Určuje, zda má být osiřelá instance MCA přejata (restartována), když je zjištěn nový příchozí požadavek na kanál odpovídající hodnotě AdoptNewMCACheck.

public int BridgeEvent {get;}
Zda se generují události mostu IMS .

public int ChannelEvent {get;}
Zda se generují události kanálu.

public int ChannelInitiatorControl {get;}
Určuje, zda má být inicializátor kanálu spuštěn automaticky při spuštění správce front.

public int ChannelInitiatorAdapters {get;}
Počet podúloh adaptéru pro zpracování volání produktu IBM MQ .

public int ChannelInitiatorDispatchers {get;}
Počet dispečerů, který má být použit pro inicializátor kanálu.

public int ChannelInitiatorTraceAutoStart {get;}
Určuje, zda se trasování inicializátoru kanálu spouští automaticky.

public int ChannelInitiatorTraceTableSize {get;}
Velikost (v megabajtech) datového prostoru pro trasování inicializátoru kanálu.

public int ChannelMonitoring {get;}
Určuje, zda je použito monitorování kanálu.

public int ChannelStatistics {get;}
Ovládá shromažďování statistických dat kanály.

public int CharacterSet {get;}
Vrací identifikátor kódované znakové sady (CCSID) správce front. Soubor `CharacterSet` je používán správcem front pro všechna pole znakových řetězců v rozhraní API.

public int ClusterSenderMonitoring {get;}
Ovládá shromažďování online monitorovacích dat 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;}
Vrátí 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 funkční úrovni, závisí na platformě. Na konkrétní platformě se můžete spolehnout na všechny správce front, které podporují funkce na nejnižší funkční úrovni společné pro všechny správce front.

public int CommandLevel {get;}
Určuje, zda má být příkazový server spuštěn 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;}
Vrátí hodnotu atributu `isConnected`.

Pokud je hodnota nastavena na true, bylo vytvořeno připojení ke správci front a není známo, že by došlo k přerušení spojení. Volání IsConnected se aktivně nepokusí o přístup ke správci front, takže je možné, že fyzická konektivita může přerušit, ale IsConnected může stále vracet hodnotu true. Stav IsConnected se aktualizuje pouze tehdy, když je aktivita, například vložení zprávy, získání zprávy, provedena na správci front.

Je-li hodnota false, připojení ke správci front nebylo provedeno nebo bylo přerušeno nebo došlo k jeho odpojení.

public int KeepAlive {get;}
Uvádí, zda se použije funkce TCP KEEPALIVE pro kontrolu toho, zda je druhý konec připojení stále dostupný. Pokud není k dispozici, kanál se zavře.

public int ListenerTimer {get;}
The time interval, in seconds, between attempts by IBM MQ to restart the listener after an APPC or TCP/IP failure.

public int LoggerEvent {get;}
Určuje, 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, vydá se příkaz z/OS SET APPC=xx.

public string LUGroupName {get; z/os}
Generický název LU, který má být použit modulem listener LU 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front.

public string LUName {get;}
Název jednotky 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 aktuální v libovolném okamžiku (včetně kanálů připojení serveru s připojenými klienty).

public int MaximumLU62Channels {get;}
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, které používají přenosový protokol LU 6.2 .

public int MaximumMessageLength {get;}
Vrací maximální délku zprávy (v bajtech), kterou může zpracovat správce front. Žádná fronta nemůže být definována s maximální délkou zprávy větší než `MaximumMessageLength`.

public int MaximumPriority {get;}
Vrátí maximální prioritu zpráv podporovanou správcem front. Priority jsou v rozsahu od nuly (nejnižší) k této hodnotě. Pokud zavoláte tuto metodu po odpojení od správce front, zobrazí se `MQException` .

public int MaximumTCPChannels {get;}
Maximální počet kanálů, které mohou být aktuální, nebo klientů, kteří mohou být připojeni, které 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, které mají být použity při vázání odchozích kanálů.

public int OutboundPortMin {get;}
Minimální hodnota v rozsahu čísel portů, které mají být použity při vázání odchozích kanálů.

public int QueueAccounting {get;}
Zda mají být data evidence třídy 3 (evidence na úrovni vlákna a na úrovni fronty) použita 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 příjem dat, včetně synchronizačních signálů, od svého partnera, než se vrátí do neaktivního stavu.

public int ReceiveTimeoutMin {get;}
Minimální doba, po kterou kanál TCP/IP čeká na příjem dat, včetně synchronizačních signálů, od svého partnera, než se vrátí do neaktivního stavu.

public int ReceiveTimeoutType {get;}
Kvalifikátor, který má být použit pro hodnotu v parametru `ReceiveTimeout`.

public int SharedQueueQueueManagerName {get;}
Určuje, jak doručit zprávy do sdílené fronty. Pokud příkaz put určuje jiného správce front ze stejné skupiny sdílení front jako cílového správce front, bude tato zpráva doručena dvěma způsoby:

MQC.MQSQQM_USE
Zprávy jsou doručovány správci front objektu před tím, než jsou vloženy do sdílené fronty.

MQCMQSQQM_IGNORE
Zprávy jsou vloženy přímo do sdílené fronty.

public int SSLEvent {get;}
Zda jsou generovány události TLS.

public int SSLFips {get;}
Zda se mají použít pouze algoritmy certifikované podle standardu FIPS, pokud je šifrování prováděno v produktu IBM MQ, nikoli kryptografického hardwaru.

public int SSLKeyResetCount {get;}
Označuje počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán tajný klíč.

public int ClusterSenderStatistics {get;}
Určuje interval (v minutách) mezi následnými shromažďováními statistických údajů.

public int SyncpointAvailability {get;}
Označuje, zda správce front podporuje jednotky práce a synchronizační body s metodami `MQQueue.get` a `MQQueue.put` .

public string TCPName {get;}
Název pouze jednoho nebo výchozího systému TCP/IP, který má být použit, v závislosti na hodnotě TCPStackType.

public int TCPStackType {get;}
Uvádí, zda iniciátor kanálu používá pouze adresní prostor TCP/IP, který je uveden v `TCPName`. Inicializátor kanálu se může také připojit k libovolné adrese TCP/IP.

public int TraceRouteRecording {get;}
Ovládá záznam informací o trasování přenosové cesty.

Metody

public MQProcess AccessProcess(string *processName*, int *openOptions*);
public MQProcess AccessProcess(string *processName*, int *openOptions*, string *queueManagerName*, string *alternateUserId*);

Vyvolá `MQException`.

Přístup k procesu produktu IBM MQ v tomto správci front za účelem dotazu na atributy procesu.

processName
Název procesu, který se má otevřít.

openOptions
Volby, které řídí otevření procesu. Platné volby, které lze přidat nebo kombinovat pomocí bitového operátoru OR, jsou:

- `MQC.MQOO_FAIL_IF_QUIESCING`

- MQC.MQOO_INQUIRE
- MQC.MQOO_SET
- MQC.MQOO_ALTERNATE_USER_AUTHORITY

QueueManagerName
Název správce front, ve kterém je proces definován. Pokud je správce front shodný s tím, ke kterému proces přistupuje, můžete ponechat prázdný název správce front nebo název správce front s hodnotou null.

AlternateUserid

Pokud je zadán parametr MQC.MQOO_ALTERNATE_USER_AUTHORITY v argumentu **openOptions** , určuje parametr *alternateUserId* alternativní ID uživatele použité ke kontrole autorizace dané akce. Není-li parametr MQOO_ALTERNATE_USER_AUTHORITY zadán, může být hodnota *alternateUserId* prázdná nebo null.

Výchozí oprávnění uživatele se používá pro připojení ke správci front, pokud není 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 MQException.

Přistupuje k frontě v tomto správci front.

Můžete získat nebo procházet zprávy, vložit zprávy, dotázat se na atributy fronty nebo nastavit atributy fronty. Je-li uvedená fronta modelová fronta, vytvoří se dynamická lokální fronta. Dotažte se na atribut name výsledného objektu MQQueue , abyste zjistili název dynamické fronty.

queueName
Název fronty, která má být otevřena.

openOptions
Volby, které řídí otevření fronty.

MQC.MQOO_ALTERNATE_USER_AUTHORITY
Validovat s uvedeným identifikátorem uživatele.

MQC.MQOO_BIND_AS_QDEF
Použít výchozí vazbu pro frontu.

MQC.MQOO_BIND_NOT_FIXED
Nepřipojujte se k určitému místu určení.

MQC.MQOO_BIND_ON_OPEN
Svázat popisovač do cíle při otevření fronty.

MQC.MQOO_BROWSE
Otevřít pro procházení zprávy.

MQC.MQOO_FAIL_IF_QUIESCING
Selhat, pokud je správce front uváděn do klidového stavu.

MQC.MQOO_INPUT_AS_Q_DEF
Otevřeno pro získání zpráv s použitím výchozí hodnoty definované frontou.

MQC.MQOO_INPUT_SHARED
Otevřeno pro získání zpráv se sdíleným přístupem.

MQC.MQOO_INPUT_EXCLUSIVE
Otevřeno pro získání zpráv s výlučným přístupem.

MQC.MQOO_INQUIRE
Otevřeno pro dotaz-nezbytné, pokud chcete dotázat se na vlastnosti.

MQC.MQOO_OUTPUT
Otevřít pro vkládání zpráv.

MQC.MQOO_PASS_ALL_CONTEXT
Povolit předávání všech kontextů.

MQC.MQOO_PASS_IDENTITY_CONTEXT
Povolit předávání kontextu identity.

MQC.MQOO_SAVE_ALL_CONTEXT
Uložit kontext při načítání zprávy.

MQC.MQOO_SET
Chcete-li nastavit vlastnosti, otevřete jej pro nastavení atributů.

MQC.MQOO_SET_ALL_CONTEXT
Umožňuje nastavit veškerý kontext.

MQC.MQOO_SET_IDENTITY_CONTEXT
Umožňuje nastavit kontext identity.

QueueManagerName
Název správce front, ve kterém je fronta definována. Název, který je zcela prázdný nebo má hodnotu null, označuje správce front, ke kterému je objekt `MQQueueManager` připojen.

Název dynamicQueue
dynamicQueueName je ignorován, pokud `queueName` neuvádí název modelové fronty. Pokud ano, *dynamicQueueName* udává název dynamické fronty, která má být vytvořena. Prázdný název nebo název s hodnotou null není platný, pokud `queueName` uvádí název modelové fronty. Pokud je poslední neprázdný znak v názvu hvězdička, `*`, nahradí správce front hvězdičku řetězcem znaků. Znaky garantují, že název generovaný pro frontu je jedinečný v tomto správci front.

AlternateUserid
Je-li `MQC.MQOO_ALTERNATE_USER_AUTHORITY` zadán v parametru `openOptions` , *alternateUserId* uvádí alternativní identifikátor uživatele, který se používá ke kontrole oprávnění pro otevření. Není-li parametr `MQC.MQOO_ALTERNATE_USER_AUTHORITY` zadán, může být *alternateUserId* ponechán prázdný nebo má 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 produktu `MQTopic` úzce souvisejí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu odkazuje `topicObject` na objekt administrativního tématu. Konstruktor produktu `MQTopic` získá řetězec tématu z objektu tématu a spojí jej s názvem `topicName` a vytvoří název tématu. Buď `topicObject` , nebo `topicName` mohou mít hodnotu null. Název tématu se shoduje se stromem témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v souboru `topicObject`.

Témata přidružená k objektu `MQTopic` jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován pomocí objektu administrativního tématu identifikovaného produktem

topicObject. Druhý řetězec tématu je *topicString*. Výsledný řetězec tématu přidružený k objektu MQTopic může identifikovat více témat, včetně 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 publikací o tématech. Pokud chcete publikovat a odebírat stejné téma, 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. Předáte-li frontu jako objekt MQDestination , předpokládá se neřízený odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní s tím, že odběr je spravován nebo nespravovaný.

cíl
destination je instancí MQQueue . Poskytnutím *destination*se MQTopic otevře jako nespravovaný odběr. Publikace na téma jsou doručeny do fronty, k němuž se přistupuje jako k produktu *destination*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *topicName* je zřetězen s řetězcem tématu definovaným v objektu administrativního tématu produktu *topicObject* . Hodnotu *topicName* lze nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v produktu *topicObject*.

topicObject
Na vstupu *topicObject* je název objektu tématu, který obsahuje řetězec tématu, který tvoří první část názvu tématu. Řetězec tématu v produktu *topicObject* je zřetězený s *topicName*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Na výstupu obsahuje *topicObject* název objektu administračních témat, který je nejpřesnější shodou ve stromu témat k tématu identifikovanému řetězcem tématu.

openAs
Přistupte k tématu, které chcete publikovat nebo odebírat. Tento 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 vyžadována více než jedna volba, přidejte hodnoty dohromady nebo zkombinujte hodnoty voleb pomocí bitového operátoru OR .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace k dokončení operace. Musíte zadat *alternateUserId*, pokud je v parametru voleb nastaven buď MQC.MQOO_ALTERNATE_USER_AUTHORITY nebo MQC.MQSO_ALTERNATE_USER_AUTHORITY .

subscriptionName
subscriptionName je požadován, pokud jsou poskytnuty volby MQC.MQSO_DURABLE nebo MQC.MQSO_ALTER . V obou případech je MQTopic implicitně otevřeno pro odběr. Pokud je nastavena hodnota MQC.MQSO_DURABLE a existuje odběr nebo pokud je nastaven produkt MQC.MQSO_ALTER a odběr neexistuje, dojde k výjimce.

vlastnosti
Nastavte některou ze speciálních vlastností odběru uvedených pomocí hašovací tabulky. Uvedené záznamy v transformační tabulce jsou aktualizovány s výstupními hodnotami. Do transformační tabulky se nepřidají záznamy pro hlášení výstupních hodnot.

- 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 se MQException

Vrací objekt MQAsyncStatus , který představuje asynchronní aktivitu pro připojení správce front.

public void Backout();

Vyvolá MQException.

V rámci synchronizačního bodu od posledního bodu synchronizace došlo k vrácení jakýchkoli zpráv, které byly načteny nebo zapsány do synchronizačních bodů.

Zprávy, které byly zapsány pomocí sady příznaků MQC.MQPMO_SYNCPOINT , jsou odstraněny z front. Zprávy načtené s parametrem MQC.MQGMO_SYNCPOINT jsou obnoveny ve frontách, ze kterých pocházejí. Pokud jsou zprávy trvalé, jsou změny protokolovány.

V případě reconnectable clients se kód příčiny MQRC_NONE vrátí klientovi po úspěšném připojení.

public void Begin();

Vyvolá MQException.

Produkt Begin je podporován pouze v režimu vazeb serveru. Spustí globální pracovní jednotku.

public void Commit();

Vyvolá MQException.

Potvrďte všechny zprávy, které byly načteny nebo zapsány v rámci synchronizačního bodu od posledního bodu synchronizace.

Zprávy zapsané pomocí parametru MQC.MQPMO_SYNCPOINT jsou zpřístupněny ostatním aplikacím. Zprávy načtené pomocí příznaku MQC.MQGMO_SYNCPOINT se odstraní. Pokud jsou zprávy trvalé, jsou změny protokolovány.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- MQRC_CALL_INTERRUPTED , pokud je připojení ztraceno při provádění volání operace commit.
- MQRC_BACKED_OUT je-li volání potvrzení vydáno po opětovném připojení.

Disconnect();

Vyvolá MQException.

Zavřete připojení ke správci front. Ke všem objektům, k nimž se přistupuje v tomto správci front, není pro tuto aplikaci k dispozici přístup. Chcete-li znovu získat přístup k objektům, vytvořte objekt MQQueueManager .

Obecně platí, že každá práce provedená jako součást transakce je potvrzena. Avšak pokud je jednotka práce spravována produktem .NET, může být jednotka práce 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 `MQException`.

Umístí jednu zprávu do fronty nebo tématu, aniž byste nejprve vytvořili objekt `MQQueue` nebo `MQTopic` .

queueName
Název fronty, do které má být zpráva umístěna.

destinationName
Název cílového objektu. Je to buď fronta, 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` , tento parametr se ignoruje.

Je-li fronta modelová fronta a vyřešený název správce front není tento správce front, je vyhozena `MQException` .

topicString

topicString je zkombinován s názvem tématu v objektu tématu *destinationName* .

topicString je ignorován, pokud *destinationName* je fronta.

zpráva
Zpráva, která má být odeslána. Zpráva je vstupní/výstupní objekt.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- `MQRC_CALL_INTERRUPTED` , pokud je připojení přerušeno při provádění volání vložení na trvalou zprávu.
- `MQRC_NONE` , je-li připojení úspěšné při provádění volání příkazu Put pro dočasnou zprávu (viz téma Obnova aplikace).

Volby putMessage

Volby ovládající akce put.

Vynecháte-li volbu *putMessageOptions*, vytvoří se výchozí instance produktu *putMessageOptions* . *putMessageOptions* je vstupní/výstupní objekt.

Použijete-li volbu `MQPMO_LOGICAL_ORDER` v reconnectable client, vrátí se kód příčiny `MQRC_RECONNECT_INCOMPATIBLE` .

AlternateUserid

Uvádí alternativní identifikátor uživatele, který se používá ke kontrole autorizace při zařazení zprávy do fronty.

Pokud nenastavíte `MQC.MQOO_ALTERNATE_USER_AUTHORITY` v *putMessageOptions*, můžete vynechat *alternateUserId* . Pokud jste nastavili `MQC.MQOO_ALTERNATE_USER_AUTHORITY`, musíte také nastavit *alternateUserId*. *alternateUserId* nemá účinek, 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 `MQException`.

Vytvoří připojení ke správci front. Vyberte mezi vytvořením připojení klienta nebo připojení k serveru.

Chcete-li se připojit ke správci front, musíte mít při pokusu o připojení ke správci front oprávnění k dotazům (`inq`). Bez dotazovacího oprávnění se pokus o připojení nezdaří.

Připojení klienta se vytvoří, je-li splněna jedna z následujících podmínek:

1. *channel* nebo *connName* jsou uvedeny v konstruktoru.
2. *HostName*, *Port*, nebo *Channel* jsou zadány v *properties*.
3. Jsou zadány volby *MQEnvironment.HostName*, *MQEnvironment.Port*nebo *MQEnvironment.Channel* .

Hodnoty vlastností připojení se standardně zobrazují v uvedeném 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, k němuž se má připojit.

Chcete-li vytvořit výchozí výběr správce front, vynechte tento parametr nebo ponechte hodnotu null nebo je prázdný. Výchozí připojení správce front na serveru je předvoleným správci front na serveru. Výchozí připojení správce front v připojení klienta je ke správci front, ke kterému je modul listener připojen.

volby
Uveďte volby připojení `MQCNO` . Hodnoty musí být použitelné na typ vytvářeného připojení. Například, pokud uvedete následující vlastnosti připojení serveru pro připojení klienta, dojde k vyvolání `MQException` .

- `MQC.MQCNO_FASTPATH_BINDING`
- `MQC.MQCNO_STANDARD_BINDING`

vlastnosti

Parametr vlastností vezme řadu párů klíč/hodnota, které přepíší vlastnosti nastavené `MQEnvironment` ; viz příklad, “Potlačit vlastnosti MQEnvironment” na stránce 1759. Převážit lze následující vlastnosti:

- `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*).

Můžete zadat seznam *názvů hostitelů* a *portů* jako argument konstruktoru `MQQueueManager (String queueManagerName, Hashtable properties)` pomocí 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 zpracuje seznam názvů připojení v uvedeném pořadí. Pokud se pokus o připojení k prvnímu názvu hostitele a portu nezdaří, pokusí se o připojení k druhému páru atributů. Klient tento proces opakuje tak dlouho, dokud nebude vytvořeno úspěšné připojení nebo dokud nebude seznam vyčerpán. Je-li seznam vyčerpán, je do klientské aplikace vrácen odpovídající kód příčiny a kód dokončení.

Není-li pro název připojení zadáno číslo portu, je výchozí port (nakonfigurovaný v produktu `mqclient.ini`). se používá.

Nastavit seznam spojení

Seznam připojení můžete nastavit tak, že při nastavení voleb automatického připojení klienta použijete následující metody:

Nastavení seznamu připojení prostřednictvím produktu MQSERVER
Seznam spojení můžete nastavit pomocí příkazového řádku.

Na příkazový řádek zadejte 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 produktu MQSERVER, nenastavujte ji v aplikaci.

Nastavíte-li v aplikaci seznam připojení, aplikace přepíše všechny nastavené hodnoty v proměnné prostředí MQSERVER.

Nastavit seznam připojení pomocí aplikace
Seznam připojení v aplikaci můžete 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 uvedete 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 spojení můžete přímo změnit v souboru `app.config` .

Nastavit seznam připojení prostřednictvím `MQEnvironment`
Chcete-li nastavit seznam připojení prostřednictvím produktu `MQEnvironment`, použijte vlastnost *ConnectionName* .

```
MQEnvironment.ConnectionName = "fred.mq.com(4288),"alex.mq.com(5211);
```

Vlastnost *ConnectionName* přepíše název hostitele a vlastnosti portu nastavené v produktu `MQEnvironment`.

Vytvořit připojení klienta

Následující příklad ukazuje, jak vytvořit připojení klienta ke správci front. Před vytvořením nového objektu `MQQueueManager` můžete vytvořit připojení klienta tak, že nastavíte proměnné `MQEnvironment` .

```
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 použitím svého ID uživatele a hesla definovaného v transformační 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. Přepsání 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`

Chcete-li požadovat, aby zachované publikace byly odeslány odběrateli, použijte příkaz `MQSubscription` . `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 1760
- “Metody” na stránce 1760
- “Konstruktory” na stránce 1760

Vlastnosti

Přístup k vlastnostem odběru pomocí třídy `MQManagedObject` ; viz “Vlastnosti” na stránce 1719.

Metody

Přistupte k odběru metod `Inquire`, `Set` a `Get` odběru přístupu pomocí třídy `MQManagedObject` , viz “Metody” na stránce 1720.

`public int RequestPublicationUpdate(int options);`

Vyvolá `MQException`.

Vyžádejte si aktualizovanou publikaci pro aktuální téma. Má-li správce front zachované publikace pro dané téma, jsou odeslány odběrateli.

Před voláním funkce `RequestPublicationUpdate`otevřete téma pro odběr a získejte objekt `MQSubscription` .

Zpravidla otevřete odběr pomocí volby `MQC.MQSO_PUBLICATIONS_ON_REQUEST` . Pokud v řetězci tématu nejsou obsaženy žádné zástupné znaky, bude jako výsledek tohoto volání odeslána pouze jedna publikace. Pokud řetězec tématu obsahuje zástupné znaky, může být odesláno mnoho publikací. Tato metoda vrací počet zachovaných publikování, které jsou odeslány do fronty odběru. Neexistuje žádná záruka, že bude přijato mnoho publikací, zvláště pokud se jedná o přechodné zprávy.

volby

`MQC.MQSRO_FAIL_IF_QUIESCING`
Metoda selže, pokud se správce front nachází v klidovém stavu. V systému z/OSpro aplikaci CICS nebo IMS produkt `MQC.MQSRO_FAIL_IF_QUIESCING` také vynutí selhání metody v případě, že se připojení nachází v klidovém stavu.

`MQC.MQSRO_NONE`
Nejsou zadány žádné volby.

Konstruktory

Konstruktor Public neexistuje.

Objekt `MQSubscription` je vrácen ve vlastnosti `SubscriptionReference` objektu `MQTopic` , který je otevřen pro odběr,

Volejte metodu `RequestPublicationUpdate` . `MQSubscription` je podtřída produktu `MQManagedObject`. Použijte odkaz pro přístup k vlastnostem a metodám produktu `MQManagedObject`.

Třída MQTopic.NET

`MQTopic` slouží k publikování nebo odběru zpráv v tématu nebo k dotazování nebo nastavení atributů daného tématu. Vytvoření objektu `MQTopic` pro publikování nebo přihlášení se k odběru 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 1761
- “Metody” na stránce 1761
- “Konstruktory” na stránce 1763

Vlastnosti

Testuje se test produktu `MQException` při získávání vlastností.

public Boolean IsDurable {get;}
Vlastnost jen pro čtení, která vrací `True` , je-li odběr trvalý, nebo `False` jinak. Pokud bylo téma otevřeno pro publikování, vlastnost je ignorována a vždy by vracela `False`.

public Boolean IsManaged {get;};
Vlastnost jen pro čtení, která vrací `True` , je-li odběr spravován správcem front, nebo `False` jinak. Pokud bylo téma otevřeno pro publikování, vlastnost se ignoruje a vždy vrátí hodnotu False.

public Boolean IsSubscribed {get;};
Vlastnost jen pro čtení, která vrací `True` , pokud bylo téma otevřeno pro odběr, a `False` , pokud bylo téma otevřeno pro publikování.

public MQSubscription SubscriptionReference {get;};
Vlastnost jen pro čtení, která vrací objekt `MQSubscription` přidružený k objektu tématu otevřeného pro odběr. Tento 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 `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 pro spravovaný odběr.

Metody

public void Put(MQMessage *message*);
public void Put(MQMessage *message*, MQPutMessageOptions *putMessageOptions*);
Vyvolá výjimku MQException.

Publikuje zprávu na téma.

Úpravy objektu `MQMessage` po dokončení volání operace Put nemají vliv na skutečnou zprávu ve frontě IBM MQ nebo v tématu publikování.

Produkt `Put` aktualizuje vlastnosti `MessageId` a `CorrelationId` objektu `MQMessage` a nemaže data zprávy. Další volání `Put` nebo `Get` odkazují na aktualizované informace v objektu `MQMessage` . Například v následujícím úseku kódu bude první zpráva obsahovat a a druhý ab.

```
msg.WriteString("a");
q.Put(msg,pmo);
```

```
msg.WriteString("b");
q.Put(msg,pmo);
```

zpráva
Objekt MQMessage obsahující data deskriptoru zpráv a zpráva, která má být odeslána. Deskriptor zprávy může být změněn v důsledku této metody. Hodnoty v deskriptoru zpráv bezprostředně po dokončení této metody jsou hodnotami, které byly vloženy do fronty nebo publikovány do tématu.

Následující kódy příčiny jsou vraceny klientovi s možností opětovného připojení:

- MQRC_CALL_INTERRUPTED je-li připojení přerušeno při spuštění volání vložení na trvalou zprávu a opětovné navázání spojení je úspěšné.
- MQRC_NONE , je-li připojení úspěšné při spuštění volání vložení na netrvalou zprávu (viz Obnova aplikace).

Volby putMessage
Volby ovládající akci vložení.

Pokud parametr *putMessageOptions* není zadán, použije se výchozí instance produktu MQPutMessageOptons .

Použijete-li volbu MQPMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Poznámka: Pokud chcete do fronty vložit jednu zprávu, použijte objekt MQQueueManager.Put pro zjednodušení a výkon. Pro tento objekt 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 výchozí instanci produktu MQGetMessageOptions k provedení získání. Použitá volba zprávy je MQGMO_NOWAIT.

Pokud dojde k selhání operace get, objekt MQMessage se nezmění. Pokud je úspěšný, jsou popisovač zprávy a části dat zprávy MQMessage nahrazeny deskriptorem zprávy a daty zprávy z příchozí zprávy.

Všechna volání do IBM MQ z konkrétního MQQueueManager jsou synchronní. Proto, pokud provedete operaci get s čekáním, všechny ostatní podprocesy používající stejný MQQueueManager jsou blokovány od dalších volání IBM MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu IBM MQ současně, každý podproces musí vytvořit svůj vlastní objekt MQQueueManager .

zpráva
Obsahuje deskriptor zprávy a vrácená data zprávy. Některá z polí v deskriptoru zpráv jsou vstupní parametry. Je důležité zajistit, aby vstupní parametry MessageId a CorrelationId byly nastaveny podle potřeby.

Reconnectable klient vrací kód příčiny MQRC_BACKED_OUT po úspěšném opětovném připojení, pro zprávy přijaté pod MQGM_SYNCPOINT.

Volby getMessage
Volby ovládající akci získání.

Použití volby MQC.MQGMO_CONVERT může vést k výjimce s kódem příčiny MQC.MQRC_CONVERTED_STRING_TOO_BIG při konverzi z jednobajtových znakových kódů do dvoubajtových kódů. V tomto případě se zpráva zkopíruje do vyrovnávací paměti bez konverze.

Není-li parametr *getMessageOptions* zadán, bude použita volba zprávy MQGMO_NOWAIT.

Použijete-li volbu MQGMO_LOGICAL_ORDER v reconnectable client, vrátí se kód příčiny MQRC_RECONNECT_INCOMPATIBLE .

Velikost MaxMsg
Největší zpráva, kterou má tento objekt zprávy přijmout. Je-li zpráva ve frontě větší než tato velikost, nastane jedna ze dvou situací:

- Je-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven v objektu MQGetMessageOptions , zpráva je vyplněna co nejvíce informací o zprávě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_ACCEPTED .
- Není-li příznak MQGMO_ACCEPT_TRUNCATED_MSG nastaven, zůstává zpráva ve frontě. Došlo k výjimce s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_TRUNCATED_MSG_FAILED .

Není-li parametr *MaxMsgSize* zadán, 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 v produktu *queueManager*.

Objekty produktu MQTopic úzce souvisejí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu odkazuje topicObject na objekt administrativního tématu. Konstruktor produktu MQTopic získá řetězec tématu z objektu tématu a spojí jej s názvem topicName a vytvoří název tématu. Buď topicObject , nebo topicName mohou mít hodnotu null. Název tématu se shoduje se stromem témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v souboru topicObject.

Témata přidružená k objektu MQTopic jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován pomocí objektu administrativního tématu identifikovaného produktem *topicObject*. Druhý řetězec tématu je *topicString*. Výsledný řetězec tématu přidružený k objektu MQTopic může identifikovat více témat, včetně 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 publikací o tématech. Pokud chcete publikovat a odebírat stejné téma, 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. Předáte-li frontu jako objekt MQDestination , předpokládá se neřízený odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní s tím, že odběr je spravován nebo nespravovaný.

queueManager
Správce front pro přístup k tématu.

cíl
destination je instancí MQQueue . Poskytnutím *destination*se MQTopic otevře jako nespravovaný odběr. Publikace na téma jsou doručeny do fronty, k němuž se přistupuje jako k produktu *destination*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *topicName* je zřetězen s řetězcem tématu definovaným v objektu administrativního tématu produktu *topicObject* . Hodnotu *topicName* lze nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v produktu *topicObject*.

topicObject
Na vstupu *topicObject* je název objektu tématu, který obsahuje řetězec tématu, který tvoří první část názvu tématu. Řetězec tématu v produktu *topicObject* je zřetězený s *topicName*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Na výstupu obsahuje *topicObject* název objektu administračních témat, který je nejpřesnější shodou ve stromu témat k tématu identifikovanému řetězcem tématu.

openAs
Přistupte k tématu, které chcete publikovat nebo odebírat. Tento 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 vyžadována více než jedna volba, přidejte hodnoty dohromady nebo zkombinujte hodnoty voleb pomocí bitového operátoru OR .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace k dokončení operace. Musíte zadat *alternateUserId*, pokud je v parametru voleb nastaven buď MQC.MQOO_ALTERNATE_USER_AUTHORITY nebo MQC.MQSO_ALTERNATE_USER_AUTHORITY .

subscriptionName
subscriptionName je požadován, pokud jsou poskytnuty volby MQC.MQSO_DURABLE nebo MQC.MQSO_ALTER . V obou případech je MQTopic implicitně otevřeno pro odběr. Pokud je nastavena hodnota MQC.MQSO_DURABLE a existuje odběr nebo pokud je nastaven produkt MQC.MQSO_ALTER a odběr neexistuje, dojde k výjimce.

vlastnosti
Nastavte některou ze speciálních vlastností odběru uvedených pomocí hašovací tabulky. Uvedené záznamy v transformační tabulce jsou aktualizovány s výstupními hodnotami. Do transformační tabulky se nepřidají záznamy pro hlášení výstupních hodnot.

- 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 produktu `MQTopic` úzce souvisejí s objekty administrativních témat, které se někdy nazývají objekty témat. Na vstupu odkazuje `topicObject` na objekt administrativního tématu. Konstruktor produktu `MQTopic` získá řetězec tématu z objektu tématu a spojí jej s názvem `topicName` a vytvoří název tématu. Buď `topicObject` , nebo `topicName` mohou mít hodnotu null. Název tématu se shoduje se stromem témat a název nejbližšího odpovídajícího objektu administrativního tématu je vrácen v souboru `topicObject`.

Témata přidružená k objektu `MQTopic` jsou výsledkem kombinace dvou řetězců témat. První řetězec tématu je definován pomocí objektu administrativního tématu identifikovaného produktem *`topicObject`*. Druhý řetězec tématu je *`topicString`*. Výsledný řetězec tématu přidružený k objektu `MQTopic` může identifikovat více témat, včetně 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 publikací o tématech. Pokud chcete publikovat a odebírat stejné téma, 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. Předáte-li frontu jako objekt `MQDestination` , předpokládá se neřízený odběr. Musíte se ujistit, že volby odběru, které jste nastavili, jsou konzistentní s tím, že odběr je spravován nebo nespravovaný.

cíl
`destination` je instancí `MQQueue` . Poskytnutím *`destination`*se `MQTopic` otevře jako nespravovaný odběr. Publikace na téma jsou doručeny do fronty, k němuž se přistupuje jako k produktu *`destination`*.

topicName
Řetězec tématu, který je druhou částí názvu tématu. *`topicName`* je zřetězen s řetězcem tématu definovaným v objektu administrativního tématu produktu *`topicObject`* . Hodnotu *`topicName`* lze nastavit na hodnotu null. V takovém případě je název tématu definován řetězcem tématu v produktu *`topicObject`*.

topicObject
Na vstupu *`topicObject`* je název objektu tématu, který obsahuje řetězec tématu, který tvoří první část názvu tématu. Řetězec tématu v produktu *`topicObject`* je zřetězený s *`topicName`*. Pravidla pro vytváření řetězců témat jsou definována v části Kombinování řetězců témat.

Na výstupu obsahuje *`topicObject`* název objektu administračních témat, který je nejpřesnější shodou ve stromu témat k tématu identifikovanému řetězcem tématu.

openAs
Přistupte k tématu, které chcete publikovat nebo odebírat. Tento 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 vyžadována více než jedna volba, přidejte hodnoty dohromady nebo zkombinujte hodnoty voleb pomocí bitového operátoru OR .

AlternateUserid
Uveďte alternativní ID uživatele, které se použije ke kontrole požadované autorizace k dokončení operace. Musíte zadat *`alternateUserId`*, pokud je v parametru voleb nastaven buď `MQC.MQOO_ALTERNATE_USER_AUTHORITY` nebo `MQC.MQSO_ALTERNATE_USER_AUTHORITY` .

subscriptionName
`subscriptionName` je požadován, pokud jsou poskytnuty volby `MQC.MQSO_DURABLE` nebo `MQC.MQSO_ALTER` . V obou případech je `MQTopic` implicitně otevřeno pro odběr. Pokud je nastavena hodnota `MQC.MQSO_DURABLE` a existuje odběr nebo pokud je nastaven produkt `MQC.MQSO_ALTER` a odběr neexistuje, dojde k výjimce.

vlastnosti
Nastavte některou ze speciálních vlastností odběru uvedených pomocí hašovací tabulky. Uvedené záznamy v transformační tabulce jsou aktualizovány s výstupními hodnotami. Do transformační tabulky se nepřidají záznamy pro hlášení výstupních hodnot.

- `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 produktu `IMQObjectTrigger.NET`

Implementujte produkt `IMQObjectTrigger` ke zpracování zpráv předávaných monitorem produktu **`runmqdnm`**.NET .

Rozhraní

```
public interface IBM.WMQMonitor.IMQObjectTrigger();
```

V závislosti na tom, zda je ovládací prvek bodu synchronizace zadán v příkazu **`runmqdnm`** , je zpráva odebrána z fronty před nebo po vrácení metody `Execute` .

Metody

`void Execute (MQQueueManager` *`queueManager`*`, MQQueue` *`queue`*`, MQMessage` *`message`*`, string` *`param`*`);`

queueManager
Správce front, který je hostitelem monitorované fronty.

fronta
Monitorovaná fronta.

zpráva
Zpráva načtená z fronty.

Parametr
Data předaná z `UserParameter`.

Rozhraní produktu MQC.NET

Chcete-li se dozvědět více o konstantě konstantního názvu pomocí `MQC.`, prohlédněte si konstantu `MQI`. `MQC` definuje všechny konstanty použité `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 produktu .NET IBM MQ

Znaková sada	Popis
37	ibm037
437	ibm437 /PC-původní
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-pobaltské jazyky
813	iso-8859-7 /greek/ ibm813
838	ibm838
850	ibm850 /PC Latin 1
852	ibm852 /PC Latin 2
855	ibm855 /PC cyrilice
856	ibm856
857	ibm857 /PC turečtina
860	ibm860 /PC portugalština
861	ibm861 /PC Islandština
862	ibm862 /PC Hebrejština
863	ibm863 /PC kanadská francouzština
864	ibm864 /PC-arabština
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 /hebrew/ 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 /Big 5 Tradiční čínština
954	EUCJIS
964	ibm964 /CNS 11643 Tradiční čínština
970	ibm970
1006	ibm1006
1025	ibm1025
1026	ibm1026
1089	iso-8859-6 /arabic/ ibm1089
1097	ibm1097
1098	ibm1098
1112	ibm1112
1122	ibm1122
1123	ibm1123
1124	ibm1124
1250	Windows Latin 2
1251	Windows Cyrilice
1252	Windows Latin 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 jazyka C++ produktu IBM MQ zapouzdřují rozhraní MQI (Message Queue Interface) produktu IBM MQ . K dispozici je 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 ve vztahu dědičnosti k bezprostředním nadřazeným třídám, pokud existují.

Ostatní příslušné třídy
Odkazy na dokumenty na jiné relevantní třídy, jako jsou například nadřízené třídy, a třídy objektů používaných v podpisech metod.

Atributy objektu
Atributy třídy. Ty jsou dodatkem k atributům definovaným pro všechny nadřízené třídy. Mnoho atributů odráží IBM MQ členů struktury dat (viz “Křížový odkaz C++ a MQI” na stránce 1771). Podrobný popis viz “Atributy objektů” na stránce 791.

Konstruktory
Podpisy speciálních metod použitých k vytvoření objektu třídy.

Metody objektů (veřejné)
Podpisy metod, které vyžadují instanci třídy pro jejich provoz, a které nemají žádná omezení využití.

Pokud se použije, zobrazí se také následující informace:

Metody třídy (veřejné)
Podpisy metod, které nevyžadují instanci třídy pro jejich provoz, a které nemají žádná omezení využití.

Přetížené metody (nadřízené třídy)
Podpisy těchto 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 jejich provoz a jsou vyhrazeny pro použití implementacemi odvozených tříd. Tato část je zajímavá pouze pro autory tříd, na rozdíl od uživatelů třídy.

Data objektu (chráněná)
Podrobnosti implementace pro data instance objektu jsou k dispozici pro implementace odvozených tříd. Tato část je zajímavá pouze pro autory tříd, na rozdíl od uživatelů třídy.

Kódy příčin
Hodnoty MQRC_ * (viz Kódy dokončení rozhraní API a kódy příčin), které lze očekávat od těchto metod, které selhaly. Úplný seznam kódů příčiny, které se mohou vyskytnout pro objekt třídy, naleznete v dokumentaci nadřízené třídy. Dokumentovaný seznam kódů příčiny pro třídu neobsahuje kódy příčiny pro nadřízené třídy.

Poznámka:

1. Objekty těchto tříd nejsou bezpečné pro podprocesy. Tím se zajistí optimální výkon, ale nepřistupujte k libovolnému objektu z více než jednoho podprocesu.
2. Doporučuje se, abyste pro vícevláknový program použili oddělený objekt ImqQueueManager pro každý podproces. Každý objekt správce musí mít svou vlastní nezávislou kolekci dalších objektů, aby bylo zajištěno, že objekty v různých podprocesech jsou vzájemně izolovány.

Třídy jsou následující:

- “Třída C++ záznamu ImqAuthentication” na stránce 1787
- “Třída C++ ImqBinary” na stránce 1789
- “Třída C++ ImqCache” na stránce 1791
- “Třída C++ ImqChannel” na stránce 1794
- “Parametr ImqCICSBridgeHeader C++” na stránce 1800

- “Třída C++ ImqDeadLetterHeader” na stránce 1806
- “Třída C++ seznamu ImqDistribution” na stránce 1808
- “Třída C++ ImqError” na stránce 1810
- “Třída C++ ImqGetMessageOptions” na stránce 1811
- “Třída C++ ImqHeader” na stránce 1814
- “Parametr ImqIMSBridgeHeader C++” na stránce 1816
- “Třída C++ ImqItem” na stránce 1819
- “Třída C++ ImqMessage” na stránce 1820
- “Třída C++ produktu ImqMessageTracker” na stránce 1827
- “Třída C++ ImqNamelist” na stránce 1830
- “Třída C++ ImqObject” na stránce 1831
- “Třída C++ ImqProcess” na stránce 1837
- “ImqPutMessageOptions Třída C++” na stránce 1838
- “Třída C++ ImqQueue” na stránce 1840
- “Třída C++ správce ImqQueue” na stránce 1851
- “Třída C++ záhlaví ImqReference” na stránce 1866
- “Třída C++ ImqString” na stránce 1869
- “Třída C++ ImqTrigger” na stránce 1874
- “Třída C++ záhlaví ImqWork” na stránce 1877

Křížový odkaz C++ a MQI

Tato kolekce témat obsahuje informace týkající se jazyka C++ pro rozhraní MQI.

Přečtěte si tyto informace spolu s “Datové typy použité v rozhraní MQI” na stránce 235.

Tato tabulka souvisí s datovými strukturami MQI do tříd C + + a zahrnují soubory. Následující témata uvádějí informace křížového odkazu pro každou třídu C + +. Tyto křížové odkazy se vztahují k použití základních procedurálních rozhraní produktu IBM MQ . Třídy ImqBinary, ImqDistributionList a ImqString nemají žádné atributy, které spadají do této kategorie a jsou vyloučeny.

Tabulka 846. Křížový odkaz datové struktury, třídy a 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 846. Křížový odkaz datové struktury, třídy a souboru začlenění (pokračování)		
datová struktura	**Třída**	**Zahrnout soubor**
MQMD	ImqMessage	imqmsg.hpp
	ImqMessageSledovač	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
PŘÍKAZ MQTMC		
MQTMC2	ImqTrigger	imqtrg.hpp
MQXQH		
MQWIHKM	Záhlaví ImqWork	imqwih.hpp

Křížový odkaz záznamu ImqAuthentication

Křížový odkaz na atributy, datové struktury, pole a volání pro třídu C++ záznamu ImqAuthentication.

Tabulka 847. 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	Posunutí LDAPUserName	MQCONNX
	MQAIR	Délka LDAPUserName	MQCONNX

Křížový odkaz ImqCache

Křížový odkaz na atributy a volání pro třídu C++ ImqCache .

Tabulka 848. Atributy a volání	
Atribut	**Volání**
automatická vyrovnávací paměť	MQGET
Délka vyrovnávací paměti	MQGET
ukazatel vyrovnávací paměti	MQGET, MQPUT

Tabulka 848. Atributy a volání (pokračování)	
Atribut	**Volání**
Délka dat	MQGET
Posun dat	MQGET
ukazatel dat	MQGET
délka zprávy	MQGET, MQPUT

Křížový odkaz ImqChannel

Křížový odkaz na atributy, datové struktury, pole a volání pro třídu C++ ImqChannel .

Tabulka 849. Atributy, datové struktury, pole a volání			
Atribut	**datová struktura**	**Pole**	**Volání**
batch heart-beat	MQCD	BatchHeartbeat	MQCONNX
Název kanálu	MQCD	ChannelName	MQCONNX
Název připojení	MQCD	ConnectionName	MQCONNX
	MQCD	Název ShortConnection	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 ukončení příjmu	MQCD		MQCONNX
přijmout názvy uživatelských procedur	MQCD	ReceiveExit	MQCONNX
	MQCD	ReceiveExitsDefinované	MQCONNX
	MQCD	ReceiveExitPtr	MQCONNX
přijmout uživatelská data	MQCD	ReceiveUserData	MQCONNX
	MQCD	ReceiveUserDataPtr	MQCONNX
Název uživatelské procedury zabezpečení zprávy	MQCD	SecurityExit	MQCONNX
uživatelská data zabezpečení	MQCD	Data SecurityUserData	MQCONNX
počet ukončení odeslání	MQCD		MQCONNX
odeslání jmen uživatelských procedur	MQCD	SendExit	MQCONNX
	MQCD	SendExitsDefinované	MQCONNX
	MQCD	SendExitPtr	MQCONNX
odeslání uživatelských dat	MQCD	Data SendUser	MQCONNX

Tabulka 849. Atributy, datové struktury, pole a volání (pokračování)			
Atribut	**datová struktura**	**Pole**	**Volání**
	MQCD	SendUserDataPtr	MQCONNX
SSL CipherSpec	MQCD	Specifikace sslCipher	MQCONNX
Typ ověření klienta SSL	MQCD	Ověřování sslClient	MQCONNX
Název partnera SSL	MQCD	SSLPEERNAME	MQCONNX
Jméno programu transakce	MQCD	TpName	MQCONNX
Typ přenosu	MQCD	TransportType	MQCONNX
Jméno uživatele	MQCD	UserIdentifier	MQCONNX

Křížový odkaz ImqCICSBridgeHeader

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ ImqCICSBridgeHeader .

Tabulka 850. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
kód abend mostu	MQCIH.	AbendCode
Deskriptor ADS	MQCIH.	AdsDescriptor
identifikátor upozornění	MQCIH.	AttentionId
ověřovatel	MQCIH.	Ověřovatel
kód dokončení mostu	MQCIH.	Kód BridgeCompletion
odchylka chyby mostu	MQCIH.	ErrorOffset
kód příčiny mostu	MQCIH.	BridgeReason
kód zrušení mostu	MQCIH.	CancelCode
dialogová úloha	MQCIH.	ConversationalTask
pozice kurzoru	MQCIH.	CursorPosition
token facility	MQCIH.	Poskytovaná služba
doba uchování zařízení	MQCIH.	FacilityKeep
zařízení jako	MQCIH.	FacilityLike
funkce	MQCIH.	Funkce
získat interval čekání	MQCIH.	Interval GetWait
Typ odkazu	MQCIH.	LinkType
identifikátor další transakce	MQCIH.	ID NextTransaction
délka výstupních dat	MQCIH.	Délka OutputData
formát odpovědi	MQCIH.	Formát ReplyTo
návratový kód mostu	MQCIH.	ReturnCode
počáteční kód	MQCIH.	StartCode
stav ukončení úlohy	MQCIH.	Stav TaskEnd

Tabulka 850. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
Identifikátor transakce	MQCIH.	TransactionId
řídicí prvek jednotky práce	MQCIH.	UowControl
verze	MQCIH.	Verze

Křížový odkaz ImqDeadLetterHeader

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ ImqDeadLetterHeader .

Tabulka 851. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
dead-dopis, kód příčiny	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 na atributy a volání pro třídu C++ ImqError .

Tabulka 852. Atributy a volání	
Atribut	**Volání**
kód dokončení	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET, MQCONM, MQDISK, MQPUT, MQSET
kód příčiny	MQBACK, MQBEGIN, MQCLOSE, MQCMIT, MQCONN, MQCONNX, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQSET, MQCONM, MQDISK, MQPUT, MQSET

Křížový odkaz ImqGetMessageOptions

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ ImqGetMessageOptions .

Tabulka 853. 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

Tabulka 853. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
segmentace	MQGMO	Segmentace
stav segmentu	MQGMO	SegmentStatus
	MQGMO	Signal1
	MQGMO	Signal2
participace synchronizačního bodu	MQGMO	Volby
Interval čekání	MQGMO	WaitInterval

Křížový odkaz ImqHeader

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ ImqHeader .

Tabulka 854. 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

Křížový odkaz ImqIMSBridgeHeader

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 855. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
ověřovatel	MQIIH.	Ověřovatel
režim vázaného zpracování	MQIIH.	CommitMode
přepsání logického terminálu	MQIIH.	LTermOverride
název mapy služeb formátu zpráv	MQIIH.	MFSMapName
formát odpovědi	MQIIH.	Formát ReplyTo
rozsah zabezpečení	MQIIH.	SecurityScope
ID instance transakce	MQIIH.	ID TranInstance
Stav transakce	MQIIH.	TranState

Křížový odkaz ImqItem

Křížový odkaz na atributy a volání pro třídu C++ ImqItem .

Tabulka 856. Atributy a volání	
Atribut	**Volání**
id struktury	MQGET

Křížový odkaz ImqMessage

Křížový odkaz na atributy, datové struktury, pole a volání pro třídu C++ ImqMessage .

Tabulka 857. 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	
priority (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 ImqMessageTracker

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ produktu ImqMessageTracker.

Tabulka 858. 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

Tabulka 858. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
ID zprávy	MQMD	MsgId

Křížový odkaz ImqNamelist

Křížový odkaz na atributy, dotazy a volání pro třídu C++ ImqNamelist .

Tabulka 859. Atributy, dotazy a volání		
Atribut	**Inquiry**	**Volání**
Počet názvů	MQIA_NAME_COUNT	MQINQ
Název seznamu názvů	NÁZEV MQCA_NATELEST_NAME	MQINQ

Křížový odkaz ImqObject

Křížový odkaz na atributy, datové struktury, pole, dotazy a volání pro třídu C++ ImqObject .

Tabulka 860. Atributy, datové struktury, pole, dotazy a volání				
Atribut	**datová struktura**	**Pole**	**Inquiry**	**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		IDENTIFIKÁTOR MQCA_Q_MGR_IDENTIFIER	MQINQ

Křížový odkaz ImqProcess

Křížový odkaz na atributy, dotazy a volání pro třídu C++ záznamu ImqAuthentication.

Tabulka 861. Atributy, dotazy a volání		
Atribut	**Inquiry**	**Volání**
ID aplikace	MQCA_APPL_ID	MQINQ

Tabulka 861. Atributy, dotazy a volání (pokračování)		
Atribut	**Inquiry**	**Volání**
Typ aplikace	MQIA_TYP_APLIKACE	MQINQ
Data prostředí	MQCA_ENV_DATA	MQINQ
Data uživatele	MQCA_USER_DATA	MQINQ

Křížový odkaz ImqPutMessageOptions

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 862. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
odkaz kontextu	MQPMO	Kontext
	MQPMO	InvalidDestCount
	MQPMO	KnownDestCount
volby	MQPMO	Volby
pole záznamu	MQPMO	PutMsgRecFields
vyřešený název správce front	MQPMO	Název ResolvedQMgr
vyřešený název fronty	MQPMO	ResolvedQName
	MQPMO	žurnálů
	MQPMO	UnknownDestCount
participace synchronizačního bodu	MQPMO	Volby

Křížový odkaz ImqQueue

Křížový odkaz na atributy, datové struktury, pole, dotazy a volání pro třídu C++ ImqQueue .

Tabulka 863. Křížový odkaz ImqQueue				
Atribut	**datová struktura**	**Pole**	**Inquiry**	**Volání**
Zpětné jméno přefrontování			MQCA_BACKOUT_REQ_Q_NAME	MQINQ
Práh vrácení			PRAHOVÁ HODNOTA MQIA_BACKUT_	MQINQ
název základní fronty			MQCA_BASE_Q_NAME	MQINQ
název klastru			MQCA_NÁZEV_KLASTRU	MQINQ
Název seznamu názvů klastru			SEZNAM NÁZVŮ KLASTRU MQCA_CLUSTER_	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

Tabulka 863. Křížový odkaz ImqQueue (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
Datum vytvoření			MQCA_CREATION_DATE	MQINQ
Čas vytvoření			ČAS VYTVOŘENÍMQCATION_TIME	MQINQ
Aktuální délka			MQIA_AKTUÁLNÍ_HODNOTA Q_DEPTH	MQINQ
výchozí vazba			MQIA_DEF_BIND	MQINQ
Výchozí volba otevření pro vstup			MQIA_DEF_INPUT_OPEN_OPTION	MQINQ
Výchozí trvání			MQIA_DEF_PERSISTENCE	MQINQ
Výchozí priorita			MQIA_DEF_PRIORITA	MQINQ
Typ definice			TYP_DEFINICE_MQIA_	MQINQ
událost vysoké hloubky			MQIA_Q_DEPTH_HIGH_EVENT, UDÁLOST	MQINQ
horní mez hloubky			MQIA_Q_DEPTH_HIGH_LIMIT	MQINQ
událost dolní meze			MQIA_Q_DEPTH_LOW_EVENT	MQINQ
dolní mez hloubky			MQIA_Q_DEPTH_LOW_LIMIT	MQINQ
maximální událost hloubky			MQIA_Q_DEPTH_MAX_LIMIT	MQINQ
distribuční seznamy			MQIA_DICT_LISTS	MQINQ, MQSET
název dynamické fronty	MQOD	DynamicQName		
Uloľení počtu vrácení			MQIA_HARDEN_GET_BACKOUT	MQINQ
Typ indexu			MQIA_INDEX_TYPE	MQINQ
inhibuje získání			MQIA_INHIBIT_GET	MQINQ, MQSET
inhibují put			MQIA_INHIBIT_PUT	MQINQ, MQSET
Název inicializační fronty			NÁZEV QCCA_INITIATION_Q_NAME	MQINQ
Maximální hloubka			MQIA_MAX_Q_DEPTH	MQINQ
Maximální délka zprávy			MQIA_MAX_MSG_LENGTH	MQINQ
Pořadí doručení zpráv			POSLOUPNOST MQIA_MSG_DELIVERY_SEQUENCE	MQINQ
další distribuovaná fronta				
Třída netrvalých zpráv			MQIA_NPM_TŘÍDA	MQINQ
Otevření pro vstup - počet			MQIA_OPEN_INPUT_COUNT	MQINQ

Tabulka 863. Křížový odkaz ImqQueue (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
Otevření pro výstup - počet			MQIA_OPEN_OUTPUT_COUNT	MQINQ
předchozí distribuovaná fronta				
Název procesu			NÁZEV_PROCESU_MQCA_	MQINQ
Účtování fronty			MQIA_ACCOUNTING_Q	MQINQ
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_NÁZEV_VZDÁLENÉ_FRONTY	MQINQ
vyřešený název správce front	MQOD	Název ResolvedQMgr		
vyřešený název fronty	MQOD	ResolvedQName		
Interval uchování			MQIA_RETENCTION_INTERVAL	MQINQ
obor			ROZSAH MQIA_	MQINQ
interval služeb			INTERVAL MQIA_Q_SERVICE_INTERVAL	MQINQ
událost intervalu služeb			MQIA_Q_SERVICE_INTERVAL_EVENT	MQINQ
Možnost sdílení			SQIDABILITY	MQINQ
paměťová třída			TŘÍDA MQCA_STORAGE_CLASS	MQINQ
Jméno přenosové fronty			MQCA_ XMIT_Q_NÁZEV	MQINQ
Řízení spouštěče			MQIA_TRIGGER_CONTROL	MQINQ, MQSET
Data spouątěče			DATA MQCA_TRIGGER_DATA	MQINQ, MQSET
Hloubka spouątěče			HLOUBKA MQIA_TRIGGERU_T	MQINQ, MQSET
Priorita zpráv spouątěče			MQIA_TRIGGER_MSG_PRIORITY	MQINQ, MQSET
typ spouštěče			TYP_SPOUŠTĚČE_MQIA_TYPE	MQINQ, MQSET
Využití			MQIA_USAGE	MQINQ

Křížový odkaz správce ImqQueue

Křížový odkaz na atributy, datové struktury, pole, dotazy a volání pro třídu C++ správce ImqQueue.

Tabulka 864. Atributy, datové struktury, pole, dotazy a volání

Atribut	datová struktura	Pole	Inquiry	Volání
potlačení evidence připojení			MQIA_ACCOUNTING_CONN_OVERRIDE	MQINQ
Interval evidence			MQIA_ACCOUNTING_INTERVAL	MQINQ
Záznam činnosti			ZÁZNAM MQIA_ACTIVITY_RECORDING	MQINQ
Převzetí nového agenta MCA - kontrola			MQIA_ADOPTNEWMCA_CHECK	MQINQ
Převzetí nového agenta MCA - typ			MQIA_ADOPTNEWMCA_TYPE	MQINQ
Typ ověřování	MQCP	AuthenticationType		MQCONNX
událost oprávnění			UDÁLOST MQIA_AUTHORITY_EVENT	MQINQ
volby začátku	OBJEKT MQBO	Volby		MQBEGIN
událost mostu			UDÁLOST MQIA_BRIGE_EVENT	MQINQ
Automatická definice kanálů			MQIA_CHANNEL_AUTO_DEF	MQINQ
událost automatické definice kanálu			AUTOMATICKÁ UDÁLOST MQIA_CHANNEL_AUTO_EVENT	MQIAK
Uživatelská procedura automatické definice kanálů			MQIA_CHANNEL_AUTO_EXIT	MQIAK
událost kanálu			UDÁLOST MQIA_CHANNEL_EVENT	MQINQ
Adaptéry inicializátoru kanálu			MQIA_CHINIT_ADAPTÉRY	MQINQ
Řízení iniciátoru kanálu			MQIA_CHINIT_CONTROL	MQINQ
Dispečeři inicializátoru kanálu			MQIA_CHINIT_DISPEČE	MQINQ
Automatické spuštění trasování inicializátoru kanálu			MQIA_CHINIT_TRACE_AUTO_START	MQINQ

Tabulka 864. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
Velikost tabulky trasování inicializátoru kanálu			VELIKOST MQIA_CHINIT_TRACE_TABLE_SIZE	MQINQ
Monitorování kanálů			MQIA_MONITORING_CHANNEL	MQINQ
odkaz na kanál	MQCD	ChannelType		MQCONNX
Statistika kanálů			MQIA_STATISTICS_CHANNEL	MQINQ
znaková sada			MQIA_CODE_CHAR_SET_ID	MQINQ
Monitorování odesílatele klastru			MQIA_MONITORING_AUTO_CLUSDR	MQINQ
Statistiky odesílatele klastru			MQIA_STATISTICS_AUTO_CLUSDR	MQINQ
Data pracovní zátěže klastru			MQCA_CLUSTER_WORKLOAD_DATA	MQINQ
Uživatelská procedura pracovní zátěže klastru			MQCA_CLUSTER_WORKLOAD_EXIT	MQINQ
Délka pracovní zátěže klastru			DÉLKA MQIA_CLUSTER_WORKLOAD_LENGTH	MQINQ
mru pro pracovní zátěž klastru			MQIA_CLWL_MRU_CHANNELS	MQINQ
Pracovní zátěž klastru - použitá fronta			MQIA_CLWL_USEQ.	MQINQ
událost příkazu			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

Tabulka 864. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
Kryptografický hardware	MQSCO	CryptoHardware		MQCONNX
název fronty nedoručených zpráv			MQCA_DEAD_LETTER_Q_NAME	MQINQ
výchozí název přenosové fronty			MQCA_DEF_MIT_QM_QNAME	MQINQ
distribuční seznamy			MQIA_DICT_LISTS	MQINQ
skupina dns			SKUPINA MQCA_DNS_GROUP	MQINQ
dns wlm			MQIA_DNS_WLM	MQINQ
záznam prvního ověření	MQSCO	AuthInfoRecOffset		MQCONNX
	MQSCO	AuthInfoRecPtr		MQCONNX
blokace události			UDÁLOST MQIA_INHIBIT_EVENT	MQINQ
Verze adresy IP			VERZE MQIA_IP_ADDRESS_VERSION	MQINQ
úložiště klíčů	MQSCO	KeyRepository		MQCONNX
počet resetování klíče	MQSCO	Počet KeyReset		MQCONNX
Časovač modulu listener			ČASOVAČ MQIA_LISTENER_ČASOVAČ	MQINQ
lokální událost			MQIA_LOKÁLNÍ_UDÁLOST	MQINQ
Událost modulu protokolování			UDÁLOST MQIA_LOGGER_EVENT	MQINQ
Název skupiny LU			MQCA_LU_GROUP_NAME	MQINQ
Název jednotky LU			NÁZEV MQCA_LU_NAME	MQINQ
Přípona ramena lu62			MQCA_LU62_ARM_SUFFIX	MQINQ
lu62 kanály			MQIA_LU62_CHANNELS	MQINQ
maximum aktivních kanálů			MQIA_ACTIVE_CHANNE	MQINQ
Maximální počet kanálů			MQIA_MAX_KANÁLY	MQINQ
maximální úchyty			MQIA_MAX_HANDLES	MQINQ
Maximální délka zprávy			MQIA_MAX_MSG_LENGTH	MQINQ
maximální priorita			MQIA_MAX_PRIORITA	MQINQ

Tabulka 864. Atributy, datové struktury, pole, dotazy a volání (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
Maximum nepotvrzených zpráv			MQIA_MAX_UNCOMMITTED_MSGS	MQINQ
Evidence MQI			MQIA_ACCOUNTING_MQI	MQINQ
Statistika MQI			MQIA_STATISTICS_MQI	MQINQ
maximum odchozího portu			MQIA_OUTBOUND_PORT_MAX	MQINQ
minimální odchozí port			MQIA_OUTBOUND_PORT_MIN	MQINQ
heslo	MQCP	CSPPasswordPtr		MQCONNX
	MQCP	CSPPasswordOffset		MQCONNX
	MQCP	CSPPasswordLength		MQCONNX
událost výkonu			MQIA_PERFORMANCE_VÝKONU	MQINQ
platforma			PLATFORMA MQIA_	MQINQ
Účtování fronty			MQIA_ACCOUNTING_Q	MQINQ
Monitorování fronty			MQIA_MONITORING_Q	MQINQ
Statistiky fronty			MQIA_STATISTICS_Q	MQINQ
Časový limit pro příjem			ČASOVÝ LIMIT MQIA_RECEIVE_TIMEOUT	MQINQ
minimální časový limit příjmu			MQIA_RECEIVE_TIMEOUT_MIN	MQINQ
Typ časového limitu pro příjem			MQIA_RECEIVE_TIMEOUT_TYPE	MQINQ
vzdálená událost			MQIA_VZDÁLENÝ_UDÁLOST	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
fips ssl			MQIA_SSL_FIPS_REQUIRED	MQINQ
Počet resetování klíče SSL			MQIA_SSL_RESET_COUNT	MQINQ
událost start-stop			MQIA_START_STOP_EVENT	MQINQ
Interval statistiky			INTERVAL MQIA_STATISTICS_INTERVAL	MQINQ

Tabulka 864. Atributy, datové struktury, pole, dotazy a volání (pokračování)				
Atribut	**datová struktura**	**Pole**	**Inquiry**	**Volání**
Dostupnost synchronizačního bodu			MQIA_SYNCPOINT	MQINQ
kanály tcp			MQIA_TCP_CHANNELS	MQINQ
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			MQIA_TRACE_ROUTE_RECORDING	MQINQ
Interval spouštěče			INTERVAL PRO SPOUŠTĚČ MQIA_TRIGGER	MQINQ
Jméno uživatele	MQCP	CSPUserIdPtr		MQCONNX
	MQCP	CSPUserIdPosunutí		MQCONNX
	MQCP	CSPUserIdDélka		MQCONNX

Křížový odkaz záhlaví ImqReference

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 865. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
cílové prostředí	MQRMH	DestEnvDélka, DestEnvOffset
Název místa určení	MQRMH	DestNameDélka, DestName
ID instance	MQRMH	ID ObjectInstance
logická délka	MQRMH	Délka DataLogical
logický posun	MQRMH	Offset DataLogical
logický posun 2	MQRMH	DataLogicalOffset2
Typ odkazu	MQRMH	ObjectType
Zdrojové prostředí	MQRMH	SrcEnvDélka, SrcEnvOffset
Zdrojový název	MQRMH	SrcNameDélka, SrcNameOffset

Křížový odkaz ImqTrigger

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 866. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
ID aplikace	MQTM	ApplId

Tabulka 866. Mapování atributů, datových struktur a polí (pokračování)		
Atribut	**datová struktura**	**Pole**
Typ aplikace	MQTM	ApplType
Data prostředí	MQTM	EnvData
Název procesu	MQTM	ProcessName
Název fronty	MQTM	QName
Data spouątěče	MQTM	TriggerData
Data uživatele	MQTM	UserData

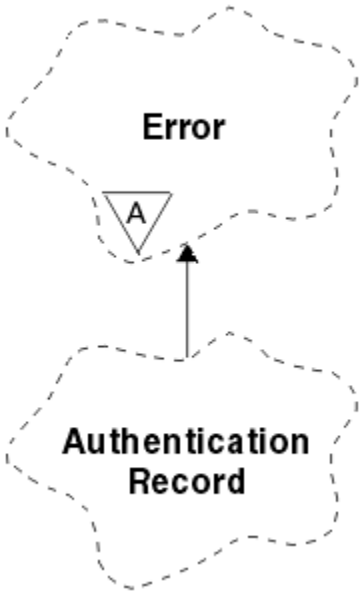
Křížový odkaz záhlaví ImqWork

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 867. Mapování atributů, datových struktur a polí		
Atribut	**datová struktura**	**Pole**
token zprávy	MQWIHKM	MessageToken
Název služby	MQWIHKM	ServiceName
servisní krok	MQWIHKM	ServiceStep

Třída C++ záznamu ImqAuthentication

Tato třída zapouzdřuje záznam ověřovacích informací (MQAIR) pro použití při 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 objektu” na stránce 1788
- “Konstruktory” na stránce 1788
- “Metody objektů (veřejné)” na stránce 1788

- “Metody objektů (chráněné)” na stránce 1789

Atributy objektu

Název připojení
Název připojení k serveru LDAP CRL. Jedná se o adresu IP nebo název DNS, následovaný volitelně číslem portu, v závorkách.

odkaz na připojení
Odkaz na objekt správce ImqQueue, který poskytuje požadované připojení k (lokálnímu) správci front. Počáteční hodnota je nula. Nezaměňujte tento název s názvem správce front, který identifikuje správce front (pravděpodobně vzdáleného) pro pojmenovanou frontu.

další záznam ověření
Další objekt této třídy, v žádném konkrétním pořadí, který má stejný **odkaz na připojení** jako tento objekt. Počáteční hodnota je nula.

heslo
Heslo zadané pro ověření připojení k serveru LDAP CRL.

předchozí ověřovací záznam
Předchozí objekt této třídy, v žádném konkrétním pořadí, který má stejný **odkaz na připojení** jako tento objekt. Počáteční hodnota je nula.

typ
Typ informací o ověření 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 *vzduchu*, přičemž nahradí existující data instance.

const ImqString & connectionName () const;
Vrací **název připojení**.

void setConnectionName (const ImqString & ***name*** **);**
Nastaví **název připojení**.

void setConnectionName (const char * ***název*** **= 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 adresáře *pAir*, přičemž nahradí existující data instance. To může zahrnovat přidělení závislého úložiště.

void clear (MQAIR * ***pAir*** **);**
Vymaže strukturu a uvolní závislé úložiště, na které odkazuje *pAir*.

Záznam ImqAuthenticationRecord * nextAuthenticationRecord () const;
Vrací **další záznam ověření**.

const ImqString & password () const;
Vrací **heslo**.

void setPassword (const ImqString & *password*);
Nastavuje **heslo**.

void setPassword (const char * *heslo* = 0);
Nastavuje **heslo**.

Záznam ImqAuthenticationRecord * previousAuthenticationRecord () const;
Vrátí **předchozí ověřovací záznam**.

Typ MQLONG () const;
Vrátí **typ**.

void setType (const MQLONG *typ*);
Nastavuje **typ**.

const ImqString & userName () const;
Vrací **jméno uživatele**.

void setUserName (const ImqString & *name*);
Nastavuje **jméno uživatele**.

void setUserNázev (const char * *název* = 0);
Nastavuje **jméno uživatele**.

Metody objektů (chráněné)

void setNextAuthenticationRecord (záznam ImqAuthenticationRecord * *pAir* = 0);
Nastaví **další záznam ověření**.

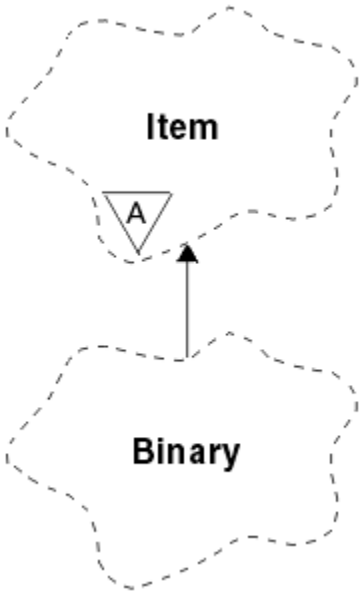
Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nevylomí seznam autentizačních záznamů.

void setPreviousAuthenticationRecord (záznam ImqAuthenticationRecord * *pAir* = 0);
Nastaví **předchozí ověřovací záznam**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nevylomí seznam autentizačních záznamů.

Třída C++ ImqBinary

Tato třída zapouzdřuje binární bajtové pole, které lze použít pro hodnoty ImqMessage **accounting token**, **correlation id**, a **message id** . Umožňuje snadné přiřazení, kopírování a porovnávání.



Obrázek 15. Třída ImqBinary

Atributy objektu

data
Pole bajtů binárních dat. Počáteční hodnota je null.

Délka dat
Počet bajtů. Počáteční hodnota je nula.

ukazatel dat
Adresa prvního bajtu **dat**. Počáteční hodnota je nula.

Konstruktory

ImqBinary( );
Výchozí konstruktor.

ImqBinary(const ImqBinary & ***binary*** **);**
Kopírovací konstruktor.

ImqBinary(const void * ***data*****, const size_t** ***délka*** **);**
Kopíruje *délku* bajtů z *dat*.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & ***msg*** **);**
Zkopíruje data **data** do vyrovnávací paměti zpráv a nahradí veškerý existující obsah. Nastaví formát *msg* **format** na hodnotu MQFMT_NONE.

Další podrobnosti naleznete v popisu metody třídy ImqItem .

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Nastaví data **data** přenesením zbývajících dat z vyrovnávací paměti zpráv a nahradí existující **data**.

Aby bylo úspěšné, ImqMessage **formátování** musí být MQFMT_NONE.

Další podrobnosti naleznete v popisu metody třídy ImqItem .

Metody objektů (veřejné)

void operator = (const ImqBinary & *binary*);
Kopíruje bajty z *binary*.

ImqBoolean operátor == (const ImqBinary & *binary*);
Porovná tento objekt s *binary*. Vrací FALSE, pokud není rovno a TRUE jinak. Objekty jsou stejné, mají-li stejnou **délku dat** a shodu bajtů.

ImqBoolean copyOut (void * *buffer*, const size_t *length*, const char *pad* = 0);
Kopíruje až *délka* bajtů z **ukazatele dat** do *vyrovnávací paměti*. Pokud je **délka dat** nedostatečná, je zbývající prostor ve vyrovnávací paměti *buffer* zaplněn *pad* bajtů. *vyrovnávací paměť* může být nula, je-li *délka* také nula. *délka* nesmí být záporná. Pokud je úspěšný, vrací TRUE.

size_t dataLength () const ;
Vrací **datovou délku**.

ImqBoolean setDataDélka (const size_t *délka*);
Nastavuje **datovou délku**. Je-li **délka dat** změněna jako výsledek této metody, data v objektu jsou neinicializovaná. Pokud je úspěšný, vrací TRUE.

void * dataPointer () const ;
Vrací **datový ukazatel**.

ImqBoolean isNull () const ;
Vrací TRUE, je-li **délka dat** nula, nebo pokud jsou všechny **data** bajtů nula. Jinak vrací hodnotu FALSE.

ImqBoolean set (const void * *buffer*, const size_t *length*);
Kopíruje *délku* bajtů z *vyrovnávací paměti*. Pokud je úspěšný, vrací TRUE.

Metody objektů (chráněné)

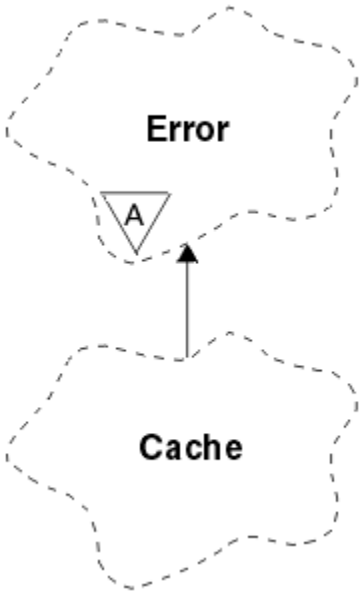
void clear ();
Zmenšuje **délku dat** na nulu.

Kódy příčin

- MQRC_NO_BUFFER
- MQRC_STORAGE_NOT_AVAILABLE
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT

Třída C++ ImqCache

Tuto třídu použijte k zadržení nebo zařazení dat do paměti.



Obrázek 16. Třída ImqCache

Tuto třídu použijte k zadržení nebo zařazení dat do paměti. Můžete určit velikost vyrovnávací paměti pevné velikosti nebo může systém automaticky poskytovat flexibilní velikost paměti. Tato třída se vztahuje k voláním MQI uvedeným v seznamu "Křížový odkaz ImqCache" na stránce 1772.

Atributy objektu

automatická vyrovnávací paměť
Označuje, zda je vyrovnávací paměť spravována automaticky systémem (TRUE), nebo je dodána uživatelem (FALSE). Na počátku je nastavena hodnota TRUE.

Tento atribut není nastaven přímo. Je nastaven nepřímo buď pomocí metody **useEmptyBuffer** , nebo metody **useFullBuffer** .

Je-li zadáno uživatelské úložiště, tento atribut má hodnotu FALSE, paměť vyrovnávací paměti nemůže růst a mohou se vyskytnout chyby přetečení vyrovnávací paměti. Adresa a délka vyrovnávací paměti zůstávají konstantní.

Pokud není zadáno uživatelské úložiště, tento atribut má hodnotu TRUE a vyrovnávací paměť se může postupně zvětšovat a přizpůsobovat se tak libovolnému objemu dat zprávy. Když se však vyrovnávací paměť rozroste, adresa vyrovnávací paměti se může změnit, proto buďte opatrní při použití **ukazatele vyrovnávací paměti** a **ukazatele dat**.

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 úspěšných za **ukazatelem dat**. Musí se rovnat nebo být menší než **délka zprávy**. Počáteční hodnota je nula.

Posun dat
Počet bajtů před **ukazatelem dat**. Musí se rovnat nebo být menší než **délka zprávy**. Počáteční hodnota je nula.

ukazatel dat
Adresa části vyrovnávací paměti, která má být zapsána nebo přečtena z další. Počáteční hodnota je null.

délka zprávy
Počet bajtů významných dat ve vyrovnávací paměti. Počáteční hodnota je nula.

Konstruktory

ImqCache( );
Výchozí konstruktor.

ImqCache(const ImqCache & ***cache*** **);**
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqCache & ***cache*** **);**
Zkopíruje data z objektu *cache* do objektu do **délky zprávy** dat. Je-li **automatická vyrovnávací paměť** FALSE, hodnota **délka vyrovnávací paměti** již musí být dostatečná pro umístění kopírovaných dat.

ImqBoolean automaticBuffer () const ;
Vrací hodnotu **automatická vyrovnávací paměť** .

size_t bufferLength () const ;
Vrací **délku vyrovnávací paměti**.

char * bufferPointer () const ;
Vrátí **ukazatel vyrovnávací paměti**.

void clearMessage ();
Nastaví **délku zprávy** a **posunutí dat** na nulu.

size_t dataLength () const ;
Vrací **datovou délku**.

size_t dataOffset () const ;
Vrátí **posun dat**.

ImqBoolean setDataOffset (const size_t ***offset*** **);**
Nastavuje **posun dat**. Hodnota **délka zprávy** se v případě potřeby zvýší, aby se zajistilo, že není menší než **posunutí dat**. Tato metoda vrací TRUE, je-li úspěšná.

char * dataPointer () const ;
Vrací kopii **ukazatele dat**.

size_t messageLength () const ;
Vrátí **délku zprávy**.

ImqBoolean setMessageLength (const size_t ***délka*** **);**
Nastavuje **délku zprávy**. Zvýší **délku vyrovnávací paměti** , pokud je to nezbytné, aby se zajistilo, že **délka zprávy** není větší než **délka vyrovnávací paměti**. Omezuje **posunutí dat** , je-li to nezbytné, aby se zajistilo, že není větší než **délka zprávy**. Pokud je úspěšný, vrací TRUE.

ImqBoolean moreBytes (const size_t ***bytes-required*** **);**
Zajistí, že *bajty-požadované* jsou k dispozici více bajtů (pro zápis) mezi **ukazatelem dat** a koncem vyrovnávací paměti. Pokud je úspěšný, vrací TRUE.

Má-li parametr **automatická vyrovnávací paměť** hodnotu TRUE, je podle potřeby získána další paměť; v opačném případě musí být **délka vyrovnávací paměti** již adekvátní.

ImqBoolean read (const size_t ***length*****, char * &** ***external-buffer*** **);**
Kopíruje *délku* bajtů z vyrovnávací paměti začínající na pozici **data pointer** do *external-buffer*. Po zkopírování dat se hodnota **posunutí dat** zvýší o *délka*. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean resizeBuffer (const size_t ***délka*** **);**
Vře **délka vyrovnávací paměti**za předpokladu, že **automatická vyrovnávací paměť** má hodnotu TRUE. Toho lze dosáhnout tak, že znovu alokuje paměť vyrovnávací paměti. Do nové **délky zprávy**

dat z existující vyrovnávací paměti se zkopíruje do nového. Maximální počet zkopírovaných bajtů je *délka* bajtů. Změní se **ukazatel vyrovnávací paměti** . **Délka zprávy** a **posunutí dat** jsou v mezích nové vyrovnávací paměti zachovány co nejpřesněji. Příkaz vrací TRUE, je-li úspěšný, a FALSE, pokud **automatická vyrovnávací paměť** je FALSE.

Poznámka: Tato metoda může selhat s hodnotou MQRC_STORAGE_NOT_AVAILABLE, pokud došlo k problému se systémovými prostředky.

ImqBoolean useEmptyBuffer (const char * *external-buffer*, const size_t *length*);
Identifikuje prázdnou vyrovnávací paměť uživatele, nastavení **ukazatele vyrovnávací paměti** tak, aby ukazovala na *externí-vyrovnávací paměť*, **délku vyrovnávací paměti** na *délku*a **délku zprávy** na nulu. Provede příkaz **clearMessage**. Je-li vyrovnávací paměť plně naplněná daty, použijte místo toho metodu **useFullBuffer** . Je-li vyrovnávací paměť částečně naplňovaná daty, použijte metodu **setMessageLength** k označení správného množství. Tato metoda vrací TRUE, je-li úspěšná.

Tuto metodu lze použít k identifikaci pevné velikosti paměti, jak již bylo popsáno dříve (*externí-vyrovnávací paměť* není null a *délka* je nenulová), v takovém případě **automatická vyrovnávací paměť** je nastavena na FALSE, nebo může být použita k vrácení na systémem spravovanou flexibilní paměti (*external-buffer* je null a *length* je nula), v tom případě **automatická vyrovnávací paměť** je nastavena na TRUE.

ImqBoolean useFullBuffer (const char * *externalBuffer*, const size_t *length*);
Co se týče **useEmptyBuffer**, kromě toho, že **délka zprávy** je nastavena na *length*. Pokud je úspěšný, vrací TRUE.

ImqBoolean write (const size_t *length*, const char * *external-buffer*);
Kopíruje *délku* bajtů z *externí vyrovnávací paměti*do vyrovnávací paměti začínající na pozici **data ukazatele** . Po zkopírování dat se **posunutí dat** zvýší o *délku*a **délka zprávy** se v případě potřeby zvýší, aby se zajistilo, že není menší než hodnota nového **posunutí dat** . Tato metoda vrací TRUE, je-li úspěšná.

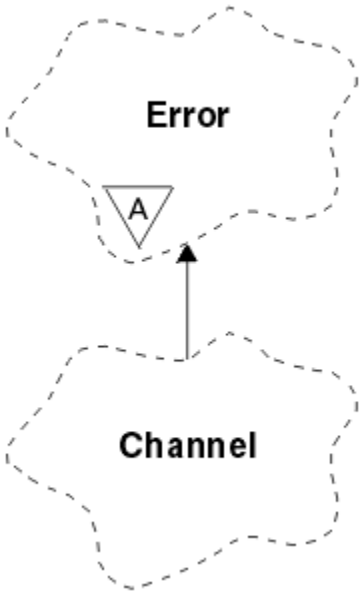
Má-li parametr **automatická vyrovnávací paměť** hodnotu TRUE, je zaručena adekvátní velikost paměti. V opačném případě nesmí být maximální hodnota **offset dat** větší než **délka vyrovnávací paměti**.

Kódy příčin

- MQRC_BUFFER_NOT_AUTOMATIC
- MQRC_DATA_OŘÍZNUTÁ
- MQRC_INSUFFICIENT_BUFFER
- MQRC_INSUFFICIENT_DATA
- MQRC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- MQRC_ZERO_LENGTH

Třída C++ ImqChannel

Tato třída zapouzdřuje definici kanálu (MQCD) pro použití při provádění správce: :connect metody pro vlastní připojení klienta.



Obrázek 17. Třída ImqChannel

Další podrobnosti naleznete v popisu správce: :connect metody a Ukázkový program HELLO WORLD (imqwrld.cpp).

Ne všechny uvedené metody jsou použitelné na všechny platformy. Další informace naleznete v popisech příkazů DEFINE CHANNEL a ALTER CHANNEL .

Třída ImqChannel není v produktu z/OSpodporována.

- "Atributy objektu" na stránce 1795
- "Konstruktory" na stránce 1796
- "Metody objektů (veřejné)" na stránce 1796
- "Kódy příčin" na stránce 1800

Atributy objektu

batch heart-beat
Počet milisekund mezi kontrolami, kdy 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 hodnotu MQCOMPRESS_NOT_AVAILABLE.

interval prezenčního signálu
Počet sekund mezi kontrolami, že připojení stále pracuje. Počáteční hodnota je 300.

Interval udržení aktivity
Počet sekund, které uplynuly do komunikačního zásobníku specifikující časování udržení aktivity pro daný kanál. Počáteční hodnota je MQKAI_AUTO.

Lokální adresa
Lokální komunikační adresa kanálu.

Maximální délka zprávy
Maximální délka zprávy podporované kanálem v jediné komunikaci. Počáteční hodnota je 4 194 304.

Komprese zpráv
Seznam technik komprese dat zprávy podporovaných kanálem. Počáteční hodnoty jsou všechny nastaveny na hodnotu MQCOMPRESS_NOT_AVAILABLE.

Název režimu
Název režimu. Počáteční hodnota je null.

heslo
Heslo zadané pro ověření připojení. Počáteční hodnota je null.

počet ukončení příjmu
Počet uživatelských procedur pro příjem. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

přijmout názvy uživatelských procedur
Názvy uživatelských procedur pro příjem.

přijmout uživatelská data
Data přidružená k ukončům příjmu.

Název uživatelské procedury zabezpečení zprávy
Název uživatelské procedury pro zabezpečení zprávy, která má být vyvolána na straně serveru připojení. Počáteční hodnota je null.

uživatelská data zabezpečení
Data, která mají být předána uživatelské proceduře pro zabezpečení zprávy. Počáteční hodnota je null.

počet ukončení odeslání
Počet uživatelských procedur odeslání. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

odeslání jmen uživatelských procedur
Názvy uživatelských procedur odeslání.

odeslání uživatelských dat
Data přidružená k uživatelským procedurám odeslání.

SSL CipherSpec
CipherSpec pro použití s protokolem TLS.

Typ ověření klienta SSL
Typ ověření klienta pro použití s TLS.

Název partnera SSL
Název partnera pro použití s TLS.

Jméno programu transakce
Název transakčního programu. Počáteční hodnota je null.

Typ přenosu
Typ transportu připojení. Počáteční hodnota je MQXPT_LU62.

Jméno uživatele
Identifikátor uživatele dodaný pro autorizaci. Počáteční hodnota je null.

Konstruktory

ImqChannel( ) ;
Výchozí konstruktor.

ImqChannel(const ImqChannel & *kanál*);
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqChannel & *kanál*);
Zkopíruje data instance z *kanálu*a nahradí veškerá existující data instance.

MQLONG batchHeartBeat () const;
Vrátí **batch heart-beat**.

ImqBoolean setBatchHeartBeat(const MQLONG *heartbeat* = 0L);
Nastaví **batch heart-beat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString channelName() const;
Vrací **název kanálu**.

ImqBoolean setChannelNázev (const char * *název* = 0);
Nastaví **název kanálu**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString connectionName() const;
Vrací **název připojení**.

ImqBoolean setConnectionNázev (const char * *název* = 0);
Nastaví **název připojení**. Tato metoda vrací TRUE, je-li úspěšná.

size_t headerCompressionCount () const;
Vrací podporovaný počet technik komprese dat záhlaví.

ImqBoolean headerCompression(const size_t count, MQLONG compress []) const;
Vrací kopie podporovaných technik komprese dat záhlaví v souboru **compress**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setHeaderCompression (const size_t count, const MQLONG compress []);

Nastaví podporované metody komprese dat záhlaví na **compress**.

Nastavuje počet podporovaných metod komprese dat záhlaví na hodnotu **count**.

Tato metoda vrací TRUE, je-li úspěšná.

MQLONG heartBeatInterval () const;
Vrátí **interval prezenčního signálu**.

ImqBoolean setHeartBeatInterval(const MQLONG *interval* = 300L);
Nastaví **interval prezenčního signálu**. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG keepAliveInterval () const;
Vrací hodnotu **keep alive interval**.

ImqBoolean setKeepAliveInterval(const MQLONG *interval* = MQKAI_AUTO);
Nastavuje **interval udržení aktivity**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString localAddress() const;
Vrátí **lokální adresu**.

ImqBoolean setLocalAdresa (const char * *adresa* = 0);
Nastavuje **lokální adresu**. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG maximumMessageLength () const;
Vrátí **maximální délku zprávy**.

ImqBoolean setMaximumMessageLength(const MQLONG *délka* = 4194304L);
Nastavuje **maximální délku zprávy**. Tato metoda vrací TRUE, je-li úspěšná.

size_t messageCompressionCount () const;
Vrátí počet podporovaných technik komprese dat zprávy.

ImqBoolean messageCompression(const size_t count, MQLONG compress []) const;
Vrací kopie podporovaných technik komprese dat zprávy v souboru **compress**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setMessageCompression (const size_t count, const MQLONG compress []);

Nastavuje podporované metody komprese dat zpráv, které mají být komprimovány.

Nastavuje počet podporovaných metod komprese dat zpráv, které mají být započíteny.

Tato metoda vrací TRUE, je-li úspěšná.

ImqString modeName() const;
Vrací **název režimu**.

ImqBoolean setModeNázev (const char * *název* = 0);
Nastavuje **název režimu**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString heslo () const;
Vrací **heslo**.

ImqBoolean setPassword(const char * *heslo* = 0);
Nastavuje **heslo**. Tato metoda vrací TRUE, je-li úspěšná.

size_t receiveExitPočet () const;
Vrátí **počet výjezdu pro příjem**.

ImqString receiveExitName ();
Vrací první z **názvů uživatelských procedur příjmu**, pokud existují. Je-li hodnota parametru **receive exit count** nula, vrátí prázdný řetězec.

ImqBoolean receiveExitNames (const size_t *počet*, ImqString * *names* []);
Vrací kopie **názvů uživatelských procedur pro příjem** v *jménech*. Nastaví jakékoli *názvy* přesahující **počet výjezdu pro přijetí** na řetězce null. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveExitName(const char * *název* = 0);
Nastaví **názvy uživatelských procedur příjmu** na jediný *název*. Název *název* může být prázdný nebo mít hodnotu null. Nastaví **počet uživatelských procedur pro příjem** na hodnotu 1 nebo nula. Vymaže **příjem uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveExitNames(const size_t *počet*, const char * *names* []);
Nastaví **názvy uživatelských procedur pro příjem** na *names*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet ukončení příjmu** na *počet*. Vymaže **příjem uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveExitNames(const size_t *počet*, const ImqString * *names* []);
Nastaví **názvy uživatelských procedur pro příjem** na *names*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet ukončení příjmu** na *počet*. Vymaže **příjem uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString receiveUserData ();
Vrátí první z položek **receive user data** , pokud existuje. Je-li **počet ukončení příjmu** nula, vrátí prázdný řetězec.

ImqBoolean receiveUserData (const size_t *počet*, ImqString * *data* []);
Vrátí kopie položek **přijetí uživatelských dat** v *datech*. Nastaví veškerá data *data* přesahující **počet uživatelských procedur příjmu** na řetězce s hodnotou null. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveUserData(const char * *data* = 0);
Nastaví **příjem uživatelských dat** na jednu položku *data*. Pokud *data* nemají hodnotu null, hodnota parametru **receive exit count** musí být alespoň 1. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveUserData(const size_t *počet*, const char * *data* []);
Nastavuje **příjem uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur příjmu**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveUserData(const size_t *počet*, const ImqString * *data* []);
Nastavuje **příjem uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur příjmu**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString securityExitNázev () const;
Vrátí **název uživatelské procedury zabezpečení**.

ImqBoolean setSecurityExitName(const char * *název* = 0);
Nastaví **název uživatelské procedury zabezpečení**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString securityUserData () const;
Vrací **data uživatele zabezpečení**.

ImqBoolean setSecurityUserData(const char * *data* = 0);
Nastavuje **uživatelská data zabezpečení**. Tato metoda vrací TRUE, je-li úspěšná.

size_t sendExitPočet () const;
Vrací **počet ukončení odeslání**.

ImqString sendExitName ();
Vrací první z **názvů uživatelských procedur odeslání**, pokud existují. Vrací prázdný řetězec, je-li **počet ukončovacích procedur odeslání** nula.

ImqBoolean sendExitNázvy (const size_t *počet*, ImqString * *names* []);
Vrací kopie **názvů uživatelských procedur odeslání** do *názvů*. Nastaví všechny názvy *názvů* přesahující **počet uživatelských procedur odeslání** do řetězců s hodnotou null. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendExitName(const char * *název* = 0);
Nastaví **názvy uživatelských procedur odeslání** na jediný *název*. Název *název* může být prázdný nebo mít hodnotu null. Nastaví **počet uživatelských procedur odeslání** na hodnotu 1 nebo nula. Vymaže **odeslání uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná

ImqBoolean setSendExitNames(const size_t *počet*, const char * *names* []);
Nastaví **názvy uživatelských procedur odeslání** na *names*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet ukončení odeslání** na hodnotu *count*. Vymaže **odeslání uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendExitNames(const size_t *počet*, const ImqString * *names* []);
Nastaví **názvy uživatelských procedur odeslání** na *names*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet ukončení odeslání** na hodnotu *count*. Vymaže **odeslání uživatelských dat**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString sendUserData ();
Vrací první z položek **odeslání uživatelských dat** , pokud existuje., Vrací prázdný řetězec, je-li **počet ukončovacích procedur odeslání** nula.

ImqBoolean sendUserData (const size_t *počet*, ImqString * *data* []);
Vrátí kopie položek **odeslání uživatelských dat** v *datech*. Nastaví veškerá data *data* přesahující **počet uživatelských procedur odeslání** na řetězce s hodnotou null. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendUserData(const char * *data* = 0);
Nastaví **odeslání uživatelských dat** na jednu položku *data*. Pokud data *data* nemají hodnotu null, hodnota **počet ukončení odeslání** musí být alespoň 1. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendUserData(const size_t *počet*, const char * *data* []);
Nastavuje **odeslání uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur pro odeslání zprávy**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendUserData(const size_t *počet*, const ImqString * *data* []);
Nastavuje **odeslání uživatelských dat** na *data*. *count* nesmí být větší než **počet uživatelských procedur pro odeslání zprávy**. Tato metoda vrací TRUE, je-li úspěšná.

Specifikace ImqString sslCipherSpecification () const;
Vrátí specifikaci šifrování TLS.

ImqBoolean setSslCipherSpecification(const char * *název* = 0);
Nastaví specifikaci šifry TLS. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG sslClientAuthentication () const;
Vrátí typ ověřování klienta TLS.

ImqBoolean setSslClientAuthentication(const MQLONG *auth* = MQSCA_REQUIRED);
Nastaví typ ověřování klienta TLS. Tato metoda vrací TRUE, je-li úspěšná.

ImqString sslPeerName () const;
Vrátí název partnera TLS.

ImqBoolean setSslPeerName(const char * *název* = 0);
Nastaví název partnera TLS. Tato metoda vrací TRUE, je-li úspěšná.

ImqString transactionProgramNázev () const;
Vrátí **název transakčního programu**.

ImqBoolean setTransactionProgramName(const char * *název* = 0);
Nastavuje **název transakčního programu**. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG transportType() const;
Vrací **typ přenosu**.

ImqBoolean setTransportTyp (const MQLONG *typ* = MQXPT_LU62);
Nastaví **typ transportu**. Tato metoda vrací TRUE, je-li úspěšná.

ImqString userId() const;
Vrátí **ID uživatele**.

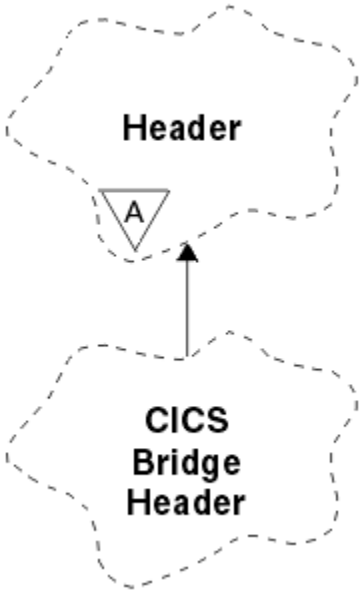
ImqBoolean setUserId (const char * *id* = 0);
Nastavuje **ID uživatele**. Tato metoda vrací TRUE, je-li úspěšná.

Kódy příčin

- CHYBA MQRC_DATA_LENGTH_ERROR
- CHYBA MQRC_ITEM_COUNT_ERROR
- MQRC_NULL_POINTER
- CHYBA MQRC_SOURCE_BUFFER_ERROR

Parametr ImqCICSBridgeHeader C++

Tato třída zapouzdřuje specifické vlastnosti datové struktury MQCIH.



Obrázek 18. Třída ImqCICSBridgeHeader

Objekty této třídy jsou používány aplikacemi, které odesílají zprávy do CICS bridge prostřednictvím IBM MQ for z/OS.

- "Atributy objektu" na stránce 1800
- "Konstruktory" na stránce 1803
- "Přetížené metody ImqItem" na stránce 1803
- "Metody objektů (veřejné)" na stránce 1803
- "Data objektu (chráněná)" na stránce 1805
- "Kódy příčin" na stránce 1805
- "Návratové kódy" na stránce 1805

Atributy objektu

Deskriptor ADS
Odeslání/přijetí deskriptoru ADS. Tuto hodnotu lze nastavit pomocí příkazu MQCADSD_NONE. Počáteční hodnota je MQCADSD_NONE. Jsou možné následující další hodnoty:

- MQCADSD_NONE

- MQCADSD_SEND
- MQCADSD_RECV
- FORMÁT ZPRÁVY MQCADSD_MSGFORMAT

identifikátor upozornění
Klíč AID. Pole musí mít délku MQ_ATTENTION_ID_LENGTH.

ověřovatel
RACF heslo nebo přístupový lístek. Počáteční hodnota obsahuje mezery, délku MQ_AUTHENTICATOR_LENGTH.

kód abend mostu
Kód ukončení přemostění, o délce MQ_ABEND_CODE_LENGTH. Počáteční hodnota jsou čtyři prázdné znaky. Hodnota vrácená v tomto poli je závislá na návratovém kódu. Další podrobnosti naleznete v části Tabulka 868 na stránce 1805.

kód zrušení mostu
Kód transakce na konec mostu. Pole je vyhrazeno, 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 je závislá na návratovém kódu. Další podrobnosti naleznete v části Tabulka 868 na stránce 1805.

odchylka chyby mostu
Offset chyby mostu. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

kód příčiny mostu
Kód příčiny. Toto pole může obsahovat buď důvod IBM MQ , nebo hodnotu CICS EIBRESP2 . Pole má počáteční hodnotu MQRC_NONE. Hodnota vrácená v tomto poli je závislá na návratovém kódu. Další podrobnosti naleznete v části Tabulka 868 na stránce 1805.

návratový kód mostu
Návratový kód z CICS bridge. Počáteční hodnota je MQCRC_OK.

dialogová úloha
Zda může být úloha dialogová. Počáteční hodnota je MQCCT_NO. Jsou možné následující další hodnoty:

- MQCKT_YES
- MQCCT_NO

pozice kurzoru
Pozice kurzoru. Počáteční hodnota je nula.

doba uchování zařízení
Doba vydání služby CICS bridge .

zařízení jako
Terminal emulovaný atribut. Pole musí mít délku MQ_FACILITY_LIKE_LENGTH.

token facility
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 je závislá na návratovém kódu. Další podrobnosti naleznete v části Tabulka 868 na stránce 1805.

Následující další hodnoty jsou možné, když funkce **function** obsahuje název volání IBM MQ :

- MQCFUNC_MQCONN
- FUNKCE MQCFUNC_MQGET
- MQCFUNC_MQINQ

- MQCFUNC_NONE
- MQCFUNC_MQOPEN
- MQCFUNC_PUT
- MQCFUNC_MQPUT1

získat interval čekání
Interval čekání pro volání MQGET vydaného úlohou CICS bridge . Počáteční hodnota je MQCGWI_DEFAULT. Pole se použije pouze tehdy, když má **řízení práce** hodnotu MQCUOWC_FIRST. Jsou možné následující další hodnoty:

- MQCGWI_DEFAULT
- MQWI_UNLIMITED

Typ odkazu
Typ odkazu. Počáteční hodnota je MQCLT_PROGRAM. Jsou možné následující další hodnoty:

- MQCLT_PROGRAM
- TRANSAKCE MQCLT_TRANSACTION

identifikátor další 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.

počáteční kód
Počáteční kód transakce. Pole musí mít délku MQ_START_CODE_LENGTH. Počáteční hodnota je MQCSC_NONE. Jsou možné následující další hodnoty:

- MQCSC_START
- POČÁTEČNÍ_DATA MQCSC_STARTDATA
- MQCSC_TERMINPUT
- MQCSC_NONE

stav ukončení úlohy
Koncový stav úlohy. Počáteční hodnota je MQCTES_NOSYNC. Jsou možné následující další hodnoty:

- MQCTES_COMMIT
- MQCTES_BACKOUT
- ÚLOHA 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 má volba **control unow** hodnotu MQCUOWC_FIRST nebo MQCUOWC_ONLY.

Řízení pracovní jednotky
Řízení pracovní jednotky. Počáteční hodnota je MQCUOWC_ONLY. Jsou možné následující další hodnoty:

- NEJPRVE MQCUOWC_FIRST
- MQCUOWC_MIDDLE
- MQCUOWC_LAST
- POUZE 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 & *header*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut(ImqMessage & *msg*);
Vloží datovou strukturu MQCIH do vyrovnávací paměti zpráv na začátku, dále přesune existující data zprávy dále a nastaví formát zprávy na MQFMT_CICS.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

virtual ImqBoolean pasteIn(ImqMessage & *msg*);
Načte datovou strukturu MQCIH z vyrovnávací paměti zpráv. Aby bylo úspěšné, zakódování objektu *msg* musí být MQENC_NATIVE. Načtěte zprávy s MQGMO_CONVERT do MQENC_NATIVE. Aby byla úspěšná, formát ImqMessage musí být MQFMT_CICS.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqCICSBridgeHeader & *header*);
Zkopíruje data instance ze záhlaví *header*a nahradí existující data instance.

Funkce MQLONG ADSDDescriptor () const;
Vrací kopii **deskriptoru ADS**.

void setADSDescriptor(const MQLONG *descriptor* = MQCADSD_NONE);
Nastaví **Deskriptor ADS**.

ImqString attentionIdentifier() const;
Vrací kopii **identifikátoru upozornění**doplněná o koncové mezery na délku MQ_ATTENTION_ID_LENGTH.

void setAttentionIdentifier (const char * *data* = 0);
Nastaví **identifikátor upozornění**doplněný o koncové mezery na délku MQ_ATTENTION_ID_LENGTH. Není-li zadána žádná data *data* , obnoví se hodnota **identifikátor upozornění** na počáteční hodnotu.

ImqString Ověřovatel () const;
Vrací kopii **ověřovatele**doplněnou na délku MQ_AUTHENTICATOR_LENGTH na délku MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator(const char * *data* = 0);
Nastavuje **ověřovatele**doplněný o koncové mezery na délku MQ_AUTHENTICATOR_LENGTH. Pokud není dodána žádná *data* , resetuje se **ověřovatel** na počáteční hodnotu.

ImqString bridgeAbendKód () const;
Vrací kopii **kódu abend mostu**, doplněnou o koncové mezery na délku MQ_ABEND_CODE_LENGTH.

ImqString bridgeCancelCode () const;
Vrací kopii **kódu zrušení mostu**, který je doplněn o délku MQ_CANCEL_CODE_LENGTH s koncovými mezerami.

void setBridgeCancelCode(const char * *data* = 0);
Nastaví **kód zrušení mostu**doplněný o koncové mezery MQ_CANCEL_CODE_LENGTH. Pokud není dodána žádná *data* , resetuje **kód zrušení mostu** na počáteční hodnotu.

MQLONG bridgeCompletionCode () const;
Vrací kopii **kódu dokončení mostu**.

MQLONG bridgeErrorOffset () const;
Vrací kopii **offsetu chyby mostu**.

MQLONG bridgeReasonKód () const;
Vrací kopii **kódu příčiny mostu**.

MQLONG bridgeReturnKód () const;
Vrací **návratový kód mostu**.

MQLONG conversationalTask() const;
Vrátí kopii **konverzační úlohy**.

void setConversationalÚloha (const MQLONG *úloha* = MQCCT_NO);
Nastaví **konverzační úlohu**.

MQLONG cursorPosition() const;
Vrací kopii **pozice kurzoru**.

void setCursorPozice (const MQLONG *pozice* = 0);
Nastaví **pozici kurzoru**.

MQLONG facilityKeepTime () const;
Vrací kopii **doby uchování zařízení**.

void setFacilityKeepTime(const MQLONG *čas* = 0);
Nastaví **dobu zachování zařízení**.

ImqString facilityLike() const;
Vrací kopii **zařízení jako**, která je doplněna koncovými mezerami do délky MQ_FACILITY_LIKE_LENGTH.

void setFacilityLike (const char * *název* = 0);
Nastavuje **zařízení jako**, doplněné 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 setFacilityToken (const ImqBinary & *token*);
Nastaví **token zařízení**. **Délka dat** prvku *token* musí být buď nula, nebo MQ_FACILITY_LENGTH. Pokud je úspěšný, vrací TRUE.

void setFacilityToken (const MQBYTE8 *token* = 0);
Nastaví **token zařízení**. Prvek *token* může být nula, což je stejné jako určení hodnoty MQCFAC_NONE. Je-li token *token* nenulový, musí adresa MQ_FACILITY_LENGTH bajtů adresovat binární data. Při použití předdefinovaných hodnot, jako je MQCFAC_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu. Příklad: (MQBYTE *) MQCFAC_NONE.

Funkce ImqString () const;
Vrací kopii **funkce**doplněné koncovými mezerami až po délku MQ_FUNCTION_LENGTH.

MQLONG getWaitInterval () const;
Vrací kopii příkazu **get wait interval**.

void setGetWaitInterval(const MQLONG *interval* = MQCGWI_DEFA
Nastavuje **interval čekání na získání**.

MQLONG linkType() const;
Vrátí kopii typu **link type**.

void setLinkType (const MQLONG *typ* = MQCLT_PROGRAM);
Nastavuje **typ odkazu**.

ImqString Identifikátor nextTransactionIdentifier () const;
Vrací kopii dat **identifikátoru další transakce** , doplněnou o koncové mezery na délku MQ_TRANSACTION_ID_LENGTH.

MQLONG outputDataDélka () const;
Vrací kopii **výstupní délky dat**.

void setOutputDataLength(const MQLONG *délka* = MQCODL_AS_INPUT);
Nastaví **délku výstupních dat**.

Formát ImqString replyToFormat () const;
Vrací kopii názvu **formátu odpovědi** , který je doplněn o koncové mezery až po délku MQ_FORMAT_LENGTH.

void setReplyToFormat(const char * *název* = 0);
Nastavuje **formát odpovědi**, který je vyplněn koncovými mezerami na délku MQ_FORMAT_LENGTH. Není-li zadán žádný *název* , resetuje se počáteční hodnota **reply-to format** .

ImqString startCode() const;
Vrací kopii **počátečního kódu**doplněnou o koncové mezery na délku MQ_START_CODE_LENGTH.

void setStartCode (const char * *data* = 0);
Nastaví data pro **počáteční kód** , doplněná o koncové mezery do délky MQ_START_CODE_LENGTH. Pokud není dodána žádná *data* , resetuje **počáteční kód** na počáteční hodnotu.

MQLONG taskEndStatus () const;
Vrací kopii **stavu ukončení úlohy**.

ImqString transactionIdentifier() const;
Vrací kopii dat **identifikátoru transakce** , doplněnou o koncové mezery na délku MQ_TRANSACTION_ID_LENGTH.

void setTransactionIdentifier (const char * *data* = 0);
Nastavuje **identifikátor transakce**doplněný o koncové mezery na délku MQ_TRANSACTION_ID_LENGTH. Pokud není dodána žádná *data* , resetuje **identifikátor transakce** na počáteční hodnotu.

MQLONG UOWControl () const;
Vrátí kopii **řídicího prvku UOW**.

void setUOWControl(const MQLONG *control* = MQCUOWC_ONLY);
Nastaví **Řídicí prvek UOW**.

MQLONG version () const;
Vrátí číslo **version** .

ImqBoolean setVersion(const MQLONG *verze* = MQCIH_VERSION_2);
Nastaví číslo verze **version** . Pokud je úspěšný, vrací TRUE.

Data objektu (chráněná)

MQLONG *olVersion*
Maximální číslo verze MQCIH, které lze umístit v úložišti přiděleném pro *opcih*.

PMQCIH *opcih*
Adresa datové struktury MQCIH. Přidělené množství úložného prostoru je označeno hodnotou *olVersion*.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_WRONG_VERSION

Návratové kódy

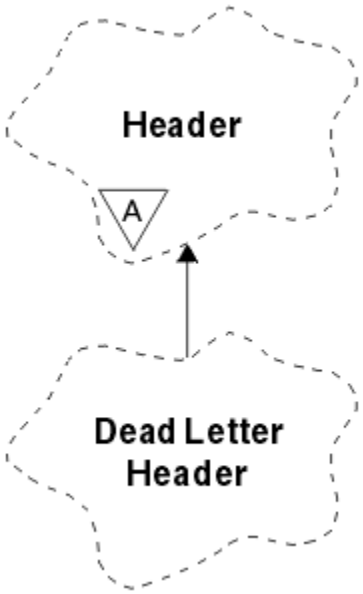
Tabulka 868. Návratové kódy třídy ImqCICSBridgeHeader

Návratový kód	Funkce	CompCode	Příčina	Kód nestandardního konce
MQCRC_OK				

Tabulka 868. Návratové kódy třídy ImqCICSBridgeHeader (pokračování)				
Návratový kód	**Funkce**	**CompCode**	**Příčina**	**Kód nestandardního konce**
CHYBA MQCRC_BRIDGE_ERROR			MQFB_CICS	
CHYBA MQCRC_MQ_API_ERROR	Název volání produktu IBM MQ	IBM MQ CompCode	IBM MQ Příčina	
MQCRC_BRIDGE_TIMEOUT	Název volání produktu IBM MQ	IBM MQ CompCode	IBM MQ Příčina	
CHYBA MQCRC_CICS_EXEC_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_SECURITY_ERROR	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_PROGRAM_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_TRANSID_NOT_AVAILABLE	CICS EIBFN	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				CICS ABCODE
UKONČENÍ MQCRC_APPLICATION_ABEND				CICS ABCODE

Třída C++ ImqDeadLetterHeader

Tato třída zapouzdřuje funkce datové struktury MQDLH.



Obrázek 19. Třída ImqDeadLetterHeader

Objekty této třídy jsou obvykle používány aplikací, která narazí na zprávu, která nemůže být zpracována. Do fronty dead-letter se vloží nová zpráva obsahující záhlaví a obsah zprávy a zpráva se zruší.

- “Atributy objektu” na stránce 1807
- “Konstruktory” na stránce 1807

Atributy objektu

dead-dopis, kód příčiny
Příčina zprávy, která byla doručena do fronty nedoručených zpráv. Počáteční hodnota je MQRC_NONE.

Název správce cílových front
Název původního správce cílové fronty. Název je řetězec s délkou MQ_Q_MGR_NAME_LENGTH. Jeho počáteční hodnota je null.

název cílové fronty
Název původní cílové fronty. Název je řetězec s délkou MQ_Q_NAME_LENGTH. Jeho počáteční hodnota je null.

Název vkládající aplikace
Název aplikace, která vložila zprávu do fronty nedoručených zpráv. Název je řetězec s délkou 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 s délkou MQ_PUT_DATE_LENGTH. Jeho počáteční hodnota je prázdný řetězec.

Čas vložení
Čas, kdy byla zpráva vložena do fronty nedoručených zpráv. Čas je řetězec s délkou MQ_PUT_TIME_LENGTH. Jeho počáteční hodnota je prázdný řetězec.

Konstruktory

ImqDeadLetterHeader( );
Výchozí konstruktor.

ImqDeadLetterHeader(const ImqDeadLetterHeader & *header*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & *msg*);
Vloží datovou strukturu MQDLH do vyrovnávací paměti zpráv na začátku a dále přesune existující data zprávy dál. Nastaví formát *msg* na hodnotu MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na stránce “Třída C++ ImqHeader” na stránce 1814 .

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Načte datovou strukturu MQDLH z vyrovnávací paměti zpráv.

Aby byla úspěšná, formát ImqMessage musí být MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na stránce “Třída C++ ImqHeader” na stránce 1814 .

Metody objektů (veřejné)

void operator = (const ImqDeadLetterHeader & *header*);
Zkopíruje data instance zkopírovaná ze záhlaví *header*, přičemž nahradí existující data instance.

MQLONG deadLetterReasonCode () const;
Vrátí kód příčiny smrtelného dopisu.

void setDeadLetterReasonCode (const MQLONG *reason*);
Nastavuje kód příčiny smrtelného dopisu.

ImqString destinationQueueManagerName () const;
Vrátí název správce cílové fronty, který je zbaven všech koncových mezer.

void setDestinationQueueManagerNázev (const char * *název*);
Nastaví název správce cílové fronty. Ořízne data delší než MQ_Q_MGR_NAME_LENGTH (48 znaků).

ImqString destinationQueueNázev () const;
Vrací kopii názvu cílové fronty bez koncových mezer.

void setDestinationQueueName (const char * *název*);
Nastaví název cílové fronty. Ořízne data delší než hodnota MQ_Q_NAME_LENGTH (48 znaků).

ImqString putApplicationName () const;
Vrací kopii názvu vsazené aplikace a odstraní všechny koncové mezery.

void setPutApplicationName (const char * *název* = 0);
Nastaví název aplikace put. Zkrátí data delší než hodnota MQ_PUT_APPL_NAME_LENGTH (28 znaků).

MQLONG putApplicationType () const;
Vrátí typ aplikace put.

void setPutApplicationType (const MQLONG *typ* = MQAT_NO_CONTEXT);
Nastaví typ aplikace put.

ImqString putDate () const;
Vrací kopii data vložení, odsazené z koncových mezer.

void setPutDatum (const char * *datum* = 0);
Nastavuje datum vložení. Zkrátí data delší než hodnota MQ_PUT_DATE_LENGTH (8 znaků).

ImqString putTime () const;
Vrací kopii času vložení, odsazené z koncových mezer.

void setPutČas (const char * *čas* = 0);
Nastavuje čas vložení. Zkrátí data delší než hodnota MQ_PUT_TIME_LENGTH (8 znaků).

Data objektu (chráněná)

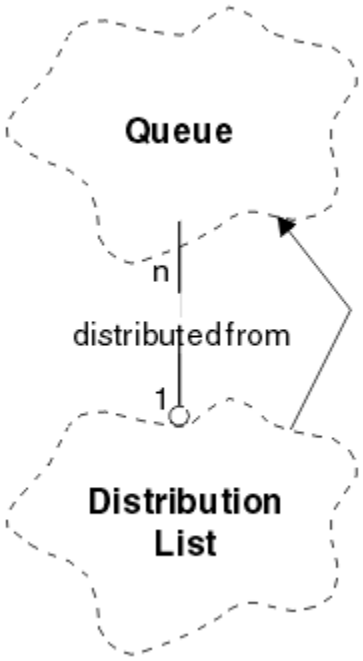
MQDLH *omqdlh*
Datová struktura MQDLH.

Kódy příčin

- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT
- CHYBA MQRC_STRUC_ID_ERROR
- CHYBA MQRC_ENCODING_ERROR

Třída C++ seznamu ImqDistribution

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. Třída seznamu ImqDistribution

- "Atributy objektu" na stránce 1809
- "Konstruktory" na stránce 1809
- "Metody objektů (veřejné)" na stránce 1809
- "Metody objektů (chráněné)" na stránce 1810

Atributy objektu

první distribuovaná fronta
První z jednoho nebo více objektů třídy, v žádném konkrétním pořadí, ve kterém **odkaz na distribuční seznam** adresuje tento objekt.

Na počátku 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 se na něj odkazují, se automaticky zavřou.

Konstruktory

Seznam ImqDistributionList ();
Výchozí konstruktor.

Seznam ImqDistributionList (const ImqDistributionList & *list*);
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqDistributionList & *list*);
Všechny objekty, které odkazují na **tento** objekt, jsou před kopírováním vyhodnoceny. Po vyvolání této metody se na objekt **tento** nebudou odkazovat žádné objekty.

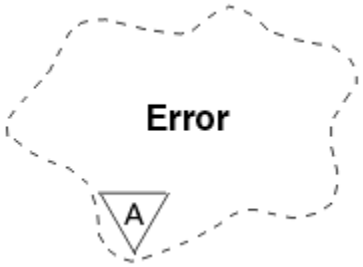
*** firstDistributedQueue () const ;**
Vrací **první distribuovanou frontu**.

Metody objektů (chráněné)

void setFirstDistributedQueue (* *queue* = 0);
Nastavuje **první distribuovanou frontu**.

Třída C++ ImqError

Tato abstraktní třída poskytuje informace o chybách přidružených k objektu.



Obrázek 21. Třída ImqError

- "Atributy objektu" na stránce 1810
- "Konstruktory" na stránce 1810
- "Metody objektů (veřejné)" na stránce 1810
- "Metody objektů (chráněné)" na stránce 1811
- "Kódy příčin" na stránce 1811

Atributy objektu

kód dokončení
Nejnovější kód dokončení. Počáteční hodnota je nula. Jsou možné následující další hodnoty:

- MQCC_OK
- VAROVÁNÍ MQCC_WARNING
- SELHÁNÍ MQCC_FAILED

kód příčiny
Nejnovější kód příčiny. Počáteční hodnota je nula.

Konstruktory

ImqError( );
Výchozí konstruktor.

ImqError(const ImqError & *error*);
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqError & *error*);
Zkopíruje data instance z *error*, nahradí existující data instance.

void clearErrorCodes ();
Nastaví kód **completion code** a **reason code** (kód příčiny) na nulu.

MQLONG completionCode () const ;
Vrátí **kód dokončení**.

MQLONG reasonCode () const ;
Vrátí **kód příčiny**.

Metody objektů (chráněné)

ImqBoolean checkReadPointer (const void * *pointer*, const size_t *length*);
Ověřuje, zda je kombinace ukazatele a délky platná pro přístup jen pro čtení, a pokud je úspěšná, vrátí hodnotu TRUE.

ImqBoolean checkWritePointer (const void * *pointer*, const size_t *length*);
Ověřuje, zda kombinace ukazatele a délky je platná pro přístup čtení-zápisu, a navrátí hodnotu TRUE, je-li úspěšná.

void setCompletionCode (const MQLONG *kód* = 0);
Nastavuje **kód dokončení**.

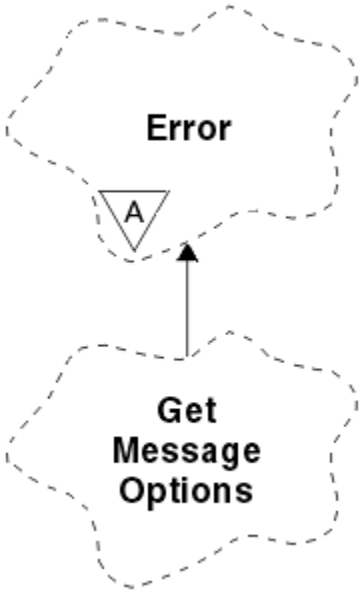
void setReasonCode (const MQLONG *kód* = 0);
Nastavuje **kód příčiny**.

Kódy příčin

- CHYBA MQRC_BUFFER_ERROR

Třída C++ ImqGetMessageOptions

Tato třída zapouzdřuje strukturu dat MQGMO



Obrázek 22. Třída ImqGetMessageOptions

- "Atributy objektu" na stránce 1811
- "Konstruktory" na stránce 1813
- "Metody objektů (veřejné)" na stránce 1813
- "Metody objektů (chráněné)" na stránce 1814
- "Data objektu (chráněná)" na stránce 1814
- "Kódy příčin" na stránce 1814

Atributy objektu

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:

- ID_SKUP_MQMOVÝCH_SKUPIN
- MQMO_MATCH_MSG_SEQ_NUMBER
- MQMO_MATCH_OFSET
- MQMO_MSG_TOKEN
- MQMO_NONE

token zprávy
Token zprávy. Binární hodnota (MQBYTE16) s délkou 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
- NEJPRVE MQGMO_BROWSE_FIRST
- PŘÍŠTĚ MQGMO_BROWSE_NEXT
- MQGMO_BROWSE_MSG_UNDER_CURSOR
- MQGMO_MSG_UNDER_CURSOR
- MQGMOVÝ_ZÁMEK
- MQGMO_ODEMKNOUT
- SOUBOR MQGMO_ACCEPT_TRUNCATED_MSG
- SIGNÁL MQGMO_SET_DATA
- FUNKCE MQGMO_FAIL_IF_QUIESCING
- MQGMO_CONVERT
- MQGMO_LOGICAL_ORDER
- ZPRÁVA MQGMO_COMPLETE_MSG
- MQGMO_ALL_MSGS_AVAILABLE
- DOSTUPNÉ 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 na tuto délku doplněny 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:

- SEGMENT MQSS_SEGMENT
- MQSS_LAST_SEGMENT

participace synchronizačního bodu
Nabývá hodnoty True, pokud jsou zprávy načítány pod řízením synchronizačního bodu.

Interval čekání
Doba, kdy metoda získání metody při čekání na vhodnou zprávu čeká na doručení vhodné zprávy, pokud ještě není k dispozici. Počáteční hodnota je nula, což má vliv na nekonečné čekání. Je možná další hodnota MQWI_UNLIMITED. Tento atribut je ignorován, pokud volby nezahrnují MQGMO_WAIT.

Konstruktory

ImqGetMessageOptions( );
Výchozí konstruktor.

ImqGetMessageOptions(const ImqGetMessageOptions & *gmo*);
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqGetMessageOptions & *gmo*);
Zkopíruje data instance z prostředí *gmo*a nahradí existující data instance.

MQCHAR groupStatus () const;
Vrátí stav skupiny.

void setGroupStav (const MQCHAR *stav*);
Nastaví stav skupiny.

Funkce MQLONG matchOptions () const;
Vrátí volby shody.

void setMatchVolby (const MQLONG *volby*);
Nastavuje volby shody.

ImqBinary messageToken() const;
Vrátí token zprávy.

ImqBoolean setMessageToken (const ImqBinary & *token*);
Nastaví token zprávy. Délka dat *token* musí být buď nula, nebo MQ_MSG_TOKEN_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setMessageToken (const MQBYTE16 *token* = 0);
Nastaví token zprávy. *token* může být nula, což je stejné jako uvedení hodnoty MQMTOK_NONE. Je-li token *token* nenulový, musí být adresovat proměnné MQ_MSG_TOKEN_LENGTH bajtů binárních dat.

Při použití předdefinovaných hodnot, jako je MQMTOK_NONE, nemusíte vytvářet přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQMTOK_NONE.

Volby MQLONG () const;
Vrátí volby.

void setOptions (const MQLONG *volby*);
Nastaví volby, včetně hodnoty účasti synchronizačního bodu.

ImqString resolvedQueueNázev () const;
Vrací kopii vyřešeného názvu fronty.

MQLONG returnedLength() const;
Vrátí navrácenou délku.

Segmentace MQCHAR () const;
Vrátí segmentaci.

void setSegmentation (const MQCHAR *hodnota*);
Nastavuje segmentaci.

MQCHAR segmentStatus () const;
Vrátí stav segmentu.

void setSegmentStav (const MQCHAR *stav*);
Nastaví stav segmentu.

ImqBoolean syncPointÚčast () const;
Vrátí hodnotu účasti synchronizačního bodu, která má hodnotu TRUE, pokud volby zahrnují buď MQGMO_SYNCPOINT, nebo MQGMO_SYNCPOINT_IF_PERSISTENT.

void setSyncPointParticipation (const ImqBoolean *sync*);
Nastaví hodnotu účasti synchronizačního bodu. Má-li položka *sync* hodnotu TRUE, změní volby zahrnutí MQGMO_SYNCPOINT a vyloučí obě hodnoty MQGMO_NO_SYNCPOINT a MQGMO_SYNCINT_IF_PERSISTENT. Je-li *sync* FALSE, pozmění volby pro zahrnutí MQGMO_NO_SYNCPOINT a vyloučí jak MQGMO_SYNCPOINT, tak MQGMO_SYNCPINT_IF_PERSISTENT.

MQLONG waitInterval () const;
Vrátí interval čekání.

void setWaitInterval (const MQLONG *interval*);
Nastavuje interval čekání.

Metody objektů (chráněné)

static void setVersionSupported (const MQLONG);
Nastaví verzi MQGMO. Výchozí hodnota je MQGMO_VERSION_3.

Data objektu (chráněná)

MQGMO *omqgmo*
Datová struktura MQGMO verze 2. Přistupte k polím MQGMO, která jsou podporována pouze pro MQGMO_VERSION_2 .

PMQGMO *opgmo*
Adresa struktury dat MQGMO. Číslo verze pro tuto adresu je uvedeno v *olVersion*. Zkontrolujte číslo verze před přístupem k polím MQGMO a ujistěte se, že jsou přítomné.

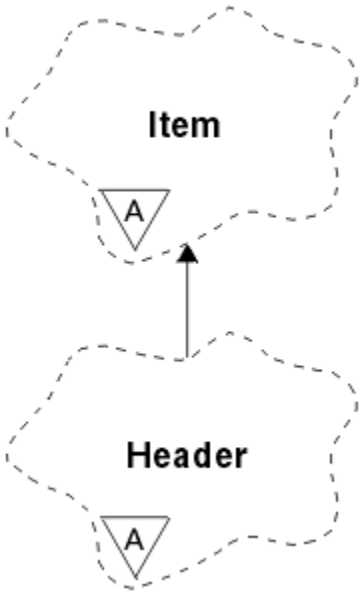
MQLONG *olVersion*
Číslo verze datové struktury MQGMO adresované pomocí *opgmo*.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR

Třída C++ ImqHeader

Tato abstraktní třída zapouzdřuje společné funkce datové struktury MQDLH.



Obrázek 23. Třída ImqHeader

- "Atributy objektu" na stránce 1815
- "Konstruktory" na stránce 1815
- "Metody objektů (veřejné)" na stránce 1815

Atributy objektu

znaková sada
Původní identifikátor kódované znakové sady. Původně MQCCSI_Q_MGR.

kódování
Původní kódování. Původně MQENC_NATIVE.

formát
Původní formát. Zpočátku MQFMT_NONE.

příznaky záhlaví
Počáteční hodnoty jsou:

- Nula pro objekty třídy ImqDeadLetterHeader
- MQIIH_NONE pro objekty třídy ImqIMSBridgeHeader
- Objekt 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 & ***header*** **);**
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqHeader & ***header*** **);**
Zkopíruje data instance ze záhlaví *header*, přičemž nahradí existující data instance.

virtuální MQLONG characterSet () const ;
Vrací **znakovou sadu**.

virtuální void setCharacterSet (const MQLONG *ccsid* = MQCCSI_Q_MGR);
Nastavuje **znakovou sadu**.

virtuální MQLONG encoding () const ;
Vrací hodnotu **encoding**.

virtuální void setEncoding (const MQLONG *encoding* = MQENC_NATIVE);
Nastavuje **kódování**.

virtual ImqString formát () const ;
Vrací kopii **formátu**včetně koncových mezer.

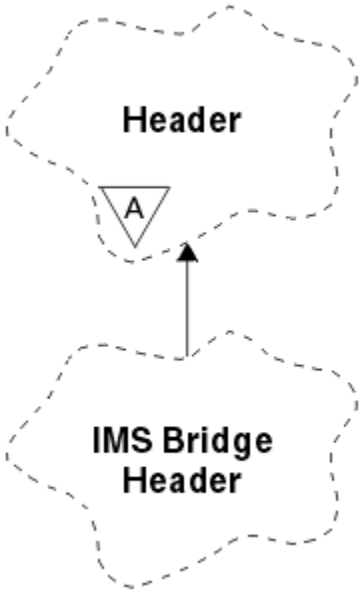
virtuální void setFormat (const char * *name* = 0);
Nastavuje **formát**vyplněný na 8 znaků s koncovými mezerami.

virtuální MQLONG headerFlags () const ;
Vrací parametry **header flags**.

virtuální void setHeaderFlags (const MQLONG *flags* = 0);
Nastaví **příznaky záhlaví**.

Parametr ImqIMSBridgeHeader C++

Tato třída zapouzdřuje funkce datové struktury MQIIH.



Obrázek 24. Třída ImqIMSBridgeHeader

Objekty této třídy jsou používány aplikacemi, které odesílají zprávy na most IMS prostřednictvím produktu IBM MQ for z/OS.

Poznámka: Znaková sada a kódování ImqHeader musí mít výchozí hodnoty a nesmí být nastaveny na žádné jiné hodnoty.

- "Atributy objektu" na stránce 1817
- "Konstruktory" na stránce 1817
- "Přetížené metody ImqItem" na stránce 1817
- "Metody objektů (veřejné)" na stránce 1817
- "Data objektu (chráněná)" na stránce 1818
- "Kódy příčin" na stránce 1818

Atributy objektu

ověřovatel
RACF heslo nebo přístupový lístek, o délce MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota je MQIAUT_NONE.

režim vázaného zpracování
Režim vázaného zpracování. Další informace o režimech vázaného zpracování produktu IMS naleznete v příručce *OTMA User's Guide* . Počáteční hodnota je MQICM_COMMIT_THEN_SEND. Je možná další hodnota, MQICM_SEND_THEN_COMMIT.

přepsání logického terminálu
Přepsání 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 jakékoli odpovědi, délka MQ_FORMAT_LENGTH. Počáteční hodnota je MQFMT_NONE.

rozsah zabezpečení
Rozsah zpracování zabezpečení produktu IMS . Počáteční hodnota je MQISS_CHECK. Dodatečná hodnota, MQISS_FULL, je možná.

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 dodatečná hodnota, MQITS_IN_CONVERSATION.

Konstruktory

ImqIMSBridgeHeader();
Výchozí konstruktor.

ImqIMSBridgeHeader(const ImqIMSBridgeHeader & *header*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & *msg*);
Vloží datovou strukturu MQIIH do vyrovnávací paměti zpráv na začátku a dále přesune existující data zprávy dále. Nastaví formát *msg* na MQFMT_IMS.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Načte datovou strukturu MQIIH z vyrovnávací paměti zpráv.

Aby bylo úspěšné, zakódování objektu *msg* musí být MQENC_NATIVE. Načtěte zprávy s MQGMO_CONVERT do MQENC_NATIVE.

Aby byla úspěšná, formát ImqMessage musí být MQFMT_IMS.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqIMSBridgeHeader & *header*);
Zkopíruje data instance ze záhlaví *header*, přičemž nahradí existující data instance.

ImqString Ověřovatel () const;
Vrací kopii ověřovatele, který je vyplněn koncovými mezerami na délku MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator (const char * *název*);
Nastavuje ověřovatel.

MQCHAR commitMode () const;
Vrátí režim vázaného zpracování.

void setCommitMode (const MQCHAR *režim*);
Nastavuje režim vázaného zpracování.

ImqString logicalTerminalPotlačit () const;
Vrátí kopii tohoto přepisu logického terminálu.

void setLogicalTerminalOverride (const char * *override*);
Nastavuje přepis logického terminálu.

ImqString messageFormatServicesMapNázev () const;
Vrací kopii názvu mapy služeb formátu zprávy.

void setMessageFormatServicesMapName (const char * *název*);
Nastaví název mapy služeb formátu zpráv.

Formát ImqString replyToFormat () const;
Vrací kopii formátu odpovědi na formát MQ_FORMAT_LENGTH s koncovými mezerami na délku MQ_FORMAT_LENGTH.

void setReplyToFormat (const char * *format*);
Nastaví formát odpovědi na formát, doplněný o koncové mezery na délku MQ_FORMAT_LENGTH.

MQCHAR securityScope () const;
Vrátí rozsah zabezpečení.

void setSecurityScope (const MQCHAR *rozsah*);
Nastavuje rozsah zabezpečení.

ImqBinary transactionInstanceID () const;
Vrací kopii ID instance transakce.

ImqBoolean setTransactionInstanceId (const ImqBinary & *id*);
Nastaví ID instance transakce. Délka dat *token* musí být buď nula, nebo MQ_TRAN_INSTANCE_ID_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setTransactionInstanceId (const MQBYTE16 *id* = 0);
Nastaví ID instance transakce. Parametr *id* může mít hodnotu nula, což je stejné jako určení hodnoty MQITII_NONE. Pokud je *id* nenulová, musí adresovat MQ_TRAN_INSTANCE_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je MQITII_NONE, může být nutné, abyste učinili přetypování, aby se zajistila shoda podpisu, například (MQBYTE *) MQITII_NONE.

MQCHAR transactionState () const;
Vrátí stav transakce.

void setTransactionState (const MQCHAR *stav*);
Nastaví stav transakce.

Data objektu (chráněná)

MQIIH *omqiih*
Datová struktura MQIIH.

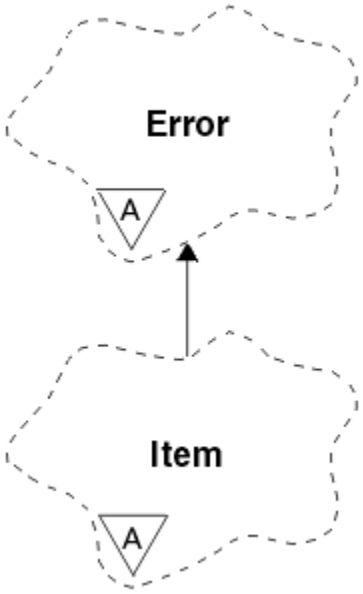
Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT
- CHYBA MQRC_ENCODING_ERROR

- CHYBA MQRC_STRUC_ID_ERROR

Třída C++ ImqItem

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 s ID struktury.

Polymorfní metody v této abstraktní třídě umožňují kopírovat položky do zpráv a ze zpráv. The ImqMessage class **readItem** and **writeItem** methods provide another style of invoking these polymorphic methods that is more natural for application programs.

- "Atributy objektu" na stránce 1819
- "Konstruktory" na stránce 1819
- "Metody třídy (veřejné)" na stránce 1819
- "Metody objektů (veřejné)" na stránce 1820
- "Kódy příčin" na stránce 1820

Atributy objektu

id struktury
Řetězec čtyř znaků na začátku datové struktury. Tento atribut je určen jen pro čtení. Zvažte tento atribut pro odvozené třídy. Není zahrnuta automaticky.

Konstruktory

ImqItem( );
Výchozí konstruktor.

ImqItem(const ImqItem & *položka*);
Kopírovací konstruktor.

Metody třídy (veřejné)

static ImqBoolean structureIdIs (const char * *structure-id-to-test*, const ImqMessage & *msg*);
Vrací TRUE, je-li **ID struktury** další položky ImqItem v příchozí *msg* stejné jako *structure-id-to-test*. Další položka je identifikována jako ta část vyrovnávací paměti zpráv momentálně adresované

serverem ImqCache **ukazatel dat**. Tato metoda spoléhá na **id struktury** , a proto není zaručeno, že bude pracovat pro všechny odvozené třídy ImqItem .

Metody objektů (veřejné)

void operator = (const ImqItem & ***item*** **);**
Zkopíruje data instance z *item*, přičemž nahradí existující data instance.

virtual ImqBoolean copyOut (ImqMessage & ***msg*** **) = 0;**
Zapíše tento objekt jako další položku v odchozí vyrovnávací paměti zpráv a připojí ji ke všem existujícím položkám. Je-li operace zápisu úspěšná, zvýší se ImqCache **délka dat**. Tato metoda vrací TRUE, je-li úspěšná.

Potlačením této metody lze pracovat se specifickou podtřídou.

virtual ImqBoolean pasteIn (ImqMessage & ***msg*** **) = 0;**
Čte tento objekt *destruktivně* z vyrovnávací paměti příchozích zpráv. Čtení je destruktivní v tom, že se ImqCache **ukazatel dat** přesouvá dál. Obsah vyrovnávací paměti však zůstává stejný, takže lze znovu načíst data resetováním ukazatele ImqCache **data pointer**.

Třída (podtřída) tohoto objektu musí být konzistentní s **ID struktury** nalezenou další ve vyrovnávací paměti zpráv objektu *msg* .

Hodnota **encoding** objektu *msg* by měla být MQENC_NATIVE. Doporučuje se načítat zprávy pomocí příkazu ImqMessage **encoding** nastaveným na hodnotu MQENC_NATIVE a s volbami ImqGetMessageOptions **options** včetně MQGMO_CONVERT.

Je-li operace čtení úspěšná, sníží se ImqCache **délka dat** . Tato metoda vrací TRUE, je-li úspěšná.

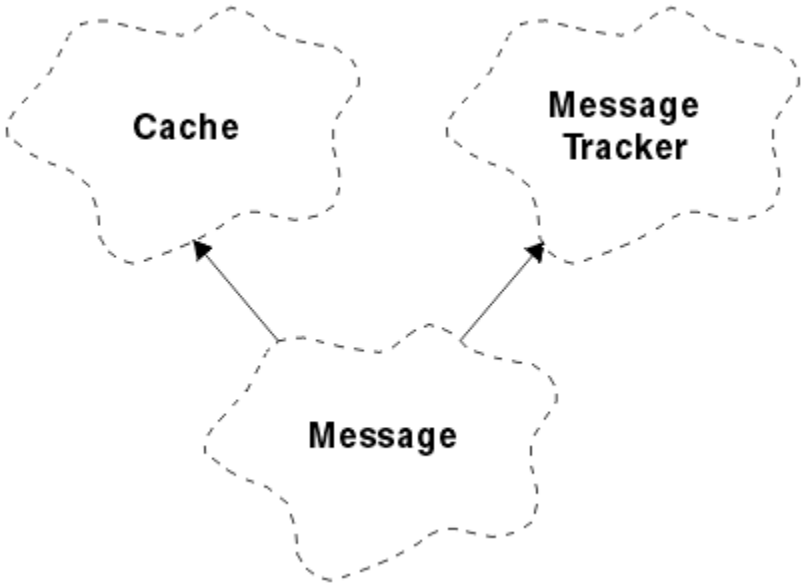
Potlačením této metody lze pracovat se specifickou podtřídou.

Kódy příčin

- CHYBA MQRC_ENCODING_ERROR
- CHYBA MQRC_STRUC_ID_ERROR
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT
- MQRC_INSUFFICIENT_BUFFER
- MQRC_INSUFFICIENT_DATA

Třída C++ ImqMessage

Tato třída zapouzdřuje datovou strukturu MQMD a také zpracovává konstrukci a rekonstrukci dat zprávy.



Obrázek 26. Třída ImqMessage

- "Atributy objektu" na stránce 1821
- "Konstruktory" na stránce 1825
- "Metody objektů (veřejné)" na stránce 1825
- "Metody objektů (chráněné)" na stránce 1827
- "Data objektu (chráněná)" na stránce 1827

Atributy objektu

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:

- MQCSI_INHERIT
- MQCCSI_EMBEDDED

Můžete také použít ID kódované znakové sady dle vašeho výběru. Další informace o tomto tématu naleznete v tématu "Převod kódové stránky" na stránce 929.

kódování
Kódování dat zprávy v počítači. Počáteční hodnota je MQENC_NATIVE.

Vypršení
Množství závislé na čase, které řídí, jak dlouho IBM MQ uchová nenačtenou zprávu, než ji bude vyřazit. Počáteční hodnota je MQEI_UNLIMITED.

formát
Název formátu (šablony), který popisuje rozvržení dat ve vyrovnávací paměti. Názvy delší než osm znaků jsou oříznuty na osm znaků. Názvy jsou vždy doplněny mezerami na osm znaků. Počáteční konstantní hodnota je MQFMT_NONE. Jsou možné následující další konstanty:

- MQFMT_ADMIN
- MQFMT_CICS
- MQFMT_COMMAND_1
- MQFMT_COMMAND_2
- HLAVIČKA MQFMT_DEAD_LETTER_HEADER
- ZÁHLAVÍ MQFMT_DICT_HEADER
- UDÁLOST MQFMT_EVENT
- MQFMT_IMS
- MQFMT_IMS_VAR_STRING
- ROZŠÍŘENÍ MQFMT_MD_EXTENSION
- MQFMT_PCF
- MQFMT_REF_MSG_HEADER
- ZÁHLAVÍ MQFMT_RF_HEADER
- ŘETĚZEC MQFMT_STRING
- SPOUŠTĚČ MQFMT_TRIGGER
- MQFMT_WORK_INFO_HEADER
- ZÁHLAVÍ MQFMT_XMIT_Q_HEADER

Můžete použít také specifický řetězec specifický pro aplikaci. Další informace o tomto tématu naleznete v poli "Formát (MQCHAR8)" na stránce 444 deskriptoru zpráv (MQMD).

Příznaky zprávy
Řídicí informace segmentace. Počáteční hodnota je MQMF_SEGMENTATION_INHIBITED. Jsou možné následující další hodnoty:

- MQMF_SEGMENTATION_ALLOWED
- MQMF_MSG_IN_GROUP
- MQM_LAST_MSG_IN_GROUP
- SEGMENT MQMF_SEGMENT
- MQMF_LAST_SEGMENT
- MQMF_NONE

typ zprávy
Široká kategorizace zprávy. Počáteční hodnota je MQMT_DATAGRAM. Jsou možné následující další hodnoty:

- MQM_SYSTEM_FIRST
- MQM_SYSTEM_LAST
- MQM_DATAGRAM
- POŽADAVEK MQMT_REQUEST
- MQMT_REPLY
- SESTAVA MQMT_REPORT
- MQM_APPL_FIRST
- MQM_APPL_LAST

Můžete také použít specifickou aplikaci, která je specifická pro aplikaci. Další informace o tomto tématu naleznete v poli "MsgType (MQLONG)" na stránce 434 deskriptoru zpráv (MQMD).

posunutí
Informace o posunutí. Počáteční hodnota je nula.

Původní délka
Původní délka segmentované zprávy. Počáteční hodnota je MQOL_UNDEFINED.

trvání, perzistence
Označuje, že zpráva je důležitá a že musí být vždy zálohována pomocí trvalého úložiště. Tato volba implikuje výkon. Počáteční hodnota je MQPER_PERSISTENCE_AS_Q_DEF. Jsou možné následující další hodnoty:

- MQPER_PERSISTENT
- MQPER_NOT_PERSISTENT

priority (priorita)
Relativní priorita pro přenos a doručení. Zprávy se stejnou prioritou se obvykle dodávají ve stejné posloupnosti, jako byly dodány (ačkoliv existuje několik kritérií, která musí být splněna, aby bylo zaručeno). Počáteční hodnota je MQPRI_PRIORITY_AS_Q_DEF.

ověření platnosti vlastnosti
Určuje, zda má být při nastavení vlastnosti zprávy provedeno ověření vlastností. Počáteční hodnota je **MQCMHO_DEFAULT_VALIDATION**. Jsou možné následující další hodnoty:

- MQCMHO_VALIDATE
- MQCMHO_NO_VALIDATION

Následující metody pracují s **validací vlastnosti**:

MQLONG propertyValidation() const;
Vrací 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
- MQAT_CICS
- MOST MQAT_CICS_BRIDGE
- MQAT_DOS
- MQAT_IMS
- MOST MQAT_IMS_BRIDGE
- MQAT_MVS
- MQAT_NOTES_AGENT
- MQAT_OS2
- MQAT_OS390
- MQAT_OS400
- MQAT_QMGR
- MQAT_UNIX
- MQAT_WINDOWS
- POČ MQAT_WINDOWS_NT
- MQAT_XCF
- VÝCHOZÍ HODNOTA MQAT_DEFAULT
- MQAT_UNKNOWN
- MQAT_USER_FIRST

- MQAT_USER_LAST

Můžete použít také specifický řetězec specifický pro aplikaci. Další informace o tomto tématu naleznete v poli “Typ PutAppl(MQLONG)” na stránce 457 deskriptoru zpráv (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, do kterého má být odeslána jakákoli 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:

- VÝJIMKA MQRO_EXCEPTION
- MQRO_EXCEPTION_WIT_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_CED_WITH_DATA
- MQRO_COD_WITH_FULL_DATA *
- MQRO_PAN
- MQRO_NAN
- MQRO_NEW_MSG_ID
- ID_NOVÉ_NOVÉ_ID_ÚLOHY
- MQRO_COPY_MSG_ID_TO_CORREL_ID
- ID_KOLEKCE_MQRO_PASS_RELACE_
- MQRO_DEAD_LETTER_Q
- MQRO_DISCARD_MSG

kde * označuje hodnoty, které nejsou v produktu IBM MQ for z/OSpodporovány.

pořadové číslo
Informace o posloupnosti identifikující zprávu v rámci skupiny. 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 zkrácena, nebo pokud nebyla poslední zpráva přečtena, protože by došlo k oříznutí. Tento atribut je určen jen pro čtení. Počáteční hodnota je nula.

Tento atribut může být užitečný v každé 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*** **);**
Kopírovací konstruktor. Podrobnosti naleznete v metodě **operator =** .

Metody objektů (veřejné)

void operator = (const ImqMessage & ***msg*** **);**
Zkopíruje data MQMD a data zprávy z *msg*. Pokud uživatel pro tento objekt poskytl vyrovnávací paměť, množství zkopírovaných dat je omezeno na dostupnou velikost vyrovnávací paměti. Jinak systém zajistí, aby byla pro zkopírovaná data k dispozici vyrovnávací paměť odpovídající velikosti.

ImqString applicationIdData () const ;
Vrací kopii **dat ID aplikace**.

void setApplicationIdData (const char * ***data*** **= 0);**
Nastavuje **Data ID aplikace**.

ImqString applicationOriginData () const ;
Vrací kopii **původních dat aplikace**.

void setApplicationOriginData (const char * ***data*** **= 0);**
Nastaví **data původu aplikace**.

MQLONG backoutCount () const ;
Vrátí hodnotu **backout count**.

MQLONG characterSet () const ;
Vrací **znakovou sadu**.

void setCharacterSet (const MQLONG ***ccsid*** **= MQCCSI_Q_MGR);**
Nastavuje **znakovou sadu**.

MQLONG encoding () const ;
Vrací hodnotu **encoding**.

void setEncoding (const MQLONG ***encoding*** **= MQENC_NATIVE);**
Nastavuje **kódování**.

MQLONG expirace () const ;
Vrací **vypršení platnosti**.

void setExpiry (const MQLONG ***expiry*** **);**
Nastavuje **vypršení platnosti**.

ImqString formát () const ;
Vrací kopii **formátu**včetně koncových mezer.

ImqBoolean formatIs (const char * ***format-to-test*** **) const ;**
Vrátí hodnotu TRUE, je-li formát **format** stejný jako *format-to-test*.

void setFormat (const char * ***name*** **= 0);**
Nastavuje **formát**vyplněný na osm znaků s koncovými mezerami.

MQLONG messageFlags () const ;
Vrátí **parametry zprávy**.

void setMessageFlags (const MQLONG ***flags*** **);**
Nastavuje **příznaky zpráv**.

MQLONG messageType () const ;
Vrátí **typ zprávy**.

void setMessageType (const MQLONG ***typ*** **);**
Nastaví **typ zprávy**.

MQLONG posun () const ;
Vrací hodnotu **offset**.

void setOffset (const MQLONG *posun*);
Nastavuje **posun**.

MQLONG originalLength () const ;
Vrátí **původní délku**.

void setOriginalLength (const MQLONG *délka*);
Nastavuje **původní délku**.

MQLONG perzistence () const ;
Vrací **trvání**.

void setPersistence (const MQLONG *persistence*);
Nastavuje **trvání**.

MQLONG priorita () const ;
Vrátí **prioritu**.

void setPriority (const MQLONG *priorita*);
Nastavuje **prioritu**.

ImqString putApplicationName () const ;
Vrací kopii příkazu **put application name**.

void setPutApplicationName (const char * *název* = 0);
Nastaví **název aplikace put**.

MQLONG putApplicationType () const ;
Vrátí **typ aplikace put**.

void setPutApplicationType (const MQLONG *typ* = MQAT_NO_CONTEXT);
Nastavuje **typ aplikace put**.

ImqString putDate () const ;
Vrací kopii **data vložení**.

void setPutDate (const char * *date* = 0);
Nastavuje **datum vložení**.

ImqString putTime () const ;
Vrátí kopii parametru **put time**.

void setPutTime (const char * *time* = 0);
Nastavuje **čas vložení**.

ImqBoolean readItem (ImqItem & *item*);
Čte do objektu *item* z vyrovnávací paměti zpráv pomocí metody ImqItem **pasteIn** . Pokud je úspěšný, vrací TRUE.

ImqString replyToQueueManagerName () const ;
Vrací kopii **názvu správce front odpovědi**.

void setReplyToQueueManagerName (const char * *name* = 0);
Nastaví **název správce front pro odpověď**.

ImqString replyToQueueName () const ;
Vrací kopii **názvu fronty pro odpovědi**.

void setReplyToQueueNázev (const char * *název* = 0);
Nastaví **název fronty pro odpovědi**.

MQLONG sestava () const ;
Vrátí **sestavu**.

void setReport (const MQLONG *report*);
Nastaví **sestavu**.

MQLONG sequenceNumber () const ;
Vrátí **pořadové číslo**.

void setSequenceNumber (const MQLONG *číslo*);
Nastavuje **pořadové číslo**.

size_t totalMessageDélka () const ;
Vrátí **celkovou délku zprávy**.

ImqString userId () const ;
Vrací kopii **ID uživatele**.

void setUserId (const char * *id* = 0);
Nastavuje **ID uživatele**.

ImqBoolean writeItem (ImqItem & *item*);
Zapisuje z objektu *item* do vyrovnávací paměti zpráv pomocí metody ImqItem **copyOut** . Zápis může mít formu vložení, nahrazení nebo připojení: to závisí na třídě objektu *item* . Tato metoda vrací TRUE, je-li úspěšná.

Metody objektů (chráněné)

static void setVersionSupported (const MQLONG);
Nastaví **verzi MQMD**. Výchozí nastavení je **MQMD_VERSION_2**.

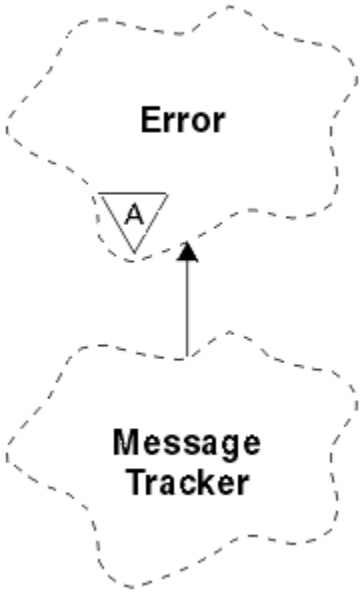
Data objektu (chráněná)

z/OS **MQMD1 *oqmd***
Datová struktura MQMD na systému z/OS.

Multi **MQMD2 *oqmd***
Datová struktura MQMD na systému Multiplatforms.

Třída C++ produktu ImqMessageTracker

Tato třída zapouzdřuje tyto atributy objektu ImqMessage nebo ImqQueue , který může být přidružen k jednomu z objektů.



Obrázek 27. Třída produktu ImqMessageTracker

Tato třída se vztahuje k voláním MQI uvedeným v seznamu “Křížový odkaz ImqMessageTracker” na stránce 1777.

- “Atributy objektu” na stránce 1828
- “Konstruktory” na stránce 1829
- “Metody objektů (veřejné)” na stránce 1829

- “Kódy příčin” na stránce 1830

Atributy objektu

Token evidence
Binární hodnota (MQBYTE32) o délce MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota je MQACT_NONE.

ID korelace
Binární hodnota (MQBYTE24) o délce MQ_CORREL_ID_LENGTH, kterou přiřazujete ke korelaci 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
- PŘETEČENÍ MQFFB_BUFFER_OVERFLOW
- MQFB_LENGTH_OFF_BY_ONE
- CHYBA MQFB_IIH_ERROR
- MQFB_NOT_AUTHORIZED_FOR_IMS
- CHYBA MQFB_IMS_ERROR
- MQFB_IMS_FIRST
- MQFB_IMS_LAST
- MQFB_CICS_APPL_ABENDED
- MQFB_CICS_APPL_NOT_STARTED
- SELHÁNÍ MQFB_CICS_BRIDGE_FAILURE
- CHYBA MQFB_CICS_CCSID_ERROR
- MQFB_CICS_CIH_ERROR
- CHYBA MQFB_CICS_COMMAGA_ERROR
- CHYBA MQFB_CICS_CORREL_ID_ERROR
- CHYBA MQFB_CICS_DLQ_ERROR
- CHYBA MQFB_CICS_ENCODING_ERROR
- MQFB_CICS_INTERNAL_ERROR
- MQFB_CICS_NOT_AUTHORIZED
- MQFB_CICS_UOW_BACKED_OUT
- CHYBA MQFB_CICS_UOW_ERROR

Můžete použít také specifický řetězec specifický pro aplikaci. Další informace o tomto tématu naleznete v poli “Zpětná vazba (MQLONG)” na stránce 438 deskriptoru zpráv (MQMD).

ID skupiny
Binární hodnota (MQBYTE24) s délkou MQ_GROUP_ID_LENGTH jedinečná v rámci fronty. Počáteční hodnota je MQGI_NONE.

ID zprávy
Binární hodnota (MQBYTE24) o délce MQ_MSG_ID_LENGTH jedinečná v rámci fronty. Počáteční hodnota je MQMI_NONE.

Konstruktory

ImqMessageTracker ();
Výchozí konstruktor.

ImqMessageTracker (const ImqMessageTracker & *tracker*);
Kopírovací konstruktor. Podrobnosti naleznete v metodě **operator =** .

Metody objektů (veřejné)

void operator = (const ImqMessageTracker & *tracker*);
Zkopíruje data instance z *trackeru*, nahradí existující data instance.

ImqBinary accountingToken () const ;
Vrací kopii **účtovacího tokenu**.

ImqBoolean setAccountingToken (const ImqBinary & *token*);
Nastaví **účtovací token**. **Délka dat** prvku *token* musí být buď nula, nebo MQ_ACCOUNTING_TOKEN_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setAccountingToken (const MQBYTE32 *token* = 0);
Nastaví **účtovací token**. Parametr *token* může být nula, což je stejné jako určení hodnoty MQACT_NONE. Je-li token *token* nenulový, musí adresovat MQ_ACCOUNTING_TOKEN_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je MQACT_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu; například, (MQBYTE *) MQACT_NONE.

ImqBinary correlationId () const ;
Vrátí kopii identifikátoru **correlation id**.

ImqBoolean setCorrelationId (const ImqBinary & *token*);
Nastaví **ID korelace**. **Délka dat** prvku *token* musí být buď nula, nebo MQ_CORREL_ID_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setCorrelationId (const MQBYTE24 *id* = 0);
Nastaví **ID korelace**. Parametr *id* může mít hodnotu nula, což je stejné jako určení hodnoty MQCI_NONE. Pokud je hodnota *id* nenulová, musí adresovat MQ_CORREL_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je MQCI_NONE, může být třeba, abyste učinili přetypování, abyste zajistili shodu podpisu; například (MQBYTE *) MQCI_NONE.

MQLONG feedback () const ;
Vrátí **zpětnou vazbu**.

void setFeedback (const MQLONG *feedback*);
Nastaví **názor**.

ImqBinary groupId () const ;
Vrátí kopii souboru **group id**.

ImqBoolean setGroupId (const ImqBinary & *token*);
Nastaví **id skupiny**. **Délka dat** prvku *token* musí být buď nula, nebo MQ_GROUP_ID_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setGroupId (const MQBYTE24 *id* = 0);
Nastaví **id skupiny**. Parametr *id* může mít hodnotu nula, což je stejné jako určení hodnoty MQGI_NONE. Pokud je hodnota *id* nenulová, musí adresovat MQ_GROUP_ID_LENGTH bajtů binárních

dat. Při použití předdefinovaných hodnot, jako je například MQGI_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQGI_NONE.

ImqBinary messageId () const ;
Vrací kopii **ID zprávy**.

ImqBoolean setMessageId (const ImqBinary & *token*);
Nastavuje **ID zprávy**. **Délka dat** prvku *token* musí být buď nula, nebo MQ_MSG_ID_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

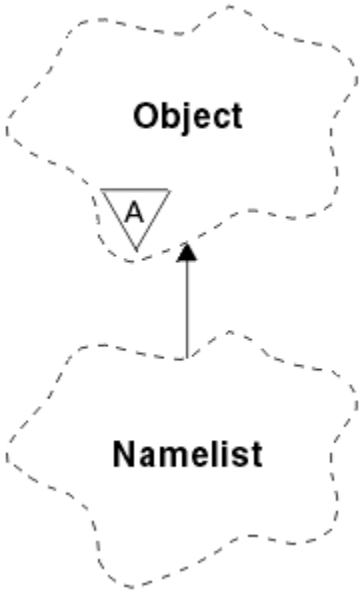
void setMessageId (const MQBYTE24 *id* = 0);
Nastavuje **ID zprávy**. Parametr *id* může být nula, což je stejné jako uvedení MQMI_NONE. Pokud je hodnota *id* nenulová, musí adresovat MQ_MSG_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je například MQMI_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQMI_NONE.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR

Třída C++ ImqNamelist

Tato třída zapouzdřuje seznam názvů.



Obrázek 28. Třída ImqNamelist

Tato třída se vztahuje k voláním MQI uvedeným v seznamu “Křížový odkaz ImqNamelist” na stránce 1778.

Atributy objektu

Počet názvů
Počet názvů objektů v seznamu **názvů seznamů názvů**. Tento atribut je určen jen pro čtení.

názvy seznamů názvů
Názvy objektů, jejichž počet je označen hodnotou **počet názvů**. Tento atribut je určen jen pro čtení.

Konstruktory

ImqNamelist( );
Výchozí konstruktor.

ImqNamelist(const ImqNamelist & ***list*** **);**
Kopírovací konstruktor. ImqObject **open status** má hodnotu false.

ImqNamelist(const char * ***název*** **);**
Nastaví název objektu ImqObject na hodnotu **název**.

Metody objektů (veřejné)

void operator = (const ImqNamelist & ***list*** **);**
Zkopíruje data instance ze seznamu *list*, přičemž nahradí existující data instance. ImqObject **open status** má hodnotu false.

ImqBoolean nameCount(MQLONG & ***počet*** **);**
Poskytuje kopii **počtu názvů**. Pokud je úspěšný, vrací TRUE.

MQLONG nameCount ();
Vrátí **počet názvů** bez uvedení možných chyb.

ImqBoolean namelistName (const MQLONG ***index*****, ImqString &** ***název*** **);**
Poskytuje kopii jednoho seznamu **názvů seznamů názvů** podle indexu založeného na nule. Pokud je úspěšný, vrací TRUE.

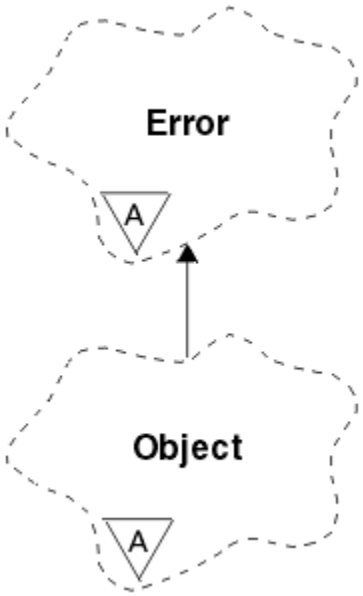
ImqString namelistName (const MQLONG ***index*** **);**
Vrátí jeden z názvů **názvů seznamů názvů** podle indexu založeného na nula bez určení možných chyb.

Kódy příčin

- CHYBA MQRC_INDEX_ERROR
- MQRC_INDEX_NOT_PRESENT

Třída C++ ImqObject

Tato třída je abstraktní. Když je objekt z této třídy zničen, je automaticky uzavřen a jeho připojení ImqQueueManager bylo přerušeno.



Obrázek 29. Třída ImqObject

Tato třída se vztahuje k voláním MQI uvedeným v seznamu “Křížový odkaz ImqObject” na stránce 1778.

Atributy třídy

chování
Řídí chování implicitního otevření.

IMQ_IMPL_OPEN (8L)
Implicitní otevření je povoleno. Toto nastavení je výchozí.

Atributy objektu

Datum změny
Datum změny. Tento atribut je určen jen pro čtení.

Čas změny
Změna času. Tento atribut je určen jen pro čtení.

Jméno alternativního uživatele
Alternativní ID uživatele, až MQ_USER_ID_LENGTH znaků. Počáteční hodnota je řetězec s hodnotou null.

alternativní ID zabezpečení
Alternativní ID zabezpečení. Binární hodnota (MQBYTE40) o délce MQ_SECURITY_ID_LENGTH. Počáteční hodnota je MQSID_NONE.

volby zavření
Volby, které platí, když je objekt uzavřen. Počáteční hodnota je MQCO_NONE. Tento atribut je ignorován během operací implicitního opětovného otevření, kde je vždy použita hodnota MQCO_NONE.

odkaz na připojení
Odkaz na objekt správce ImqQueue, který poskytuje požadované připojení k (lokálnímu) správci front. Pro objekt správce ImqQueueje to objekt samotný. Počáteční hodnota je nula.

Poznámka: Nezaměňujte tento název s názvem správce front, který identifikuje správce front (pravděpodobně vzdáleného) pro pojmenovanou frontu.

description
Popisný název (maximálně 64 znaků) správce front, fronty, seznamu názvů nebo procesu. Tento atribut je určen jen pro čtení.

název
Jméno (maximálně 48 znaků) správce front, fronty, seznamu názvů nebo procesu. Počáteční hodnota je řetězec s hodnotou null. Název modelové fronty se změní po **open** na název výsledné dynamické fronty.

Poznámka: Správce ImqQueuemůže mít název s hodnotou null představující výchozího správce front. Název se změní na skutečného správce front po úspěšném otevření. Seznam ImqDistributionje dynamický a musí mít název s hodnotou null.

další spravovaný objekt
Toto je další objekt této třídy, v žádném konkrétním pořadí, se stejným odkazem na připojení jako tento objekt. Počáteční hodnota je nula.

Volby otevření
Volby, které platí, když je objekt otevřen. Počáteční hodnota je MQOO_INQUIRE. Zde jsou dva způsoby nastavení příslušných hodnot:

1. Nenastavujte otevřené volby a nepoužívejte otevřenou metodu. Produkt IBM MQ automaticky upravuje otevřené volby a automaticky otevírá, znovu otevírá a zavírá objekty podle potřeby. To může vést ke zbytečným novým operacím opětovného otevření, protože produkt IBM MQ používá metodu openFor a toto přidá volby otevřené pouze přírůstkově.
2. Před použitím metod, které vedou k volání MQI, nastavte volby otevření (viz "Křížový odkaz C++ a MQI" na stránce 1771). Tím je zajištěno, že nedojde k zbytečnému znovuotevření operací. Nastavte otevřené volby explicitně, pokud se pravděpodobně vyskytnou některé potenciální problémy se znovuotevřením (viz Znovu otevřít).

   Pokud použijete otevřenou metodu, musíte se ujistit, že volby otevření jsou vhodné jako první. Použití otevřené metody však není povinné; produkt IBM MQ stále vykazuje stejné chování jako v případě 1, ale za těchto okolností je chování efektivní.

Nulová hodnota není platnou hodnotou; před pokusem o otevření objektu nastavte odpovídající hodnotu. To lze provést pomocí **setOpenOptions** (*lOpenOptions*), za nímž následuje **open** (), nebo **openFor** (*lRequiredOpenOption*).

Poznámka:

1. Hodnota MQOO_OUTPUT v metodě **open** pro distribuční seznam je nahrazena MQOO_INQUIRE, protože v tomto okamžiku je jediným platným parametrem **open option** MQOO_OUTPUT. Je však dobrým zvykem vždy nastavit MQOO_OUTPUT explicitně 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, v žádném konkrétním pořadí, má stejný odkaz 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*** **);**
Kopírovací konstruktor. Otevřený stav bude FALSE.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrátí chování.

void setBehavior(const MQLONG ***chování*** **= 0);**
Nastavuje chování.

Metody objektů (veřejné)

void operator = (const ImqObject & ***object*** **);**
Provede zavření, je-li to nutné, a zkopíruje data instance z *objektu*. Otevřený stav bude FALSE.

ImqBoolean alterationDate(ImqString & ***date*** **);**
Poskytuje kopii data změny. Pokud je úspěšný, vrací TRUE.

ImqString alterationDate( );
Vrátí datum změny bez uvedení možných chyb.

ImqBoolean alterationTime(ImqString & ***čas*** **);**
Poskytuje kopii času změny. Pokud je úspěšný, vrací TRUE.

ImqString alterationTime( );
Vrátí čas změny bez uvedení možných chyb.

ImqString alternateUserID () const;
Vrací kopii alternativního ID uživatele.

ImqBoolean setAlternateUserId (const char * ***id*** **);**
Nastavuje alternativní ID uživatele. Alternativní ID uživatele lze nastavit pouze v případě, že otevřený stav je FALSE. Tato metoda vrací TRUE, je-li úspěšná.

ImqBinary alternateSecurityID () const;
Vrací kopii alternativního ID zabezpečení.

ImqBoolean setAlternateSecurityId(const ImqBinary & ***token*** **);**
Nastavuje alternativní ID zabezpečení. Alternativní ID zabezpečení lze nastavit pouze v případě, že otevřený stav je FALSE. Délka dat *token* musí být buď nula, nebo MQ_SECURITY_ID_LENGTH. Pokud je úspěšný, vrací TRUE.

ImqBoolean setAlternateSecurityId(const MQBYTE* ***token*** **= 0);**
Nastavuje alternativní ID zabezpečení. *token* může být nula, což je stejné jako určení MQSID_NONE. Je-li token *token* nenulový, musí adresovat hodnotu MQ_SECURITY_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je MQSID_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu; například (MQBYTE *) MQSID_NONE.

Alternativní ID zabezpečení lze nastavit pouze v případě, že otevřený stav je TRUE. Pokud je úspěšný, vrací TRUE.

ImqBoolean setAlternateSecurityId(const unsigned char * ***id*** **= 0);**
Nastavuje alternativní ID zabezpečení.

ImqBoolean close ();
Nastaví stav otevření na hodnotu FALSE. Pokud je úspěšný, vrací TRUE.

MQLONG closeOptions () const;
Vrátí volby zavření.

void setCloseOptions (const MQLONG ***volby*** **);**
Nastavuje 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 & ***description*** **) = 0;**
Poskytuje kopii popisu. Pokud je úspěšný, vrací TRUE.

Popis ImqString ();
Vrací kopii popisu bez uvedení možných chyb.

virtuální název ImqBoolean (ImqString & ***name*** **);**
Poskytuje kopii názvu. Pokud je úspěšný, vrací TRUE.

Název ImqString name ();
Vrací kopii názvu bez uvedení možných chyb.

ImqBoolean setName (const char * ***název*** **= 0);**
Nastaví název. Název může být nastaven pouze tehdy, je-li otevřený stav FALSE, a pro správce ImqQueue, zatímco stav připojení je FALSE. Pokud je úspěšný, vrací TRUE.

ImqObject * nextManagedObject () const;
Vrátí další spravovaný objekt.

ImqBoolean open ();
Změní otevřený stav na TRUE otevřením objektu podle potřeby, s použitím mimo jiné atributy voleb otevření a názvu. Tato metoda používá referenční informace o připojení a metodu připojení produktu ImqQueueManager, je-li to nezbytné, aby bylo zajištěno, že stav připojení správce ImqQueueManager je TRUE. Vrací otevřený stav.

ImqBoolean openFor (const MQLONG *required-options* = 0);
Pokusí se o to, aby byl objekt otevřený s otevřenými volbami, nebo s otevřenými volbami, které zaručují chování implikované hodnotou parametru *required-option*s .

Je-li *požadované-volby* nula, vstup je povinný a všechny vstupní volby postačuje. Takže, pokud otevřené volby již obsahují jednu z následujících možností:

- MQO_INPUT_AS_Q_DEF
- MQO_INPUT_SHARED
- MQO_INPUT_EXCLUSIVE

Volby otevření jsou již uspokojivé a nejsou změněny; pokud volby otevření již neobsahují žádnou z těchto voleb, MQOO_INPUT_AS_Q_DEF je nastaven v otevřených volbách.

Je-li parametr *required-options* nenulový, jsou požadované volby přidány do voleb otevření. je-li *požadované-volby* některé z těchto voleb, ostatní se resetují.

Pokud se některá z otevřených voleb změní a objekt je již otevřen, je objekt uzavřen dočasně a znovu otevřen, aby bylo možné upravit volby otevření.

Pokud je úspěšný, vrací TRUE. Úspěch označuje, že objekt je otevřený s příslušnými volbami.

MQLONG openOptions () const;
Vrátí volby otevření.

ImqBoolean setOpenVolby (const MQLONG *volby*);
Nastavuje volby otevření. Volby otevření lze nastavit pouze v případě, že otevřený stav je FALSE. Pokud je úspěšný, vrací TRUE.

ImqBoolean openStatus () const;
Vrátí otevřený stav.

ImqObject * previousManagedObject () const;
Vrátí předchozí spravovaný objekt.

Identifikátor ImqBoolean queueManagerIdentifier (ImqString & *id*);
Poskytuje kopii identifikátoru správce front. Pokud je úspěšný, vrací TRUE.

ImqString queueManagerIdentifier ();
Vrátí identifikátor správce front bez uvedení možných chyb.

Metody objektů (chráněné)

virtuální ImqBoolean closeTemporarily ();
Zavře objekt bezpečně před opětovným otevřením. Pokud je úspěšný, vrací TRUE. Tato metoda předpokládá, že otevřený stav je TRUE.

MQHCONN connectionHandle () const;
Vrací MQHCONN přidružený k odkazu na připojení. Tato hodnota je nulová, pokud neexistuje žádný odkaz na připojení, nebo pokud není správce připojen.

ImqBoolean dotázat se (const MQLONG *int-attr*, MQLONG & *value*);
Vrací celočíselnou hodnotu, jejímž indexem je hodnota MQIA_ *. V případě chyby je hodnota nastavena na MQIAV_UNDEFINED.

ImqBoolean dotázat se (const MQLONG *char-attr*, char * & *buffer*, const size_t *délka*);
Vrací znakový řetězec, jehož index je hodnota MQCA_ *.

Poznámka: Obě tyto metody vrací pouze jedinou hodnotu atributu. Pokud je *snímek* požadován více než jednou hodnotou, přičemž hodnoty jsou konzistentní s každou jinou hodnotou pro instant, IBM MQ C + + + toto zařízení neposkytuje a vy musíte použít volání MQINQ s odpovídajícími parametry.

virtuální void openInformationDisperse ();
Disperuje informace z proměnné datové struktury MQOD bezprostředně po volání MQOPEN.

virtual ImqBoolean openInformationPrepare ();
Připravuje informace pro část proměnné datové struktury MQOD bezprostředně před voláním MQOPEN a vrátí hodnotu TRUE, je-li úspěšná.

ImqBoolean set (const MQLONG *int-attr*, const MQLONG *hodnota*);
Nastavuje celočíselný atribut IBM MQ .

ImqBoolean set (const MQLONG *char-attr*, const char * *buffer*, const size_t *required-length*);
Nastaví atribut znaku IBM MQ .

void setNextManagedObject (const ImqObject * *object* = 0);
Nastavuje další spravovaný objekt.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že seznam spravovaných objektů nezlomí.

void setPreviousManagedObject (const ImqObject * *object* = 0);
Nastaví předchozí spravovaný objekt.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že seznam spravovaných objektů nezlomí.

Data objektu (chráněná)

MQHOBJ *ohobj*
Popisovač objektu IBM MQ (je platný pouze, když je otevřený stav TRUE).

MQOD *oqod*
Vestavěná datová struktura MQOD. Množství úložného prostoru přidělené pro tuto datovou strukturu je povinné pro produkt MQOD verze 2. Zkontrolujte číslo verze *(omqod.Version)* a přistupte k dalším polím následujícím způsobem:

MQOD_VERSION_1
Ke všem ostatním polím v souboru *omqod* lze přistupovat.

MQOD_VERSION_2
Ke všem ostatním polím v souboru *omqod* lze přistupovat.

MQOD_VERSION_3
omqod.pmqod je ukazatel na dynamicky alokovaný, větší, MQOD. Žádná jiná pole v souboru *omqod* nejsou přístupná. Ke všem polím adresovaným parametrem *omqod.pmqod* lze přistupovat.

Poznámka: Soubor *omqod.pmqod.Version* může být menší než *omqod.Version*, což značí, že produkt IBM MQ MQI client má více funkcí než server IBM MQ .

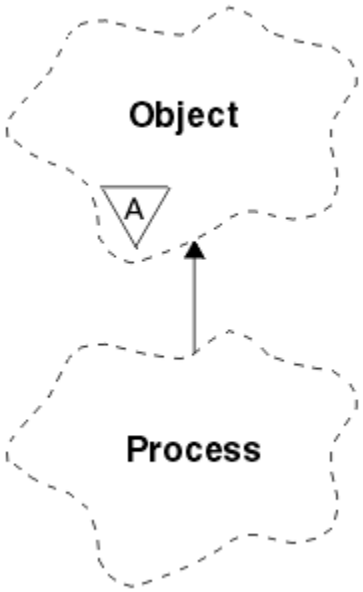
Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- MQRC_INCONSISTENT_OBJECT_STATE
- ODKAZ 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 C++ ImqProcess

Tato třída zapouzdřuje aplikační proces (objekt IBM MQ typu MQOT_PROCESS), který může být spuštěn monitorem spouštěčů.



Obrázek 30. Třída ImqProcess

- "Atributy objektu" na stránce 1837
- "Konstruktory" na stránce 1837
- "Metody objektů (veřejné)" na stránce 1838

Atributy objektu

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 & *process*);
Kopírovací konstruktor. Objekt ImqObject **open status** má hodnotu FALSE.

ImqProcess(const char * *název*);
Nastaví objekt ImqObject **name**.

Metody objektů (veřejné)

void operator = (const ImqProcess & *process*);
Provede zavření, je-li to nezbytné, a pak zkopíruje data instance z *procesu*. Volba ImqObject **open status** bude FALSE.

ImqBoolean applicationId (ImqString & *id*);
Poskytuje kopii **ID aplikace**. Pokud je úspěšný, vrací TRUE.

ImqString applicationId ( );
Vrací **id aplikace** bez uvedení možných chyb.

ImqBoolean applicationType (MQLONG & *typ*);
Poskytuje kopii **aplikačního typu**. Pokud je úspěšný, vrací TRUE.

MQLONG applicationType ();
Vrátí **typ aplikace** bez uvedení možných chyb.

ImqBoolean environmentData (ImqString & *data*);
Poskytuje kopii **dat prostředí**. Pokud je úspěšný, vrací TRUE.

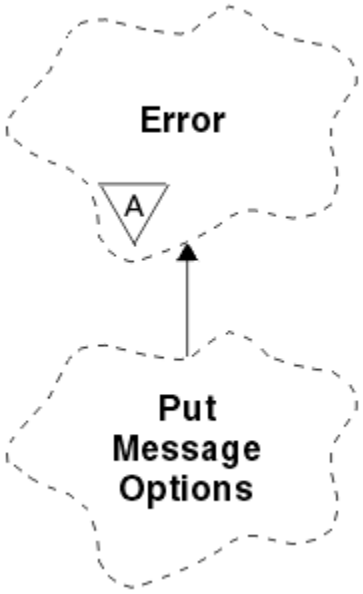
ImqString environmentData ( );
Vrací **data prostředí** bez uvedení možných chyb.

ImqBoolean userData (ImqString & *data*);
Poskytuje kopii **uživatelských dat**. Pokud je úspěšný, vrací TRUE.

ImqString userData ( );
Vrací **uživatelská data** bez uvedení možných chyb.

ImqPutMessageOptions Třída C++

Tato třída zapouzdřuje datovou strukturu MQPMO.



Obrázek 31. Třída ImqPutMessageOptions

- "Atributy objektu" na stránce 1839
- "Konstruktory" na stránce 1839
- "Metody objektů (veřejné)" na stránce 1839
- "Data objektu (chráněná)" na stránce 1840
- "Kódy příčin" na stránce 1840

Atributy objektu

odkaz kontextu
ImqQueue , která poskytuje kontext pro zprávy. Zpočátku zde není žádný odkaz.

volby
Volby vložení zprávy. Počáteční hodnota je MQPMO_NONE. Jsou možné následující další hodnoty:

- MQPMO_SYNCPOINT
- MQPMO_NE_SYNCPOINT
- MQPMO_NOVÉ_ID_ZPRÁVY
- MQPMO_NOVÉ_ KOREL_ID
- MQPMO_LOGICAL_ORDER
- MQPMOTO_NE_KONTEXT
- MQPMO_VÝCHOZÍ_KONTEXT
- KONTEXT MQPMO_PASS_IDENTITY_CONTEXT
- MQPMO_PASS_ALL_CONTEXT
- KONTEXT MQPMO_SET_IDENTITY_CONTEXT
- MQPMO_SET_ALL_CONTEXT
- MQPMO_ALTERNATE_USER_AUTHORITY
- UVÁDĚNÍ MQPMO_FAIL_IF_QUIESCING

pole záznamu
Příznaky, které řídí zahrnutí záznamů vložených zpráv, když je vložena zpráva. Počáteční hodnota je MQPMRF_NONE. Jsou možné následující další hodnoty:

- MQPMRF_ID_ZPRÁVY
- MQPMRF_CORREL_ID
- ID SKUPINY MQPMRF_GROUP_ID
- ZPĚTNÁ VAZBA MQPMRF_FEEDBACK
- MQPMRF_ACCOUNTING_TOKEN

Atributy sledovače ImqMessagejsou převzaty z objektu pro libovolné pole, které je zadané. Atributy sledování ImqMessagejsou převzaty z objektu ImqMessage pro libovolné pole, které není určeno.

vyřešený název správce front
Název správce cílové fronty určeného během vložení. Počáteční hodnota je null. Tento atribut je určen jen pro čtení.

vyřešený název fronty
Název cílové fronty určené během vložení. Počáteční hodnota je null. Tento atribut je určen jen pro čtení.

participace synchronizačního bodu
Nabývá hodnoty True, pokud jsou zprávy uvedeny pod ovládací prvek synchronizačního bodu.

Konstruktory

ImqPutMessageOptions( );
Výchozí konstruktor.

ImqPutMessageOptions(const ImqPutMessageOptions & ***pmo*** **);**
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqPutMessageOptions & ***pmo*** **);**
Zkopíruje data instance z *pmo*a nahradí existující data instance.

ImqQueue * contextReference () const;
Vrátí 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 *volby*);
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 vyřešeného názvu správce front.

ImqString resolvedQueueNázev () const;
Vrací kopii vyřešeného názvu fronty.

ImqBoolean syncPointÚčast () const;
Vrátí hodnotu účasti synchronizačního bodu, která má hodnotu TRUE, pokud tyto volby zahrnují MQPMO_SYNCPOINT.

void setSyncPointParticipation (const ImqBoolean *sync*);
Nastaví hodnotu účasti synchronizačního bodu. Má-li položka *sync* hodnotu TRUE, volby jsou upraveny tak, aby zahrnovaly MQPMO_SYNCPOINT a aby vyloučily MQPMO_NO_SYNCPOINT. Je-li *sync* FALSE, volby jsou upraveny tak, aby zahrnovaly MQPMO_NO_SYNCPOINT a aby vyloučily MQPMO_SYNCPOINT.

Data objektu (chráněná)

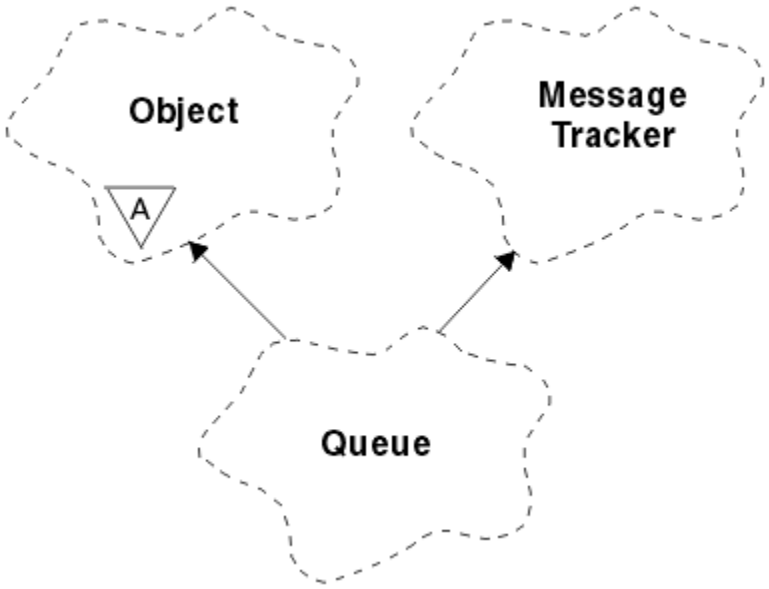
MQPMO *omqpmo*
Datová struktura MQPMO.

Kódy příčin

- MQRC_STORAGE_NOT_AVAILABLE

Třída C++ ImqQueue

Tato třída zapouzdřuje frontu zpráv (objekt IBM MQ typu MQOT_Q).



Obrázek 32. Třída ImqQueue

Tato třída se vztahuje k voláním MQI uvedeným v seznamu Tabulka 863 na stránce 1779.

- “Atributy objektu” na stránce 1841
- “Konstruktory” na stránce 1844
- “Metody objektů (veřejné)” na stránce 1844
- “Metody objektů (chráněné)” na stránce 1850
- “Kódy příčin” na stránce 1850

Atributy objektu

Zpětné jméno přefrontování
Nadměrný název fronty vrácených zpráv. 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 je alias interpretováno. 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 vytíženíklastru
Úroveň vytíženíklastru. Tento atribut je určen jen pro čtení.

Priorita vytíženíklastru
Priorita vytíženíklastru. 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 trvalost 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 vysoké hloubky
Řídicí atribut pro vysoké události hloubky fronty. Tento atribut je určen jen pro čtení.

horní mez hloubky
Horní mez hloubky fronty. Tento atribut je určen jen pro čtení.

událost dolní meze
Řídicí atribut pro události nízké hloubky fronty. Tento atribut je určen jen pro čtení.

dolní mez hloubky
Dolní mez hloubky fronty. Tento atribut je určen jen pro čtení.

maximální událost hloubky
Řídicí atribut pro maximální počet událostí 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: Když se otevře objekt ImqQueue , otevře se jakýkoli otevřený objekt ImqDistributionList, na který se odkazuje, automaticky se zavře.

distribuční seznamy
Možnost přenosové fronty pro podporu distribučních seznamů. 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 se má ukrýt počet odvolání. Tento atribut je určen jen pro čtení.

Typ indexu
Typ indexu. Tento atribut je určen jen pro čtení.

inhibuje získání
Zda jsou povoleny operace get. Počáteční hodnota je závislá na definici fronty. Tento atribut je platný pouze pro alias nebo lokální frontu.

inhibují put
Zda jsou povoleny operace vložení. Počáteční hodnota je závislá na definici fronty.

Název inicializační fronty
Název inicializační fronty. Tento atribut je určen jen pro čtení.

Maximální hloubka
Maximální povolený počet zpráv ve frontě. Tento atribut je určen jen pro čtení.

Maximální délka zprávy
Maximální délka pro každou zprávu v této frontě, která může být menší než maximální hodnota pro kteroukoli frontu spravovanou přidruženým správcem front. Tento atribut je určen jen pro čtení.

Pořadí doručení zpráv
Určuje, 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.

Je-li objekt v řetězu odstraněn, je předchozí objekt a další objekt aktualizován tak, aby jejich odkazy na distribuovanou frontu již neukazovaly na odstraněný objekt.

nestálá třída zpráv
Úroveň spolehlivosti pro dočasné zprávy zařazené do této fronty. Tento atribut je určen jen pro čtení.

Otevření pro vstup - počet
Počet objektů ImqQueue , které jsou otevřené 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, v žádném konkrétním pořadí, který má stejný **odkaz na distribuční seznam** jako tento objekt. Počáteční hodnota je nula.

Je-li objekt v řetězu odstraněn, je předchozí objekt a další objekt aktualizován tak, aby jejich odkazy na distribuovanou frontu již neukazovaly na odstraněný objekt.

Název procesu
Název definice procesu. Tento atribut je určen jen pro čtení.

Účtování fronty
Úroveň účtovacích informací pro fronty. Tento atribut je určen jen pro čtení.

jméno-správce-front
Název správce front (případně vzdáleného), ve kterém je fronta umístěna. Nezaměňujte správce front s názvem ImqObject **odkaz na připojení**, který odkazuje na (lokální) správce front, který poskytuje 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.

vyřešený název 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.

obor
Obor definice fronty. Tento atribut je určen jen pro čtení.

interval služeb
Interval služby. 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 můľe 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
Řízení 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
Využití. Tento atribut je určen jen pro čtení.

Konstruktory

ImqQueue( );
Výchozí konstruktor.

ImqQueue(const ImqQueue & *fronta*);
Kopírovací konstruktor. Volba ImqObject **open status** bude FALSE.

ImqQueue(const char * *název*);
Nastaví objekt ImqObject **name**.

Metody objektů (veřejné)

void operator = (const ImqQueue & *queue*);
Provede zavření, je-li to nezbytné, a pak kopíruje data instance z *fronty*. Volba ImqObject **open status** bude FALSE.

ImqBoolean backoutRequeue (ImqString & *name*);
Poskytuje kopii **názvu fronty vrácených zpráv**. Pokud je úspěšný, vrací TRUE.

ImqString backoutRequeue ();
Vrací **název fronty vrácených zpráv** bez uvedení možných chyb.

ImqBoolean backoutThreshold (MQLONG & *threshold*);
Poskytuje kopii **prahu vrácení**. Pokud je úspěšný, vrací TRUE.

MQLONG backoutThreshold ();
Vrací hodnotu **práh vrácení** bez uvedení možných chyb.

ImqBoolean baseQueueNázev (ImqString & *název*);
Poskytuje kopii **základního názvu fronty**. Pokud je úspěšný, vrací TRUE.

ImqString baseQueueNázev ();
Vrací **základní název fronty** bez uvedení možných chyb.

ImqBoolean clusterName(ImqString & *název*);
Poskytuje kopii **názvu klastru**. Pokud je úspěšný, vrací TRUE.

ImqString clusterName( );
Vrací **název klastru** bez uvedení možných chyb.

ImqBoolean clusterNamelistName (ImqString & *name*);
Poskytuje kopii **názvu seznamu názvů klastru**. Pokud je úspěšný, vrací TRUE.

ImqString clusterNamelistNázev ();
Vrátí **název seznamu názvů klastru** bez uvedení chyb.

ImqBoolean clusterWorkLoadPriority (MQLONG & priority);
Poskytuje kopii hodnoty priority zátěže klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkLoadPriority ();
Vrátí hodnotu priority pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean clusterWorkLoadRank (MQLONG & rank);
Poskytuje kopii hodnoty ohodnocení důležitosti pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkLoadRank ();
Vrátí hodnotu očíslování pořadí zátěže klastru bez uvedení možných chyb.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq);
Poskytuje kopii hodnoty fronty využití pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkLoadUseQ ();
Vrátí hodnotu fronty využití pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean creationDate (ImqString & ***date*** **);**
Poskytuje kopii **data vytvoření**. Pokud je úspěšný, vrací TRUE.

ImqString creationDate ( );
Nevrátí **datum vytvoření** bez uvedení možných chyb.

ImqBoolean creationTime (ImqString & ***time*** **);**
Poskytuje kopii **času vytvoření**. Pokud je úspěšný, vrací TRUE.

ImqString creationTime ( );
Navrací **čas vytvoření** bez uvedení možných chyb.

ImqBoolean currentDepth (MQLONG & ***depth*** **);**
Poskytuje kopii **aktuální hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG currentDepth ();
Vrací **aktuální hloubku** bez uvedení možných chyb.

ImqBoolean defaultInputOpenOption (MQLONG & ***volba*** **);**
Poskytuje kopii **výchozí vstupní volby otevření**. Pokud je úspěšný, vrací TRUE.

MQLONG defaultInputOpenOption ();
Vrací **výchozí vstupní volbu vstupu** bez uvedení možných chyb.

ImqBoolean defaultPersistence (MQLONG & ***persistence*** **);**
Poskytuje kopii **výchozí perzistence**. Pokud je úspěšný, vrací TRUE.

MQLONG defaultPersistence ();
Vrací **výchozí trvání** bez uvedení možných chyb.

ImqBoolean defaultPriority (MQLONG & ***priority*** **);**
Poskytuje kopii **výchozí priority**. Pokud je úspěšný, vrací TRUE.

MQLONG defaultPriority ();
Vrátí **výchozí prioritu** bez uvedení možných chyb.

ImqBoolean defaultBind (MQLONG & ***bind*** **);**
Poskytuje kopii **výchozí vazby**. Pokud je úspěšný, vrací TRUE.

MQLONG defaultBind ();
Vrací **výchozí vazbu** bez uvedení možných chyb.

ImqBoolean definitionType (MQLONG & ***typ*** **);**
Poskytuje kopii typu **definiční typ**. Pokud je úspěšný, vrací TRUE.

MQLONG definitionType ();
Vrací **definiční typ** bez uvedení možných chyb.

ImqBoolean depthHighEvent (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení pro **událost vysoké hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG depthHighEvent ();
Vrací stav povolení **události vysoké hloubky** bez uvedení možných chyb.

ImqBoolean depthHighLimit (MQLONG & ***limit*** **);**
Poskytuje kopii **nejvyššího limitu hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG depthHighLimit ();
Vrací hodnotu **depth high limit** bez uvedení možných chyb.

ImqBoolean depthLowEvent (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení pro **událost nízké hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG depthLowudálosti ();
Vrací stav povolení **události nízké hloubky** bez uvedení možných chyb.

ImqBoolean depthLowLimit (MQLONG & ***limit*** **);**
Poskytuje kopii **dolní meze hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG depthLowLimit ();
Vrací hodnotu **depth low limit** bez uvedení možných chyb.

ImqBoolean depthMaximumEvent (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení pro **událost maximální hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG depthMaximumEvent ();
Vrací stav povolení **maximální události hloubky** bez uvedení možných chyb.

Seznam ImqDistributionList * distributionListReference () const ;
Vrací **odkaz na distribuční seznam**.

void setDistributionListReference (ImqDistributionList & ***list*** **);**
Nastaví **odkaz na distribuční seznam**.

void setDistributionListReference (ImqDistributionList * ***list*** **= 0);**
Nastaví **odkaz na distribuční seznam**.

ImqBoolean distributionLists (MQLONG & ***support*** **);**
Poskytuje kopii hodnoty **distribučních seznamů** . Pokud je úspěšný, vrací TRUE.

MQLONG distributionLists ();
Vrátí hodnotu **distribučních seznamů** bez uvedení možných chyb.

ImqBoolean setDistributionLists (podpora const MQLONG ***podpora*** **);**
Nastaví hodnotu **distribučních seznamů** . Pokud je úspěšný, vrací TRUE.

ImqString dynamicQueueNázev () const ;
Vrací kopii **dynamického názvu fronty**.

ImqBoolean setDynamicQueueName (const char * ***název*** **);**
Nastavuje **dynamický název fronty**. **Název dynamické fronty** lze nastavit pouze tehdy, když je ImqObject **open status** FALSE. Pokud je úspěšný, vrací TRUE.

ImqBoolean get (ImqMessage & ***msg*****, ImqGetMessageOptions &** ***options*** **);**
Načítá zprávu z fronty pomocí uvedených *voleb*. Vyvolá metodu ImqObject **openFor** , je-li to nutné, aby se zajistilo, že ImqObject **volby otevření** obsahují buď jednu z hodnot MQOO_INPUT_ *, nebo hodnotu MQOO_BROWSE, v závislosti na *volbách*. Má-li objekt *msg* hodnotu ImqCache **automatic buffer**, vyrovnávací paměť se bude zvětšená a vyroste tak, že bude obsahovat všechny načtené zprávy. Metoda **clearMessage** je vyvolána proti objektu *msg* před načtením.

Tato metoda vrací TRUE, je-li úspěšná.

Poznámka: Výsledkem vyvolání metody je hodnota FALSE, pokud je objekt ImqObject **reason code** MQRC_TRUNCATED_MSG_FAILED, přestože je tento **kód příčiny** klasifikován jako varování. Pokud je přijata oříznutá zpráva, ImqCache **délka zprávy** odpovídá oříznuté délce. V každém případě ImqMessage **celková délka zprávy** označuje počet bajtů, které byly k dispozici.

ImqBoolean get (ImqMessage & ***msg*** **).**
Co se týče předchozí metody, kromě toho, že jsou použity výchozí volby pro získání zprávy.

ImqBoolean get (ImqMessage & ***msg*****, ImqGetMessageOptions &** ***options*****, const size_t** ***velikost-vyrovnávací_paměti*** **);**
Co se týče předchozích dvou metod, kromě toho, že je indikována přepisující *velikost-vyrovnávací-paměti* . Pokud objekt *zpr.* používá ImqCache **automatická vyrovnávací paměť**, je metoda **resizeBuffer** vyvolána na objektu *zpr.* před načtením zprávy a vyrovnávací paměť se dále nezvětšuje tak, aby obsála větší zprávu.

ImqBoolean get (ImqMessage & ***msg*****, const size_t** ***buffer-size*** **);**
Co se týče předchozí metody, kromě toho, že jsou použity výchozí volby pro získání zprávy.

ImqBoolean hardenGetBackout (MQLONG & ***harden*** **);**
Poskytuje kopii hodnoty **harden get backout** . Pokud je úspěšný, vrací TRUE.

MQLONG hardenGetBackout ();
Vrací hodnotu **harden get backout** bez uvedení možných chyb.

ImqBoolean indexType(MQLONG & ***typ*** **);**
Poskytuje kopii **typu indexu**. Pokud je úspěšný, vrací TRUE.

MQLONG indexType();
Vrací **typ indexu** bez uvedení možných chyb.

ImqBoolean inhibitGet (MQLONG & ***inhibit*** **);**
Poskytuje kopii hodnoty **inhibit get** . Pokud je úspěšný, vrací TRUE.

MQLONG inhibitGet ();
Vrátí hodnotu **inhibit get** bez uvedení možných chyb.

ImqBoolean setInhibitGet (const MQLONG ***inhibit*** **);**
Nastaví hodnotu **inhibit get** . Pokud je úspěšný, vrací TRUE.

ImqBoolean inhibitPut (MQLONG & ***inhibit*** **);**
Poskytuje kopii hodnoty **inhibit put** . Pokud je úspěšný, vrací TRUE.

MQLONG inhibitPut ();
Vrací hodnotu **inhibit put** bez uvedení možných chyb.

ImqBoolean setInhibitPut (const MQLONG ***inhibit*** **);**
Nastaví hodnotu **inhibit put** . Pokud je úspěšný, vrací TRUE.

ImqBoolean initiationQueueNázev (ImqString & ***název*** **);**
Poskytuje kopii **názvu inicializační fronty**. Pokud je úspěšný, vrací TRUE.

ImqString initiationQueueNázev ();
Vrátí **název inicializační fronty** bez uvedení možných chyb.

ImqBoolean maximumDepth (MQLONG & ***depth*** **);**
Poskytuje kopii **maximální hloubky**. Pokud je úspěšný, vrací TRUE.

MQLONG maximumDepth ();
Vrací **maximální hloubku** bez uvedení možných chyb.

ImqBoolean maximumMessageLength (MQLONG & ***délka*** **);**
Poskytuje kopii **maximální délky zprávy**. Pokud je úspěšný, vrací TRUE.

MQLONG maximumMessageLength ();
Nevrátí **maximální délku zprávy** bez uvedení možných chyb.

ImqBoolean messageDeliverySequence (MQLONG & ***sequence*** **);**
Poskytuje kopii **posloupnosti doručení zpráv**. Pokud je úspěšný, vrací TRUE.

MQLONG messageDeliverySequence ();
Vrací hodnotu **posloupnosti doručení zprávy** bez uvedení možných chyb.

ImqQueue * nextDistributedQueue () const ;
Vrací **další distribuovanou frontu**.

ImqBoolean nonPersistentMessageClass (MQLONG & monq);
Poskytuje kopii netrvalé hodnoty třídy zpráv. Pokud je úspěšný, vrací TRUE.

MQLONG nonPersistentMessageClass ();
Vrací hodnotu nestálé třídy zprávy bez uvedení možných chyb.

ImqBoolean openInputCount (MQLONG & ***count*** **);**
Poskytuje kopii **počtu otevřených vstupů**. Pokud je úspěšný, vrací TRUE.

MQLONG openInputCount ();
Vrátí **počet otevřených vstupů** bez uvedení možných chyb.

ImqBoolean openOutputCount (MQLONG & ***count*** **);**
Poskytuje kopii **počtu otevřených výstupů**. Pokud je úspěšný, vrací TRUE.

MQLONG openOutputCount ();
Vrátí **počet otevřených výstupů** bez uvedení možných chyb.

ImqQueue * previousDistributedQueue () const ;
Vrací **předchozí distribuovanou frontu**.

ImqBoolean processName (ImqString & ***name*** **);**
Poskytuje kopii procesu **název procesu**. Pokud je úspěšný, vrací TRUE.

ImqString processName ( );
Vrací **název procesu** bez uvedení možných chyb.

ImqBoolean put (ImqMessage & ***msg*** **);**
Umístí do fronty zprávu s použitím výchozích voleb vkládání zpráv. Používá metodu ImqObject **openFor** , je-li to nezbytné k zajištění, že ImqObject **volby otevření** zahrnují MQOO_OUTPUT.

Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean put (ImqMessage & ***msg*****, ImqPutMessageOptions &** ***pmo*** **);**
Umístí zprávu do fronty pomocí zadané hodnoty *pmo*. Používá metodu ImqObject **openFor** , která je nezbytná k zajištění toho, že ImqObject **volby otevření** zahrnují MQOO_OUTPUT a (pokud *pmo* **volby** zahrnují jakékoli hodnoty MQPMO_PASS_IDENTITY_CONTEXT, MQPMO_PASS_ALL_CONTEXT, MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT) odpovídající hodnoty MQOO_ * _CONTEXT.

Tato metoda vrací TRUE, je-li úspěšná.

Poznámka: Pokud *pmo* obsahuje **kontextový odkaz**, je odkazovaný objekt otevřený, je-li to nezbytné, aby poskytl kontext.

ImqBoolean queueAccounting (MQLONG & acctq);
Poskytuje kopii účetní hodnoty fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueAccounting ();
Vrátí hodnotu účtování fronty bez uvedení možných chyb.

ImqString queueManagerNázev () const ;
Vrací **název správce front**.

ImqBoolean setQueueManagerName (const char * ***název*** **);**
Nastavuje **název správce front**. Objekt **název správce front** lze nastavit pouze v případě, že je hodnota ImqObject **stav otevření** FALSE. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean queueMonitoring (MQLONG & monq);
Poskytuje kopii hodnoty monitorování fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueMonitoring ();
Vrátí hodnotu monitorování fronty bez uvedení možných chyb.

ImqBoolean queueStatistics (MQLONG & statq);
Poskytuje kopii hodnoty statistiky fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueStatistics ();
Vrátí hodnotu statistiky fronty bez uvedení možných chyb.

ImqBoolean queueType (MQLONG & ***typ*** **);**
Poskytuje kopii hodnoty **typu fronty** . Pokud je úspěšný, vrací TRUE.

MQLONG queueType ();
Vrací **typ fronty** bez uvedení možných chyb.

ImqBoolean remoteQueueManagerName (ImqString & ***name*** **);**
Poskytuje kopii **názvu vzdáleného správce front**. Pokud je úspěšný, vrací TRUE.

ImqString remoteQueueManagerName ( );
Vrátí **název vzdáleného správce front** bez uvedení možných chyb.

ImqBoolean remoteQueueNázev (ImqString & ***název*** **);**
Poskytuje kopii **vzdáleného názvu fronty**. Pokud je úspěšný, vrací TRUE.

ImqString remoteQueueNázev ();
Vrací **název vzdálené fronty** bez uvedení možných chyb.

ImqBoolean resolvedQueueManagerName(ImqString &*name*);
Poskytuje kopii **vyřešeného názvu správce front**. Pokud je úspěšný, vrací TRUE.

Poznámka: Tato metoda selže, pokud MQOO_RESOLVE_NAMES není mezi volbami ImqObject **open options**.

ImqString resolvedQueueManagerName( ) ;
Vrací **vyřešený název správce front**bez uvedení možných chyb.

ImqBoolean resolvedQueueNázev (ImqString &*name*);
Poskytuje kopii **vyřešeného názvu fronty**. Pokud je úspěšný, vrací TRUE.

Poznámka: Tato metoda selže, pokud MQOO_RESOLVE_NAMES není mezi volbami ImqObject **open options**.

Název ImqString resolvedQueueName ();
Vrací **vyřešený název fronty**, bez uvedení možných chyb.

ImqBoolean retentionInterval (MQLONG &*interval*);
Poskytuje kopii **intervalu uchování**. Pokud je úspěšný, vrací TRUE.

MQLONG retentionInterval ();
Vrátí **interval uchování** bez uvedení možných chyb.

ImqBoolean scope (MQLONG &*scope*);
Poskytuje kopii **rozsahu**. Pokud je úspěšný, vrací TRUE.

MQLONG rozsah ();
Vrátí **rozsah** bez uvedení možných chyb.

ImqBoolean serviceInterval (MQLONG &*interval*);
Poskytuje kopii **servisního intervalu**. Pokud je úspěšný, vrací TRUE.

MQLONG serviceInterval ();
Vrací **servisní interval** bez uvedení možných chyb.

ImqBoolean serviceIntervalUdálost (MQLONG &*událost*);
Poskytuje kopii stavu povolení **události servisního intervalu**. Pokud je úspěšný, vrací TRUE.

MQLONG serviceIntervalEvent ();
Vrací stav povolení **události servisního intervalu** bez uvedení možných chyb.

ImqBoolean shareability (MQLONG &*shareability*);
Poskytuje kopii hodnoty **shareability** . Pokud je úspěšný, vrací TRUE.

MQLONG shareability ();
Vrátí hodnotu **shareability** bez uvedení možných chyb.

ImqBoolean storageClass(ImqString &*třída*);
Poskytuje kopii **paměťové třídy**. Pokud je úspěšný, vrací TRUE.

ImqString storageClass( );
Vrací **paměťovou třídu** bez uvedení možných chyb.

ImqBoolean transmissionQueueNázev (ImqString &*název*);
Poskytuje kopii **názvu přenosové fronty**. Pokud je úspěšný, vrací TRUE.

ImqString transmissionQueueName ();
Vrátí **název přenosové fronty** bez uvedení možných chyb.

ImqBoolean triggerControl (MQLONG &*control*);
Poskytuje kopii hodnoty **ovladače triggeru** . Pokud je úspěšný, vrací TRUE.

MQLONG triggerControl ();
Vrací hodnotu **ovladače triggeru** bez uvedení možných chyb.

ImqBoolean setTriggerControl (const MQLONG *control*);
Nastavuje hodnotu **ovladače triggeru** . Pokud je úspěšný, vrací TRUE.

ImqBoolean triggerData (ImqString & ***data*** **);**
Poskytuje kopii **aktivačních dat**. Pokud je úspěšný, vrací TRUE.

ImqString triggerData ( );
Vrací kopii **aktivačních dat** bez uvedení možných chyb.

ImqBoolean setTriggerData (const char * ***data*** **);**
Nastavuje **data spouštěče**. Pokud je úspěšný, vrací TRUE.

ImqBoolean triggerDepth (MQLONG & ***depth*** **);**
Poskytuje kopii **hloubky spouštěče**. Pokud je úspěšný, vrací TRUE.

MQLONG triggerDepth ();
Vrací **hloubku spouštěče** bez uvedení možných chyb.

ImqBoolean setTriggerDepth (const MQLONG ***depth*** **);**
Nastavuje **hloubku spouštěče**. Pokud je úspěšný, vrací TRUE.

ImqBoolean triggerMessagePriority (MQLONG & ***priority*** **);**
Poskytuje kopii **priority zpráv spouštěcího impulsu**. Pokud je úspěšný, vrací TRUE.

MQLONG triggerMessagePriority ();
Vrátí **prioritu zprávy spouštěče** bez uvedení možných chyb.

ImqBoolean setTriggerMessagePriority (const MQLONG ***priorita*** **);**
Nastaví **prioritu zprávy spouštěče**. Pokud je úspěšný, vrací TRUE.

ImqBoolean triggerType (MQLONG & ***typ*** **);**
Poskytuje kopii **typu spouštěče**. Pokud je úspěšný, vrací TRUE.

MQLONG triggerType ();
Vrací **typ spouštěče** bez uvedení možných chyb.

ImqBoolean setTriggerTyp (const MQLONG ***typ*** **);**
Nastavuje **typ spouštěče**. Pokud je úspěšný, vrací TRUE.

ImqBoolean usage (MQLONG & ***usage*** **);**
Poskytuje kopii hodnoty **použití** . Pokud je úspěšný, vrací TRUE.

MQLONG použití ();
Vrací hodnotu **usage** bez uvedení možných chyb.

Metody objektů (chráněné)

void setNextDistributedQueue (ImqQueue * ***queue*** **= 0);**
Nastavuje **další distribuovanou frontu**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší rozdělený seznam front.

void setPreviousDistributedQueue (ImqQueue * ***queue*** **= 0);**
Nastaví **předchozí distribuovanou frontu**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nepřeruší rozdělený seznam front.

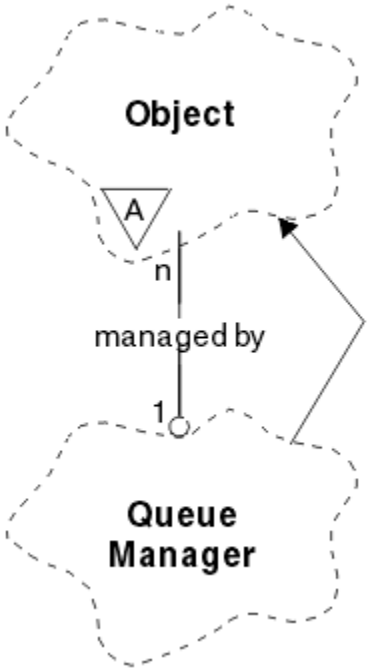
Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- MQRC_CONTEXT_OBJECT_NOT_VALID
- MQRC_CONTEXT_OPEN_ERROR
- MQRC_CURSOR_NOT_VALID
- MQRC_NO_BUFFER
- CHYBA MQRC_REOPEN_EXCL_INPUT_ERROR
- MQRC_REOPEN_INQUIRE_ERROR
- MQRC_REOPEN_TEMPORARY_Q_ERROR
- (kódy příčiny z MQGET)

- (kódy příčiny z MQPUT)

Třída C++ správce ImqQueue

Tato třída zapouzdřuje správce front (objekt IBM MQ typu MQOT_Q_MMGR).



Obrázek 33. Třída správce ImqQueueManager

Tato třída se vztahuje k voláním MQI uvedeným v seznamu “Křížový odkaz správce ImqQueue” na stránce 1782. Ne všechny uvedené metody jsou použitelné na všechny platformy; další podrobnosti viz ALTER QMGR .

Atributy třídy

chování
Řídí chování implicitního připojení a odpojení.

Hodnota IMQ_EXPL_DISC_BACOUT (0L)
Explicitní volání metody disconnect znamená vrácení zpět. Tento atribut se vzájemně vylučuje s hodnotou IMQ_EXPL_DISC_COMMIT.

IMQ_EXPL_DISC_COMMIT (1L)
Explicitní volání metody odpojení znamená potvrzení (výchozí nastavení). Tento atribut se vzájemně vylučuje s hodnotou IMQ_EXPL_DISC_BACOUT.

IMQ_IMPL_CONN (2L)
Implicitní připojení je povoleno (výchozí nastavení).

IMQ_IMPL_DISC_BACOUT (0L)
Implicitní volání metody odpojení, které může nastat při zničení objektu, znamená vrácení zpět. Tento atribut se vzájemně vylučuje s hodnotou IMQ_IMPL_DISC_COMMIT.

IMQ_IMPL_DISC_COMMIT (4L)
Implicitní volání metody odpojení, které se může vyskytnout během zničení objektu, implikuje potvrzení (výchozí nastavení). Tento atribut se vzájemně vylučuje s IMQ_IMPL_DISC_BACOUT.

Ve verzi IBM MQ V7.0 a vyšších jsou aplikace C + +, které využívají implicitní připojení, nutné zadat IMQ_IMPL_CONN spolu s dalšími volbami poskytnutými v metodě `setBehavior()` na objektu třídy `ImqQueueManager`. Pokud vaše aplikace nepoužívá metodu produktu `setBehavior()` k explicitnímu nastavení voleb chování, například

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

Tato změna nemá vliv na to, že hodnota MQ_IMPL_CONN je ve výchozím nastavení povolena.

Pokud vaše aplikace výslovně nastavuje volby chování, například,

```
ImqQueueManager_object.setBehavior(IMQ_IMPL_DISC_COMMIT)
```

Musíte zahrnout parametr 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 objektu

potlačení evidence připojení
Umožňuje aplikacím potlačit nastavení evidence MQI a evidence front values.This . Atribut je určen pouze pro čtení.

Interval evidence
Jak dlouho před zápisem intermediačních záznamů evidence (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
Zaškrtnuté prvky určují, zda má být při zjištění nového příchozího kanálu s názvem MCA, který je již aktivní, přijat nový příchozí kanál MCA. Tento atribut je určen jen pro čtení.

Převzetí nového agenta MCA - typ
Zda má být osamocená instance MCA určitého typu kanálu automaticky restartována, když je zjištěn nový požadavek příchozího kanálu odpovídající převzetí nových parametrů kontroly mca pro převzetí. Tento atribut je určen jen pro čtení.

Typ ověřování
Označuje typ ověření, které se provádí.

událost oprávnění
Řídí události oprávnění. Tento atribut je určen jen pro čtení.

volby začátku
Volby, které se použijí na počáteční metodu. Počáteční hodnota je MQBO_NONE.

událost mostu
Zda se generují 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 se generují události kanálu. Tento atribut je určen jen pro čtení.

Adaptéry inicializátoru kanálu
Počet podúloh adaptéru, které mají být použity pro zpracování volání produktu IBM MQ . Tento atribut je určen jen pro čtení.

Řízení iniciátoru kanálu
Určuje, zda má být iniciátor kanálu spuštěn automaticky při spuštění správce front. Tento atribut je určen jen pro čtení.

Dispečeři inicializátoru kanálu
Počet dispečerů, který má být použit pro inicializátor kanálu. Tento atribut je určen jen pro čtení.

automatické spuštění trasování inicializátoru kanálu
Určuje, zda má být trasování inicializátoru kanálu spuštěno automaticky či nikoli. Tento atribut je určen jen pro čtení.

Velikost tabulky trasování inicializátoru kanálu
Velikost prostoru pro 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 tento atribut může být nastaven 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
Ovládá shromažďování online monitorovacích dat 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 pro pracovní zátěž klastru
Pracovní zátěž klastru naposledy použitá hodnota kanálů. 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 příkazů. Tento atribut je určen jen pro čtení.

Název fronty vstupu příkazů
Název vstupní fronty příkazu systému. 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 spuštěn automaticky při spuštění správce front. Tento atribut je určen jen pro čtení.

Volby připojení
Volby, které se použijí na metodu připojení. Počáteční hodnota je MQCNO_NONE. V závislosti na platformě mohou být možné následující další hodnoty:

- VAZBA MQCNO_STANDARD_BINDING
- 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
Hodnota TRUE při připojení ke správci front. Tento atribut je určen jen pro čtení.

Značka připojení
Značka, která má být přidružena k připojení. Tento atribut může být nastaven pouze tehdy, když není připojen. Počáteční hodnota je null.

Kryptografický hardware
Podrobnosti konfigurace kryptografického hardwaru. Pro připojení klienta MQ MQI.

název fronty nedoručených zpráv
Název fronty nedoručených zpráv. Tento atribut je určen jen pro čtení.

výchozí název 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, kterou se má připojit modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, při použití podpory služeb DNS (Dynamic Domain Name Services) správce pracovní zátěže. Tento atribut je určen jen pro čtení.

dns wlm
Určuje, zda má být modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registrován ve správci pracovní zátěže pro služby DNS (Dynamic Domain Name Services). Tento atribut je určen jen pro čtení.

záznam prvního ověření
První z jednoho nebo více objektů třídy ImqAuthenticationRecord, v žádném konkrétním pořadí, ve kterém se odkaz na připojení záznamu ImqAuthenticationadresuje tento objekt. Pro připojení klienta MQ MQI.

první spravovaný objekt
První z jednoho nebo více objektů třídy ImqObject, v žádném konkrétním pořadí, ve kterém se odkaz na připojení ImqObject adresuje tomuto objektu. Počáteční hodnota je nula.

blokace události
Obslužné prvky inhibují události. 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 resetování klíče
Počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace TLS, než je znovu vyjednán tajný klíč. Tento atribut se používá pouze pro připojení klienta pomocí MQCONNX. Viz též ssl key reset count.

Časovač modulu listener
Časový interval (v sekundách) mezi pokusy pomocí 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 o zotavení. Tento atribut je určen jen pro čtení.

Název skupiny LU
Generický název LU, který má používat modul listener LU 6.2 , který zpracovává příchozí přenosy pro skupinu sdílení front, by měl být použit. Tento atribut je určen jen pro čtení.

Název jednotky LU
Název jednotky LU, která má být použita pro odchozí přenosy LU 6.2 . Tento atribut je určen jen pro čtení.

Přípona ramena lu62
Přípona SYS1.PARMLIB člen APPCPMxx, který nominuje LUADD pro tento inicializátor kanálu. Tento atribut je určen jen pro čtení.

lu62 kanály
Maximální počet kanálů, které mohou být aktuální nebo klienti, kteří mohou být připojeni, a které používají přenosový protokol LU 6.2 . Tento atribut je určen jen pro čtení.

maximum 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 úchyty
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.

maximum odchozího portu
Vyšší konec rozsahu čísel portů, které mají být použity při vázání odchozích kanálů. Tento atribut je určen jen pro čtení.

minimální odchozí port
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 správce front umístěn. 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
Zhruba, jak dlouho bude kanál zpráv TCP/IP čekat na příjem dat, včetně synchronizačních signálů od svého partnera, než se vrátí do neaktivního stavu. Tento atribut je určen jen pro čtení.

minimální časový limit příjmu
Minimální doba, po kterou bude kanál TCP/IP čekat na příjem dat, včetně synchronizačních signálů od svého partnera, než se vrátí do neaktivního stavu. Tento atribut je určen jen pro čtení.

Typ časového limitu pro příjem
Kvalifikátor použitý k vypršení časového limitu pro příjem. 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 správce sdílené fronty
Určuje, zda má operace MQOPEN sdílené fronty, ve které je název ObjectQMgrjiného správce front v rámci skupiny sdílení front, být v lokálním správci front rozpoznán jako otevření sdílené fronty. Tento atribut je určen jen pro čtení.

událost ssl
Zda se generují události SSL. Tento atribut je určen jen pro čtení.

Požadován standard SSL FIPS
Zda se mají použít pouze algoritmy certifikovaný FIPS, pokud se šifrování provádí v softwaru IBM MQ . Tento atribut je určen jen pro čtení.

Počet resetování klíče SSL
Počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL před opětovným vyjednávanstvím tajného klíče. Tento atribut je určen jen pro čtení.

událost start-stop
Řídí start-stop události. Tento atribut je určen jen pro čtení.

Interval statistiky
Jak často jsou data monitorování statistiky zapsána do fronty monitorování. Tento atribut je určen jen pro čtení.

Dostupnost synchronizačního bodu
Dostupnost synchronizace synchronizačního bodu. Tento atribut je určen jen pro čtení.

Poznámka: Správci front-koordinované globální pracovní jednotky nejsou na platformě IBM i podporovány. IBM i Můžete naprogramovat jednotku práce, externě koordinovanou pomocí IBM i, pomocí nativních systémových volání _Rcommit a _Rback. Spusťte tento typ jednotky práce tak, že spustíte aplikaci IBM MQ pod vázaným zpracováním na úrovni úlohy pomocí příkazu STRCMTCTL. Další podrobnosti naleznete v tématu Rozhraní pro externího správce synchronizačního bodu produktu

IBM i . Odvrácení a potvrzení jsou podporovány na platformě IBM i pro lokální jednotky práce koordinované správcem front.

kanály tcp
Maximální počet kanálů, které mohou být aktuální nebo klienti, kteří mohou být připojeni a které používají přenosový protokol TCP/IP. Tento atribut je určen jen pro čtení.

TCP - Udržování aktivity
Zda se má služba TCP KEEPALIVE používat ke kontrole toho, zda je druhý konec připojení stále dostupný. Tento atribut je určen jen pro čtení.

Název TCP
Název jediného nebo výchozího systému TCP/IP, který má být použit, v závislosti na hodnotě typu zásobníku tcp. Tento atribut je určen jen pro čtení.

Typ sady protokolů TCP
Určuje, zda má iniciátor kanálu povoleno používat pouze adresní prostor TCP/IP určený v názvu tcp, nebo se může vázat na libovolnou vybranou adresu TCP/IP. Tento atribut je určen jen pro čtení.

Záznam přenosových tras
Ovládá záznam informací o trasování přenosové cesty. 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 je to skutečné ID uživatele aplikace. Na platformách Windows je ID uživatele aplikace.

Konstruktory

Správce ImqQueueManager ();
Výchozí konstruktor.

Správce ImqQueueManager (const ImqQueueManager & *manager*);
Kopírovací konstruktor. Stav připojení bude FALSE.

ImqQueueManager (const char * *název*);
Nastaví název objektu ImqObject na hodnotu *název*.

Destruktory

Když je objekt správce ImqQueuezlikvidován, je automaticky odpojen.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrátí chování.

void setBehavior(const MQLONG *chování* = 0);
Nastavuje chování.

Metody objektů (veřejné)

void operator = (const ImqQueueManager & *mgr*);
Odpojí se, je-li to nutné, a kopíruje data instance z *mgr*. Stav připojení je FALSE.

ImqBoolean accountingConnOverride (MQLONG & statint);
Poskytuje kopii hodnoty přepsání evidence připojení. Pokud je úspěšný, vrací TRUE.

MQLONG accountingConnOverride ();
Vrací hodnotu přepsání účtovacích připojení bez udávání možných chyb.

ImqBoolean accountingInterval (MQLONG & statint);
Poskytuje kopii hodnoty intervalu evidence. Pokud je úspěšný, vrací TRUE.

MQLONG accountingInterval ();
Vrátí hodnotu časového intervalu účtování bez udávání možných chyb.

ImqBoolean activityRecording (MQLONG & rec);
Poskytuje kopii hodnoty záznamu aktivity. Pokud je úspěšný, vrací TRUE.

MQLONG activityRecording ();
Vrací hodnotu záznamu aktivity bez uvedení možných chyb.

ImqBoolean adoptNewMCACheck (MQLONG & check);
Poskytuje kopii nové hodnoty kontroly MCA pro převzetí. Pokud je úspěšný, vrací TRUE.

MQLONG adoptNewMCACheck ();
Vrací převzetí nové hodnoty kontroly MCA bez uvedení možných chyb.

ImqBoolean adoptNewMCATType (MQLONG & typ);
Poskytuje kopii nového typu MCA pro převzetí. Pokud je úspěšný, vrací TRUE.

MQLONG adoptNewMCATType ();
Vrací převzetí nového typu MCA bez uvedení možných chyb.

QLONG authenticationType () const;
Vrátí typ ověřování.

void setAuthenticationTyp (const MQLONG type = MQCSP_AUTH_NONE);
Nastavuje typ ověřování.

ImqBoolean authorityEvent(MQLONG & *událost*);
Poskytuje kopii stavu povolení události oprávnění. Pokud je úspěšný, vrací TRUE.

MQLONG authorityEvent();
Vrací stav povolení události oprávnění bez uvedení možných chyb.

ImqBoolean backout ();
Zálohuje nepotvrzené změny. Pokud je úspěšný, vrací TRUE.

ImqBoolean begin ();
Začne jednotku práce. Volby zahájení ovlivňují chování této metody. Vrací TRUE, je-li úspěšný, ale vrací také TRUE, i když volání MQBEGIN vrací MQRC_NO_EXTERNAL_PARTICANTS nebo MQRC_PARTICAANT_NOT_AVAILABLE (které jsou obě přidruženy k funkci MQCC_WARNING).

MQLONG beginOptions() const;
Vrátí volby začátku.

void setBeginVolby (const MQLONG *volby* = MQBO_NONE);
Nastaví počáteční volby.

ImqBoolean bridgeEvent (MQLONG & event);
Poskytuje kopii hodnoty události mostu. Pokud je úspěšný, vrací TRUE.

MQLONG bridgeEvent ();
Vrací hodnotu události mostu bez uvedení možných chyb.

ImqBoolean channelAutoDefinition (MQLONG & *value*);
Poskytuje kopii hodnoty automatické definice kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelAutoDefinition ();
Vrátí hodnotu automatické definice kanálu bez uvedení možných chyb.

ImqBoolean channelAutoDefinitionEvent(MQLONG & *hodnota*);
Poskytuje kopii hodnoty události automatické definice kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelAutoDefinitionEvent();
Vrací hodnotu události automatické definice kanálu bez uvedení možných chyb.

ImqBoolean channelAutoDefinitionExit(ImqString & *name*);
Poskytuje kopii názvu uživatelské procedury automatické definice kanálu. Pokud je úspěšný, vrací TRUE.

ImqString channelAutoDefinitionExit( );
Vrací název uživatelské procedury automatické definice kanálu bez uvedení možných chyb.

ImqBoolean channelEvent (MQLONG & event);
Poskytuje kopii hodnoty události kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelEvent();
Vrátí hodnotu události kanálu bez uvedení možných chyb.

MQLONG channelInitiatorAdapters ();
Vrátí hodnotu adaptéru inicializátoru kanálu bez jakýchkoli informací o možných chybách.

Adaptéry ImqBoolean channelInitiator(MQLONG & adapters);
Poskytuje kopii hodnoty adaptérů inicializátoru kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelInitiatorControl ();
Vrátí hodnotu spuštění inicializátoru kanálu bez jakýchkoli informací o možných chybách.

ImqBoolean channelInitiatorŘízení (MQLONG & init);
Poskytuje kopii spouštěcí hodnoty ovládacího prvku iniciátoru kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelInitiatorDispatcher ();
Vrátí hodnotu dispečerů pro inicializátor kanálu bez jakýchkoli informací o možných chybách.

ImqBoolean channelInitiatorDispečery (MQLONG & dispečeři);
Poskytuje kopii hodnoty dispečerů inicializátoru kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelInitiatorTraceAutoStart ();
Vrátí hodnotu automatického spuštění trasování kanálu kanálu bez uvedení možných chyb.

ImqBoolean channelInitiatorTraceAutoStart (MQLONG & auto);
Poskytuje kopii hodnoty automatického spuštění trasování inicializátoru kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelInitiatorTraceTableSize ();
Vrací hodnotu velikosti tabulky trasování inicializátoru kanálu bez uvedení možných chyb.

ImqBoolean channelInitiatorTraceTableVelikost (MQLONG & size);
Poskytuje kopii hodnoty velikosti tabulky trasování inicializátoru kanálu. Pokud je úspěšný, vrací TRUE.

ImqBoolean channelMonitoring (MQLONG & monchl);
Poskytuje kopii hodnoty monitorování kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelMonitoring ();
Vrátí hodnotu monitorování kanálu bez uvedení možných chyb.

ImqBoolean channelReference(ImqChannel * & ***pchannel*** **);**
Poskytuje kopii odkazu na kanál. Je-li odkaz na kanál neplatný, nastaví *pchannel* na hodnotu null. Tato metoda vrací TRUE, je-li úspěšná.

ImqChannel * channelReference( );
Vrátí odkaz na kanál bez jakýchkoli informací o možných chybách.

ImqBoolean setChannelReference (ImqChannel & ***kanál*** **);**
Nastaví odkaz na kanál. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setChannelReference (ImqChannel * ***kanál*** **= 0);**
Nastaví nebo resetuje odkaz na kanál. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean channelStatistics (MQLONG & statchl);
Poskytuje kopii hodnoty statistiky kanálu. Pokud je úspěšný, vrací TRUE.

MQLONG channelStatistics ();
Vrátí hodnotu statistiky kanálu bez uvedení možných chyb.

ImqBoolean characterSet(MQLONG & ***ccsid*** **);**
Poskytuje kopii znakové sady. Pokud je úspěšný, vrací TRUE.

MQLONG characterSet();
Vrací kopii znakové sady, aniž by došlo k indikaci možných chyb.

MQLONG clientSslKeyResetCount () const;
Vrátí hodnotu počtu resetování klíče SSL použitou u připojení klienta.

void setClientSslKeyResetCount(const MQLONG count);
Nastaví počet obnovení klíčů zabezpečení SSL používaný u připojení klienta.

ImqBoolean clusterSenderMonitoring (MQLONG & monacls);
Poskytuje kopii výchozí hodnoty monitorování odesílatele klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterSenderMonitoring ();
Vrátí výchozí hodnotu monitorování odesílatele klastru bez uvedení možných chyb.

ImqBoolean clusterSenderStatistics (MQLONG & statacls);
Poskytuje kopii hodnoty statistiky odesílatele klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterSenderStatistics ();
Vrátí hodnotu statistiky odesílatele klastru bez uvedení možných chyb.

ImqBoolean clusterWorkloadData (ImqString & ***data*** **);**
Poskytuje kopii dat uživatelské procedury pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

ImqString clusterWorkloadData ();
Vrátí data uživatelské procedury pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean clusterWorkloadExit (ImqString & ***name*** **);**
Poskytuje kopii názvu uživatelské procedury pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

ImqString clusterWorkloadExit ();
Vrátí název uživatelské procedury pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean clusterWorkloadLength (MQLONG & ***délka*** **);**
Poskytuje kopii délky pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkloadLength ();
Vrátí délku pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean clusterWorkLoadMRU (MQLONG & mru);
Poskytuje kopii pracovní zátěže klastru naposledy použitých hodnot kanálů. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkLoadMRU ();
Vrátí pracovní zátěž klastru naposledy použitou hodnotu kanálu bez uvedení možných chyb.

ImqBoolean clusterWorkLoadUseQ (MQLONG & useq);
Poskytuje kopii hodnoty fronty využití pracovní zátěže klastru. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkLoadUseQ ();
Vrátí hodnotu fronty využití pracovní zátěže klastru bez uvedení možných chyb.

ImqBoolean commandEvent (MQLONG & event);
Poskytuje kopii hodnoty události příkazu. Pokud je úspěšný, vrací TRUE.

MQLONG commandEvent ();
Vrací hodnotu události příkazu bez uvedení možných chyb.

ImqBoolean commandInputQueueName(ImqString & ***název*** **);**
Poskytuje kopii názvu vstupní fronty příkazů. Pokud je úspěšný, vrací TRUE.

ImqString commandInputQueueName( );
Vrátí název vstupní fronty příkazu bez uvedení možných chyb.

ImqBoolean commandLevel(MQLONG & ***úroveň*** **);**
Poskytuje kopii úrovně příkazu. Pokud je úspěšný, vrací TRUE.

MQLONG commandLevel();
Vrátí úroveň příkazu bez uvedení možných chyb.

MQLONG commandServerControl ();
Vrátí hodnotu spuštění příkazového serveru bez uvedení možných chyb.

ImqBoolean commandServerControl (MQLONG & server);
Poskytuje kopii spouštěcí hodnoty ovladače příkazového serveru. Pokud je úspěšný, vrací TRUE.

ImqBoolean commit ();
Potvrzené nepotvrzené změny. Pokud je úspěšný, vrací TRUE.

ImqBoolean connect ();
Připojí se ke správci front s daným názvem ImqObject , přičemž výchozí hodnotou je lokální správce front. Chcete-li se připojit ke specifickému správci front, použijte před připojením metodu ImqObject

setName . Pokud se zde nachází odkaz na kanál, používá se k předávání informací o definici kanálu MQCONNX na objekt MQCD. Hodnota ChannelType v objektu MQCD je nastavena na hodnotu MQCHT_CLNTCONN. referenční informace o kanálu, které mají význam pouze pro připojení klienta, jsou pro připojení k serveru ignorovány. Volby připojení ovlivňují chování této metody. Tato metoda nastaví stav připojení na TRUE, je-li úspěšný. Funkce vrátí nový stav připojení.

Pokud existuje první ověřovací záznam, řetězec autentizačních záznamů se použije k ověření digitálních certifikátů pro zabezpečené kanály klienta.

Ke stejnému správci front můžete připojit více než jeden objekt ImqQueueManager. Všechny používají stejný manipulátor připojení MQHCONN a sdílejí funkčnost UOW pro připojení přidružené k podprocesu. První správce ImqQueueManager pro připojení získává popisovač MQHCONN. Poslední operace ImqQueueManager pro odpojení provádí MQDISC.

Pro vícevláknový program se doporučuje, aby byl pro každý podproces použit oddělený objekt ImqQueueManager.

ImqBinary connectionId () const;
Vrátí ID připojení.

ImqBinary connectionTag () const;
Vrátí značku připojení.

ImqBoolean setConnectionZnačka (const MQBYTE128 *tag* = 0);
Nastavuje značku připojení. Je-li příznak *příznak* nula, bude příznak připojení vymazán. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setConnectionZnačka (const ImqBinary & *tag*);
Nastavuje značku připojení. Délka dat značky *tag* musí být nula (pro vymazání značky spojení) nebo MQ_CONN_TAG_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG connectOptions() const;
Vrátí volby připojení.

void setConnectOptions (const MQLONG *volby* = MQCNO_NONE);
Nastavuje volby připojení.

ImqBoolean connectionStatus() const;
Vrátí stav připojení.

ImqString cryptographicHardware ( );
Vrátí kryptografický hardware.

ImqBoolean setCryptographicHardware (const char * *hardware* = 0);
Nastavuje kryptografický hardware. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean deadLetterQueueName(ImqString & *name*);
Poskytuje kopii názvu fronty nedoručených zpráv. Pokud je úspěšný, vrací TRUE.

ImqString deadLetterQueueName( );
Vrátí kopii názvu fronty nedoručených zpráv, aniž by došlo k indikaci možných chyb.

ImqBoolean defaultTransmissionQueueName(ImqString & *name*);
Poskytuje kopii výchozího názvu přenosové fronty. Pokud je úspěšný, vrací TRUE.

ImqString defaultTransmissionQueueName( );
Vrací výchozí název přenosové fronty bez uvedení 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 oddělijí jejich odkaz na připojení. Je-li ke stejnému správci front připojen více než jeden objekt ImqQueueManager, provede fyzické odpojení pouze poslední odpojení; ostatní provedou logické odpojení. Nepotvrzené změny jsou potvrzeny pouze ve fyzickém odpojování.

Tato metoda vrací TRUE, je-li úspěšná. Je-li volán, když neexistuje žádné existující připojení, návratový kód je také pravdivý.

ImqBoolean distributionLists(MQLONG & *podpora*);
Poskytuje kopii hodnoty rozdělovníků. Pokud je úspěšný, vrací TRUE.

MQLONG distributionLists();
Vrátí hodnotu rozdělovníků bez jakýchkoli indikací možných chyb.

ImqBoolean dnsGroup (ImqString & group);
Poskytuje kopii názvu skupiny DNS. Pokud je úspěšný, vrací TRUE.

ImqString dnsGroup ( );
Vrátí název skupiny DNS bez uvedení možných chyb.

ImqBoolean dnsWlm (MQLONG & wlm);
Poskytuje kopii hodnoty DNS WLM. Pokud je úspěšný, vrací TRUE.

MQLONG dnsWlm ();
Vrací hodnotu DNS WLM bez uvedení možných chyb.

Záznam 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 povolení stavu povolení události blokování. Pokud je úspěšný, vrací TRUE.

MQLONG inhibitEvent();
Vrací stav povolení události blokování bez uvedení možných chyb.

ImqBoolean ipAddressVerze (MQLONG & verze);
Poskytuje kopii hodnoty verze adresy IP. Pokud je úspěšný, vrací TRUE.

MQLONG ipAddressVerze ();
Vrátí hodnotu verze adresy IP bez uvedení možných chyb.

ImqBoolean keepAlive (MQLONG & keepalive);
Poskytuje kopii hodnoty udržení aktivity. Pokud je úspěšný, vrací TRUE.

MQLONG keepAlive ();
Vrací hodnotu keep alive bez uvedení možných chyb.

ImqString keyRepository ( );
Vrátí úložiště klíčů.

ImqBoolean setKeyRepository (const char * *repository* = 0);
Nastavuje úložiště klíčů. Pokud je úspěšný, vrací TRUE.

ImqBoolean listenerTimer (MQLONG & timer);
Poskytuje kopii hodnoty časovače modulu listener. Pokud je úspěšný, vrací TRUE.

MQLONG listenerTimer ();
Vrátí hodnotu časovače modulu listener bez uvedení možných chyb.

ImqBoolean localEvent(MQLONG & *událost*);
Poskytuje kopii stavu povolení lokální události. Pokud je úspěšný, vrací TRUE.

MQLONG localEvent();
Vrací stav povolení lokální události bez uvedení možných chyb.

ImqBoolean loggerEvent (MQLONG & count);
Poskytuje kopii hodnoty události modulu protokolování. Pokud je úspěšný, vrací TRUE.

MQLONG loggerEvent ();
Vrací hodnotu události modulu protokolování bez uvedení možných chyb.

ImqBoolean luGroupNázev (ImqString & name);.
Poskytuje kopii názvu skupiny LU. Při úspěšném dokončení vrací hodnotu TRUE.

ImqString luGroupNázev ();
Vrátí název skupiny LU bez uvedení možných chyb.

ImqBoolean lu62ARMSuffix (ImqString & suffix);
Poskytuje kopii přípony ARM LU62 . Pokud je úspěšný, vrací TRUE.

ImqString lu62ARMSuffix ( );
Vrací příponu ARM LU62 bez uvedení možných chyb

ImqBoolean luName (ImqString & name);
Poskytuje kopii jména LU. Pokud je úspěšný, vrací TRUE.

ImqString luName ( );
Vrací jméno LU bez označení možných chyb.

ImqBoolean maximumActiveChannels (MQLONG & channels);
Poskytuje kopii hodnoty maximálního počtu aktivních kanálů. Pokud je úspěšný, vrací TRUE.

MQLONG maximumActiveChannels ();
Vrací hodnotu maximálního počtu aktivních kanálů bez uvedení možných chyb.

ImqBoolean maximumCurrentChannels (MQLONG & channels);
Poskytuje kopii hodnoty maximálního počtu aktuálních kanálů. Pokud je úspěšný, vrací TRUE.

MQLONG maximumCurrentChannels ();
Vrací hodnotu maximálního počtu aktuálních kanálů bez uvedení možných chyb.

ImqBoolean maximumHandles(MQLONG & ***číslo*** **);**
Poskytuje kopii maximálního počtu manipulátorů. Pokud je úspěšný, vrací TRUE.

MQLONG maximumHandles();
Vrací maximální počet popisovačů bez uvedení možných chyb.

ImqBoolean maximumLu62Channels (MQLONG & channels);
Poskytuje kopii maximální hodnoty kanálů LU62 . Pokud je úspěšný, vrací TRUE.

MQLONG maximumLu62Channels ();.
Vrací maximální hodnotu kanálu LU62 bez uvedení možných chyb

ImqBoolean maximumMessageLength (MQLONG & ***délka*** **);**
Poskytuje kopii maximální délky zprávy. Pokud je úspěšný, vrací TRUE.

MQLONG maximumMessageLength ();
Vrátí maximální délku zprávy bez uvedení možných chyb.

ImqBoolean maximumPriority(MQLONG & ***priority*** **);**
Poskytuje kopii maximální priority. Pokud je úspěšný, vrací TRUE.

MQLONG maximumPriority();
Vrací kopii maximální priority, bez uvedení možných chyb.

ImqBoolean maximumTcpKanály (MQLONG & channels);
Poskytuje kopii maximální hodnoty kanálů TCP. Pokud je úspěšný, vrací TRUE.

MQLONG maximumTcpChannels ();
Vrací maximální hodnotu kanálů TCP bez uvedení možných chyb.

ImqBoolean maximumUncommittedMessages (MQLONG & ***number*** **);**
Poskytuje kopii maximálního počtu nepotvrzených zpráv. Pokud je úspěšný, vrací TRUE.

MQLONG maximumUncommittedMessages ();
Vrací maximální nepotvrzené zprávy bez uvedení možných chyb.

ImqBoolean mqiAccounting (MQLONG & statint);
Poskytuje kopii účetní hodnoty MQI. Pokud je úspěšný, vrací TRUE.

MQLONG mqiAccounting ();
Vrací hodnotu sledování MQI bez jakýchkoli informací o možných chybách.

ImqBoolean mqiStatistics (MQLONG & statmqi);
Poskytuje kopii hodnoty statistiky MQI. Pokud je úspěšný, vrací TRUE.

MQLONG mqiStatistics ();
Vrací hodnotu statistiky MQI bez uvedení možných chyb.

ImqBoolean outboundPortMax (MQLONG & max);
Poskytuje kopii maximální hodnoty odchozího portu. Pokud je úspěšný, vrací TRUE.

MQLONG outboundPortMax ();
Vrací maximální hodnotu odchozího portu bez uvedení možných chyb.

ImqBoolean outboundPortMin (MQLONG & min);
Poskytuje kopii minimální hodnoty odchozího portu. Pokud je úspěšný, vrací TRUE.

MQLONG outboundPortMin ();
Vrací minimální hodnotu odchozího portu bez uvedení možných chyb.

ImqBinary password () const;
Vrací heslo použité při připojení klienta.

ImqBoolean setPassword (const ImqString & password);
Nastaví heslo použité pro připojení klienta.

ImqBoolean setPassword (const char * = 0 password);
Nastaví heslo použité pro připojení klienta.

ImqBoolean setPassword (const ImqBinary & password);
Nastaví heslo použité pro připojení klienta.

ImqBoolean performanceEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení události výkonu. Pokud je úspěšný, vrací TRUE.

MQLONG performanceEvent();
Vrací stav povolení události výkonu bez uvedení možných chyb.

platforma ImqBoolean (MQLONG & ***platforma*** **);**
Poskytuje kopii platformy. Pokud je úspěšný, vrací TRUE.

MQLONG platform ();
Vrátí platformu bez jakýchkoli informací o možných chybách.

ImqBoolean queueAccounting (MQLONG & acctq);
Poskytuje kopii účetní hodnoty fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueAccounting ();
Vrátí hodnotu účtování fronty bez uvedení možných chyb.

ImqBoolean queueMonitoring (MQLONG & monq);
Poskytuje kopii hodnoty monitorování fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueMonitoring ();
Vrátí hodnotu monitorování fronty bez uvedení možných chyb.

ImqBoolean queueStatistics (MQLONG & statq);
Poskytuje kopii hodnoty statistiky fronty. Pokud je úspěšný, vrací TRUE.

MQLONG queueStatistics ();
Vrátí hodnotu statistiky fronty bez uvedení možných chyb.

ImqBoolean receiveTimeout (MQLONG & timeout);
Poskytuje kopii hodnoty časového limitu pro příjem. Pokud je úspěšný, vrací TRUE.

MQLONG receiveTimeout ();
Vrátí hodnotu časového limitu přijetí bez uvedení možných chyb.

ImqBoolean receiveTimeoutMin (MQLONG & min);
Poskytuje kopii minimální hodnoty časového limitu pro příjem. Pokud je úspěšný, vrací TRUE.

MQLONG receiveTimeoutMin ();
Vrací minimální hodnotu časového limitu přijetí bez uvedení možných chyb.

ImqBoolean receiveTimeoutType (MQLONG & type);
Poskytuje kopii typu časového limitu pro příjem. Pokud je úspěšný, vrací TRUE.

MQLONG receiveTimeoutType ();
Vrátí typ časového limitu přijetí bez uvedení možných chyb.

ImqBoolean remoteEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení vzdálené události. Pokud je úspěšný, vrací TRUE.

MQLONG remoteEvent();
Vrací stav povolení vzdálené události, aniž by došlo k indikaci možných chyb.

ImqBoolean repositoryName(ImqString & ***name*** **);**
Poskytuje kopii názvu úložiště. Pokud je úspěšný, vrací TRUE.

ImqString repositoryName( );
Vrací název úložiště bez uvedení možných chyb.

ImqBoolean repositoryNamelistName (ImqString & ***name*** **);**
Poskytuje kopii názvu seznamu názvů úložiště. Pokud je úspěšný, vrací TRUE.

ImqString repositoryNamelistnázev ();
Vrací kopii názvu seznamu názvů úložiště bez uvedení možných chyb.

ImqBoolean sharedQueueQueueManagerNázev (MQLONG & name);
Poskytuje kopii hodnoty názvu správce front sdílené fronty. Pokud je úspěšný, vrací TRUE.

MQLONG sharedQueueQueueManagerName ();
Vrací hodnotu názvu správce front sdílené fronty bez uvedení možných chyb.

ImqBoolean sslEvent (MQLONG & event);
Poskytuje kopii hodnoty události SSL. Pokud je úspěšný, vrací TRUE.

MQLONG sslEvent ();
Vrací hodnotu události SSL bez uvedení možných chyb.

ImqBoolean sslFips (MQLONG & sslfips);
Poskytuje kopii hodnoty FIPS SSL. Pokud je úspěšný, vrací TRUE.

MQLONG sslFips ();
Vrací hodnotu SSL FIPS bez uvedení možných chyb.

ImqBoolean sslKeyResetCount (MQLONG & count);
Poskytuje kopii hodnoty počtu resetování klíče SSL. Pokud je úspěšný, vrací TRUE.

MQLONG sslKeyResetCount ();
Vrátí hodnotu počtu obnovení klíče SSL bez uvedení možných chyb.

ImqBoolean startStopUdálost (MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení události start-start-stop. Pokud je úspěšný, vrací TRUE.

MQLONG startStopEvent ();
Vrací stav povolení události start-stop bez uvedení možných chyb.

ImqBoolean statisticsInterval (MQLONG & statint);
Poskytuje kopii hodnoty intervalu statistiky. Pokud je úspěšný, vrací TRUE.

MQLONG statisticsInterval ();
Vrací hodnotu intervalu statistiky bez uvedení možných chyb.

ImqBoolean syncPointDostupnost (MQLONG & ***sync*** **);**
Poskytuje kopii hodnoty dostupnosti synchronizačního bodu. Pokud je úspěšný, vrací TRUE.

MQLONG syncPointAvailability ();
Vrací kopii hodnoty dostupnosti synchronizačního bodu bez jakýchkoli informací o možných chybách.

ImqBoolean tcpName (ImqString & name);
Poskytuje kopii názvu systému TCP. Pokud je úspěšný, vrací TRUE.

ImqString tcpName ( );
Vrátí název systému TCP bez uvedení možných chyb.

ImqBoolean tcpStackType (MQLONG & type);
Poskytuje kopii typu sady protokolů TCP. Pokud je úspěšný, vrací TRUE.

MQLONG tcpStacktyp ();
Vrací typ zásobníku TCP bez uvedení možných chyb.

ImqBoolean traceRouteZáznam (MQLONG & routerec);
Poskytuje kopii hodnoty záznamu trasy trasování. Pokud je úspěšný, vrací TRUE.

MQLONG traceRouteRecording ();
Vrátí hodnotu záznamu trasy trasování bez uvedení možných chyb.

ImqBoolean triggerInterval(MQLONG & ***interval*** **);**
Poskytuje kopii intervalu spouštěče. Pokud je úspěšný, vrací TRUE.

MQLONG triggerInterval();
Vrátí interval spouštěče bez uvedení možných chyb.

ImqBinary userId () const;
Vrací ID uživatele použité na 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)

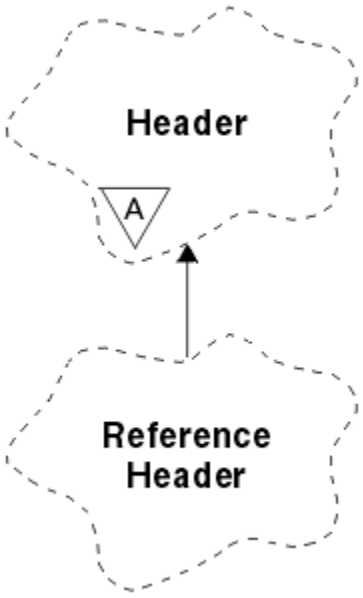
MQHCONN ***ohn ohn***
Manipulátor připojení produktu IBM MQ (smysluplný pouze v případě, že stav připojení je TRUE).

Kódy příčin

- MQRC_ATTRIBUTE_LOCKED
- CHYBA PROSTŘEDÍ MQRC_ENVIRONMENT_ERROR
- PODPOROVÁNO MQRC_FUNCTION_NOT_SUPPORTED
- CHYBA 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 C++ záhlaví ImqReference

Tato třída zapouzdřuje funkce datové struktury MQRMH.



Obrázek 34. Třída záhlaví ImqReference

Tato třída se vztahuje k voláním MQI uvedeným v seznamu “Křížový odkaz záhlaví ImqReference” na stránce 1786.

Atributy objektu

cílové prostředí
Prostředí pro místo určení. 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) o délce MQ_OBJECT_INSTANCE_INSTANCE_LENGTH. Počáteční hodnota je MQOII_NONE.

logická délka
Logická nebo zamýšlená délka dat zprávy, která následuje za tímto záhlavím. Počáteční hodnota je nula.

logický posun
Logické posunutí dat zprávy, které následuje, aby bylo interpretováno v kontextu dat jako celku, v konečném cíli. Počáteční hodnota je nula.

logický posun 2
Rozšíření s vysokým uspořádením k logickému posunu. Počáteční hodnota je nula.

Typ odkazu
Referenční typ. 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.

ImqReferenceZáhlaví (const ImqReferenceHeader & *header*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & *msg*);
Vloží datovou strukturu MQRMH do vyrovnávací paměti zpráv na začátku, dále přesunuje existující data zprávy a nastaví formát *msg* na hodnotu MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader v příručce "Třída C++ ImqHeader" na stránce 1814 .

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Načte datovou strukturu MQRMH z vyrovnávací paměti zpráv.

Aby byla úspěšná, formát ImqMessage musí být MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader v příručce "Třída C++ ImqHeader" na stránce 1814 .

Metody objektů (veřejné)

void operator = (const ImqReferenceHeader & *header*);
Zkopíruje data instance ze záhlaví *header*, přičemž nahradí existující data instance.

ImqString destinationEnvironment () const;
Vrací kopii cílového prostředí.

void setDestinationProstředí (const char * *prostředí* = 0);
Nastavuje cílové prostředí.

ImqString destinationName () const;
Vrací kopii názvu místa určení.

void setDestinationNázev (const char * *název* = 0);
Nastavuje název místa určení.

ImqBinary instanceId () const;
Vrací kopii ID instance.

ImqBoolean setInstanceID (const ImqBinary & *id*);
Nastaví ID instance. Délka dat prvku *token* musí být buď 0, nebo MQ_OBJECT_INSTANCE_INSTANCE_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

void setInstanceId (const MQBYTE24 *id* = 0);
Nastaví ID instance. Parametr *id* může mít hodnotu nula, což je stejné jako určení hodnoty MQOII_NONE. Pokud je hodnota *id* nenulová, musí adresovat MQ_OBJECT_INSTANCE_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je MQOII_NONE, může být nutné, abyste učinili přetypování, abyste zajistili shodu podpisu, například (MQBYTE *) MQOII_NONE.

MQLONG logicalLength () const;
Vrátí logickou délku.

void setLogicalLength (const MQLONG *délka*);
Nastavuje logickou délku.

MQLONG logicalOffset () const;
Vrátí logický posun.

void setLogicalOffset (const MQLONG *posun*);
Nastavuje logický posun.

MQLONG logicalOffset2 () const;
Vrátí logický offset 2.

void setLogicalOffset2 (const MQLONG *posun*);
Nastavuje logický offset 2.

ImqString referenceType () const;
Vrací kopii referenčního typu.

void setReferenceType (const char * *název* = 0);
Nastaví typ odkazu.

ImqString sourceEnvironment () const;
Vrací kopii zdrojového prostředí.

void setSourceProstředí (const char * *prostředí* = 0);
Nastavuje zdrojové prostředí.

ImqString sourceName () const;
Vrací kopii názvu zdroje.

void setSourceNázev (const char * *název* = 0);
Nastaví název zdroje.

Data objektu (chráněná)

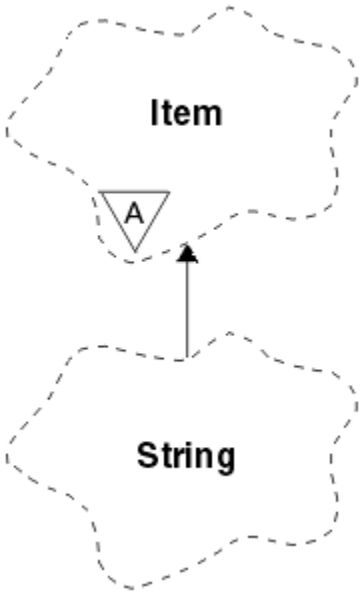
MQRMH *omqrmh*
Datová struktura MQRMH.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR
- CHYBA MQRC_STRUC_LENGTH_ERROR
- CHYBA MQRC_STRUC_ID_ERROR
- MQRC_INSUFFICIENT_DATA
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT
- CHYBA MQRC_ENCODING_ERROR

Třída C++ ImqString

Tato třída poskytuje řetězcovou řetězcovou paměť a manipulaci s řetězci s ukončenými hodnotami null.



Obrázek 35. Třída ImqString

Použijte ImqString místo **char *** ve většině situací, kde parametr volá po **char ***.

- “Atributy objektu” na stránce 1870
- “Konstruktory” na stránce 1870
- “Metody třídy (veřejné)” na stránce 1871
- “Přetížené metody ImqItem” na stránce 1871
- “Metody objektů (veřejné)” na stránce 1871
- “Metody objektů (chráněné)” na stránce 1874
- “Kódy příčin” na stránce 1874

Atributy objektu

znaků
Znaky v **paměti** , které předchází koncové hodnotě null.

délka
Počet bajtů ve **znacích**. Pokud zde není žádná **paměť**, **délka** je nula. Počáteční hodnota je nula.

úložný prostor
Nestálá pole bajtů libovolné velikosti. Koncová hodnota null musí být vždy přítomna v **paměti** za **znaky**, aby bylo možné detekovat konec **znaků** . Metody zajišťují zachování této situace, ale zajišťují, aby při nastavení bajtů v poli přímo existovala koncová hodnota null po úpravě. Na počátku není k dispozici žádný atribut **storage** .

Konstruktory

ImqString( );
Výchozí konstruktor.

ImqString(const ImqString & *řetězec*);
Kopírovací konstruktor.

ImqString(const char *c*);
Znaky zahrnují *c*.

ImqString(const char * *text*);
Znaky **znaky** se zkopírují z *textu*.

ImqString(const void * *buffer*, const size_t *délka*);
Kopíruje *délku* bajtů od *vyrovnávací paměti* a přiřadí je **znakům**. Substituce se provádí pro jakékoli kopie znaků null. Náhradní znak je tečka (.). Žádná zvláštní pozornost nebyla dána žádnému jinému netisknutelnému nebo nezobrazitelným znakům zkopírovaným.

Metody třídy (veřejné)

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, const char *pad* = 0);
Kopíruje do *destination-buffer* bajtů z *source-buffer* do *destination-buffer*. Je-li počet znaků ve *zdrojové vyrovnávací paměti* nedostatečný, zaplní zbývající prostor do pole *destination-buffer* znaky *pad* . *zdroj-vyrovnávací paměť* může být nula. Parametr *destination-buffer* může být nula, je-li *délka* také nula. Všechny chybové kódy se ztratí. Tato metoda vrací TRUE, je-li úspěšná.

static ImqBoolean copy (char * *destination-buffer*, const size_t *length*, const char * *source-buffer*, ImqError & *error-object*, const char *pad* = 0);
Kopíruje do *destination-buffer* bajtů z *source-buffer* do *destination-buffer*. Je-li počet znaků ve *zdrojové vyrovnávací paměti* nedostatečný, zaplní zbývající prostor do pole *destination-buffer* znaky *pad* . *zdroj-vyrovnávací paměť* může být nula. Parametr *destination-buffer* může být nula, je-li *délka* také nula. Všechny chybové kódy jsou nastaveny jako *error-object*. Tato metoda vrací TRUE, je-li úspěšná.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & *msg*);
Zkopíruje **znaky** do vyrovnávací paměti zpráv a nahradí veškerý existující obsah. Nastaví formát *msg* **format** na MQFMT_STRING.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Nastaví **znaky** přenesením zbývajících dat z vyrovnávací paměti zpráv, přičemž nahradí existující **znaky**.

Aby bylo úspěšné, **kódování** objektu *msg* musí být MQENC_NATIVE. Načtěte zprávy s MQGMO_CONVERT do MQENC_NATIVE.

Aby bylo úspěšné, ImqMessage **formátování** musí být MQFMT_STRING.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

Metody objektů (veřejné)

char & operator [] (const size_t *posun*) const;
Odkazuje na znak na offsetu *offset* v **paměti**. Ujistěte se, že příslušný bajt existuje a je adresovatelný.

Operátor ImqString () (const size_t *posun*, const size_t *délka* = 1) const;
Vrátí podřetězec zkopírováním bajtů ze znaků **znaků** , které začínají řetězcem *offset*. Je-li *délka* nula, vrátí zbytek **znaků**. Pokud kombinace hodnoty *offset* a *length* neprodukuje odkaz v rámci **znaků**, vrátí prázdný řetězec ImqString.

void operator = (const ImqString & *řetězec*);
Zkopíruje data instance z řetězce *string*, přičemž nahradí existující data instance.

Operátor ImqString + (const char *c*) const;
Vrátí výsledek připojení *c* k **znakům**.

Operátor ImqString + (const char * *text*) const;
Vrátí výsledek připojení *textu* k **znakům**. To může být také inverzně zobrazený. Příklad:

```
strOne + "string two" ;
"string one" + strTwo ;
```

Poznámka: Ačkoli většina kompilátorů přijímá **strOne + "string two";** Microsoft Visual C++ vyžaduje **strOne + (char *) "string two";**

Operátor ImqString + (const ImqString & *string1*) const;
Vrátí výsledek připojení řetězce *string1* k **znakům**.

Operátor ImqString + (const double *číslo*) const;
Vrátí výsledek připojení *čísla* k **znakům** po převodu na text.

Operátor ImqString + (const long *číslo*) const;
Vrátí výsledek připojení *čísla* k **znakům** po převodu na text.

neobsazený operátor + = (const char *c*);
Připojí *c* k **znakům**.

void operator + = (const char * *text*);
Připojí *text* k **znakům**.

neobsazený operátor + = (const ImqString & *řetězec*);
Připojí *řetězec* k **znakům**.

neobsazený operátor + = (const double *číslo*);
Připojí *číslo* k **znakům** po převodu na text.

void operator + = (const long *číslo*);
Připojí *číslo* k **znakům** po převodu na text.

operátor char * () const;
Vrátí adresu prvního bajtu v **paměti**. Tato hodnota může být nulová a je nestálá. Tuto metodu používejte pouze pro čtení.

Operátor ImqBoolean < (const ImqString & *string*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Výsledek je TRUE, je-li menší než a FALSE, je-li větší než nebo rovno.

Operátor ImqBoolean > (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Výsledek je TRUE, je-li větší než a FALSE, je-li menší než nebo rovno.

Operátor ImqBoolean < = (const ImqString & *string*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Výsledek je TRUE, je-li menší než nebo rovno a FALSE, je-li větší než.

Operátor ImqBoolean > = (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Výsledek je TRUE, je-li větší než nebo rovno a FALSE, je-li menší než.

Operátor ImqBoolean == (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Vrací TRUE nebo FALSE.

ImqBoolean operator! = (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězce* pomocí metody **compare** . Vrací TRUE nebo FALSE.

short compare (const ImqString & *řetězec*) const;
Porovná **znaky** s těmi, které jsou *řetězec*. Výsledek je nula, pokud jsou **znaky** stejné, záporné, pokud jsou menší než a kladné, jsou-li větší než. Porovnání rozlišuje velikost písmen. Null ImqString je považován za méně než null ImqStringbez hodnoty null.

ImqBoolean copyOut(char * *buffer*, const size_t *délka*, const char *pad* = 0);
Zkopíruje do *length* bajtů z **znaků** do *vyrovnávací paměti*. Pokud je počet znaků **znaků** nedostatečný, zaplní zbývající prostor ve *vyrovnávací paměti* znaky *pad* . *vyrovnávací paměť* může být nula, je-li *délka* také nula. Pokud je úspěšný, vrací TRUE.

size_t copyOut(dlouhé & *číslo*) const;
Nastaví *číslo* z **znaků** po převodu z textu a vrátí počet znaků zahrnutých do převodu. Je-li tato hodnota nula, nebyla provedena žádná konverze a *číslo* není nastaveno. Skonvertibilní 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 *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*);

Nastavuje *číslo* jako pro metodu **copy** , ale také odebere z **znaků** počet bajtů indikovaných návratovou hodnotou. Například řetězec zobrazený v následujícím příkladu lze snížit na tři čísla pomocí volby **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 ze **znaků** znaků *strToken* a také všechny znaky *c* , které předchází znakům *token* . Pokud *c* není prázdná, odstraní znaky *c* , které mají přímý úspěch *tokenu* znaků. Vrátí počet odstraněných znaků. Například řetězec zobrazený v následujícím příkladu lze vyjmout do tří tokenů pomocí volby **cutOut** (*token*). Třikrát:

```
strText = " Program Version 1.1 "

while ( strText.cutOut( token ) );

// token becomes "Program", then "Version",
// then "1.1" leaving strText == " "
```

Následující příklad ukazuje, jak analyzovat název cesty systému 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 & *string*);

Hledání přesné shody pro *řetězec* kdekoli v rámci **znaků**. Pokud není nalezena žádná shoda, vrátí hodnotu FALSE. Jinak vrací hodnotu TRUE. Má-li parametr *řetězec* hodnotu null, vrací hodnotu TRUE.

ImqBoolean find (const ImqString & *string*, size_t & *offset*);

Hledá přesnou shodu pro řetězec *string* někde uvnitř **znaků** od offsetu *offset* od sebe. Má-li parametr *řetězec* hodnotu null, vrací hodnotu TRUE bez aktualizace parametru *offset*. Pokud není nalezena žádná shoda, vrací FALSE (hodnota *offsetu* mohla být zvýšena). Je-li nalezena shoda, vrátí hodnotu TRUE a aktualizuje *posun* na posun řetězce *string* v rámci **znaků**.

délka () velikost_t const;

Vrací **délku**.

ImqBoolean pasteIn(const double *číslo*, const char * *format* = "%f");

Připojí *číslo* k **znakům** po převodu na text. Pokud je úspěšný, vrací TRUE.

Specifikace *format* se používá k formátování převodu s pohyblivou řádovou čárkou. Je-li určen, musí být vhodný pro použití s čísly **printf** a čísly s pohyblivou řádovou čárkou, například **%.3f**.

ImqBoolean pasteIn(dlouhé číslo *number*);

Připojí *číslo* k **znakům** po převodu na text. Pokud je úspěšný, vrací TRUE.

ImqBoolean pasteIn(const void * *buffer*, const size_t *délka*);
Připojí *délku* bajtů od *vyrovnávací paměti* k **znakům**a přidá konečnou koncovou hodnotu null. Nahradí se všechny prázdné znaky null. Náhradní znak je tečka (.). Žádná zvláštní pozornost nebyla dána žádnému jinému netisknutelnému nebo nezobrazitelným znakům zkopírovaným. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean set (const char * *buffer*, const size_t *délka*);
Nastaví **znaky** z pole znaků pevné délky, které může obsahovat hodnotu null. V případě potřeby přidá k znakům z pole pevné délky hodnotu null. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setStorage(const size_t *délka*);
Přiděluje (nebo znovu alokuje) **úložiště**. Zachovává všechny původní **znaky**, včetně všech koncových hodnot null, pokud je pro ně stále ještě prostor, ale neinicializuje žádné další úložiště.

Tato metoda vrací TRUE, je-li úspěšná.

velikost_úložiště () const;
Vrací počet bajtů v **úložišti**.

size_t stripLeading(const char *c* = '');
Odřízne úvodní znaky *c* ze **znaků** a vrátí číslo odebrané.

size_t stripTrailing(const char *c* = '');
Odřízne koncové znaky *c* z **znaků** a vrátí číslo odebrané.

ImqString upperCase() const;
Vrací kopii souboru **znaků**na velká písmena.

Metody objektů (chráněné)

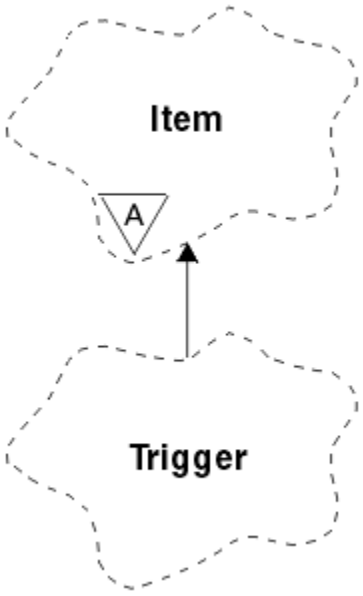
ImqBoolean assign (const ImqString & *string*);
Ekvivalent ekvivalentní metodě **operator =** , ale nevirtuální. Pokud je úspěšný, vrací TRUE.

Kódy příčin

- MQRC_DATA_OŘÍZNUTÁ
- MQRC_NULL_POINTER
- MQRC_STORAGE_NOT_AVAILABLE
- CHYBA MQRC_BUFFER_ERROR
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT

Třída C++ ImqTrigger

Tato třída zapouzdřuje datovou strukturu MQTM (trigger message).



Obrázek 36. Třída ImqTrigger

Objekty této třídy jsou obvykle používány programem monitoru spouštěčů. Úkolem programu pro monitorování spouštěčů je čekat na tyto konkrétní zprávy a pracovat na nich, aby bylo zajištěno, že ostatní aplikace produktu IBM MQ budou spuštěny, když na ně čekají zprávy.

Příklad použití najdete v ukázkovém programu IMQSTRG.

- "Atributy objektu" na stránce 1875
- "Konstruktory" na stránce 1876
- "Přetížené metody ImqItem" na stránce 1876
- "Metody objektů (veřejné)" na stránce 1876
- "Data objektu (chráněná)" na stránce 1877
- "Kódy příčin" na stránce 1877

Atributy objektu

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
- MQAT_CICS
- MQAT_DOS
- MQAT_IMS
- MQAT_MVS
- MQAT_NOTES_AGENT
- MQAT_OS2
- MQAT_OS390
- MQAT_OS400
- MQAT_UNIX
- MQAT_WINDOWS
- POČ 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*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut (ImqMessage & *msg*);
Zapisuje datovou strukturu MQTM do vyrovnávací paměti zpráv a nahradí veškerý stávající obsah. Nastaví formát *msg* na MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem v příručce “Třída C++ ImqItem” na stránce 1819 .

virtual ImqBoolean pasteIn (ImqMessage & *msg*);
Načte datovou strukturu MQTM z vyrovnávací paměti zpráv.

Aby byl úspěšný, formát ImqMessage musí být MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem v příručce “Třída C++ ImqItem” na stránce 1819 .

Metody objektů (veřejné)

void operator = (const ImqTrigger & *trigger*);
Kopíruje data instance z *triggeru*, která nahradí existující data instance.

ImqString applicationId () const;
Vrací kopii ID aplikace.

void setApplicationId (const char * *id*);
Nastavuje ID aplikace.

MQLONG applicationType () const;
Vrátí typ aplikace.

void setApplicationType (const MQLONG *typ*);
Nastaví typ aplikace.

ImqBoolean copyOut (MQTMC2 * *ptmc2*);
Zapouzdřuje datovou strukturu MQTM, která byla přijata v inicializačních frontách. Filly v ekvivalentní datové struktuře MQTMC2 poskytnuté volajícím a nastavuje pole QMgrName (které není přítomno ve struktuře dat MQTM) na všechny prázdné znaky. Datová struktura MQTMC2 se tradičně používá jako parametr pro aplikace spouštěné monitorem spouštěčů. Tato metoda vrací TRUE, je-li úspěšná.

ImqString environmentData () const;
Vrací kopii dat prostředí.

void setEnvironmentData (const char * *data*);
Nastavuje data prostředí.

ImqString processName () const;
Vrací kopii názvu procesu.

void setProcessName (const char * *název*);
Nastaví název procesu, doplněný mezerami na 48 znaků.

ImqString queueName () const;
Vrací kopii názvu fronty.

void setQueueNázev (const char * *název*);
Nastaví název fronty, výplň s mezerami na 48 znaků.

ImqString triggerData () const;
Vrací kopii dat spouštěče.

void setTriggerData (const char * *data*);
Nastavuje data spouštěče.

ImqString userData () const;
Vrací kopii uživatelských dat.

void setUserData (const char * *data*);
Nastavuje uživatelská data.

Data objektu (chráněná)

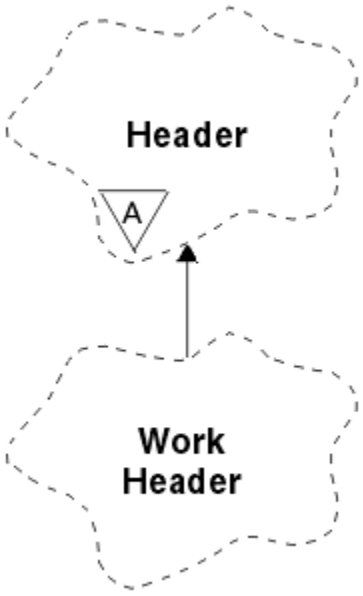
MQTM *omqtm*
Datová struktura MQTM.

Kódy příčin

- MQRC_NULL_POINTER
- FORMÁT NEKONZISTENCE MQRC_INCONSISTENT_FORMAT
- CHYBA MQRC_ENCODING_ERROR
- CHYBA MQRC_STRUC_ID_ERROR

Třída C++ záhlaví ImqWork

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 produktem z/OS Workload Manager.

- "Atributy objektu" na stránce 1878
- "Konstruktory" na stránce 1878
- "Přetížené metody ImqItem" na stránce 1878
- "Metody objektů (veřejné)" na stránce 1879
- "Data objektu (chráněná)" na stránce 1879
- "Kódy příčin" na stránce 1879

Atributy objektu

token zprávy
Token zprávy pro produkt z/OS Workload Manager s délkou MQ_MSG_TOKEN_LENGTH. Počáteční hodnota je MQMTOK_NONE.

Název služby
32znakový název procesu. Název je na začátku prázdný.

servisní krok
Osmiznakový název kroku v rámci procesu. Název je na začátku prázdný.

Konstruktory

ImqWorkHlavička ();
Výchozí konstruktor.

Záhlaví ImqWork(const ImqWorkHeader & *header*);
Kopírovací konstruktor.

Přetížené metody ImqItem

virtual ImqBoolean copyOut(ImqMessage & *msg*);
Vloží datovou strukturu MQWIH do začátku vyrovnávací paměti zpráv, přesune stávající data zprávy dále a nastaví *formát zprávy* **format** na hodnotu MQFMT_WORK_INFO_HEADER.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

virtual ImqBoolean pasteIn(ImqMessage & *msg*);
Načte datovou strukturu MQWIH z vyrovnávací paměti zpráv.

Aby bylo úspěšné, zakódování objektu *msg* musí být MQENC_NATIVE. Načtěte zprávy s MQGMO_CONVERT do MQENC_NATIVE.

Formát ImqMessage musí být MQFMT_WORK_INFO_HEADER.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqWorkHeader & *header*);
Zkopíruje data instance ze záhlaví *header*, přičemž nahradí existující data instance.

ImqBinary messageToken () const;
Vrací **token zprávy**.

ImqBoolean setMessageToken (const ImqBinary & *token*);
Nastaví **token zprávy**. Délka dat *token* musí být buď nula, nebo MQ_MSG_TOKEN_LENGTH. Pokud je úspěšný, vrací TRUE.

void setMessageToken (const MQBYTE16 *token* = 0);
Nastaví **token zprávy**. *token* může být nula, což je stejné jako uvedení hodnoty MQMTOK_NONE. Je-li token *token* nenulový, musí adresovat MQ_MSG_TOKEN_LENGTH bajtů binárních dat.

Při použití předdefinovaných hodnot, jako je MQMTOK_NONE, může být třeba vytvořit přetypování, abyste zajistili shodu podpisu; například, (MQBYTE *) MQMTOK_NONE.

ImqString serviceName () const;
Vrací **název služby**, včetně koncových mezer.

void setServiceNázev (const char * *název*);
Nastaví **název služby**.

ImqString serviceStep () const;
Vrací **krok služby**, včetně koncových mezer.

void setServiceKrok (const char * *krok*);
Nastaví **krok služby**.

Data objektu (chráněná)

MQWIH omqwih
Datová struktura MQWIH.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR

Vlastnosti objektů IBM MQ classes for JMS

Všechny objekty v produktu IBM MQ classes for JMS mají vlastnosti. Různé vlastnosti platí pro 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 funkce pro nastavení a dotazování na vlastnosti objektů pomocí nástroje pro administraci produktu IBM MQ JMS , Průzkumníka IBM MQ nebo v aplikaci. Mnohé z vlastností jsou relevantní pouze pro specifickou podmnožinu typů objektů.

Informace o tom, jak používat administrativní nástroj produktu IBM MQ JMS , najdete v tématu Konfigurace objektů produktu JMS pomocí nástroje pro administraci.

Produkt Tabulka 869 na stránce 1880 podává stručný popis každé vlastnosti a uvádí pro každou vlastnost, pro které typy objektů se používá. Typy objektů jsou identifikovány pomocí klíčových slov, viz téma

Konfigurace objektů produktu JMS pomocí nástroje pro administraci , kde najdete vysvětlení těchto objektů.

Čísla odkazují na poznámky na konci tabulky. Další informace najdete v tématu “Závislosti mezi vlastnostmi objektů produktu IBM MQ classes for JMS” na stránce 1883.

Vlastnost se skládá z dvojice název-hodnota ve formátu:

```
PROPERTY_NAME(property_value)
```

Témata v tomto oddílu, pro každou vlastnost, název vlastnosti a stručný popis a uvádí platné hodnoty vlastností použité v nástroji pro administraci. a metoda set, která se používá k nastavení hodnoty vlastnosti v aplikaci. Témata také zobrazují platné hodnoty vlastností pro každou vlastnost a mapování mezi hodnotami symbolických vlastností používanými v nástroji a jejich programovatelnými ekvivalenty.

Názvy vlastností nejsou citlivé na velikost písmen a jsou omezeny na sadu rozpoznaných názvů zobrazených v těchto tématech.

Tabulka 869. Názvy vlastností a použitelné typy objektů

Vlastnost	**Krátký formát**	**Typ objektu**							
		CF	**QCF**	**TCF**	**Q**	**T**	**XACF**	**XAQCF**	**XATCF**
“APPLICATIONNAME” na stránce 1885	APPNAME	Y	Y	Y			Y	Y	Y
“VÝJIMKA ASYNCEXCEPTION” na stránce 1885	AEX	Y	Y	Y			Y	Y	Y
“BROKERCCDURSUBQ” na stránce 1886 [1]	CCDSUB					Y			
“BROKERCCSUBQ” na stránce 1887 [1]	CCSUB	Y		Y			Y		Y
“BROKERCONQ” na stránce 1887 [1]	BCON	Y		Y			Y		Y
“BROKERDURSUBQ” na stránce 1888 [1]	BDSUB					Y			
“BROKERPUBQ” na stránce 1888 [1]	BPUB	Y		Y		Y	Y		Y
“BROKERPUBQMGR” na stránce 1889 [1]	BPQM					Y			
“BROKERQMGR” na stránce 1889 [1]	BQM	Y		Y			Y		Y
“BROKERSUBQ” na stránce 1889 [1]	BSUB	Y		Y			Y		Y
“BROKERVER” na stránce 1890 [1]	BVER	A [2]		A [2]		Y	Y		Y
“CCDTURL” na stránce 1891 [3]	CCDT	Y	Y	Y			Y	Y	Y
“CCSID” na stránce 1891	CCS	Y	Y	Y	Y	Y	Y	Y	Y
“CHANNEL” na stránce 1892 [3]	CHAN	Y	Y	Y			Y	Y	Y
“CLEANUP” na stránce 1892 [1]	CL	Y		Y			Y		Y
“CLEANUPINT” na stránce 1893 [1]	CLINT	Y		Y			Y		Y
“ConnectionNameList” na stránce 1893	CNLIST	Y	Y	Y					
“CLIENTRECONNECTOPTIONS” na stránce 1893	CROPT	Y	Y	Y					

Tabulka 869. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
“CLIENTRECONNECTTIMEOUT” na stránce 1894	CRT	Y	Y	Y					
“CLIENTID” na stránce 1895	CID	A [2]	Y	A [2]			Y	Y	Y
“CLONESUPP” na stránce 1895	CLS	Y		Y			Y		Y
“COMPHDR” na stránce 1896	HC	Y		Y			Y		Y
“COMPMSG” na stránce 1896	MC	Y	Y	Y			Y	Y	Y
“CONNOPT” na stránce 1897	CNOPT	Y	Y	Y			Y	Y	Y
“CONNTAG” na stránce 1898	CNTAG	Y	Y	Y			Y	Y	Y
“DESCRIPTION” na stránce 1898	DESC	A [2]	Y	A [2]	Y	Y	Y	Y	Y
“DIRECTAUTH” na stránce 1898	DAUTH	A [2]		A [2]					
“ENCODING” na stránce 1899	ENC				Y	Y			
“EXPIRY” na stránce 1900	EXP				Y	Y			
“FAILIFQUIESCE” na stránce 1900	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
“HOSTNAME” na stránce 1901	HOST	A [2]	Y	A [2]			Y	Y	Y
“LOCALADDRESS” na stránce 1902	LA	A [2]	Y	A [2]			Y	Y	Y
“STYL MAPNAMESTYLE” na stránce 1902	MNST	Y	Y	Y			Y	Y	Y
“MAXBUFFSIZE” na stránce 1903	MBSZ	A [2]		A [2]					
“MDREAD” na stránce 1903	MDR				Y	Y			
“MDWRITE” na stránce 1904	MDW				Y	Y			
“MDMSGCTX” na stránce 1904	MDCTX				Y	Y			
“MSGBATCHSZ” na stránce 1905 [1]	MBS	Y	Y	Y			Y	Y	Y
“MSGBODY” na stránce 1905	MBODY				Y	Y			
“MSGRETENTION” na stránce 1906	MRET	Y	Y				Y	Y	
“MSGSELECTION” na stránce 1906 [1]	MSEL	Y		Y			Y		Y
“MULTICAST” na stránce 1907	MCAST	A [2]		A [2]		Y			
“OPTIMISTICPUBLICATION” na stránce 1908 [1]	OPTPUB	Y		Y					
“OUTCOMENOTIFICATION” na stránce 1908 [1]	NOTIFY	Y		Y					
“PERSISTENCE” na stránce 1909	PER				Y	Y			
“POLLINGINT” na stránce 1909 [1]	PINT	Y	Y	Y			Y	Y	Y
“PORT” na stránce 1910	PORT	A [2]	Y	A [2]			Y	Y	Y
“PRIORITY” na stránce 1910	PRI				Y	Y			

Tabulka 869. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
"PROCESSDURATION" na stránce 1911 [1]	PROCDUR	Y		Y					
"PROVIDERVERSION" na stránce 1911	PVER	Y	Y	Y			Y	Y	Y
"PROXYHOSTNAME" na stránce 1914	PHOST	A [2]		A [2]					
"PROXYPORT" na stránce 1914	PPORT	A [2]		A [2]					
"PUBACKINT" na stránce 1914 [1]	PAI	Y		Y			Y		Y
"PUTASYNCALLOWED" na stránce 1915	PAALDOVA				Y	Y			
"QMANAGER" na stránce 1915	QMGR	Y	Y	Y	Y		Y	Y	Y
"QUEUE" na stránce 1916	QU				Y				
"READAHEADALLOWED" na stránce 1916	RAALDOVÁ				Y	Y			
"READAHEADCLOSEPOLICY" na stránce 1917	RACP				Y	Y			
"RECEIVECCSID" na stránce 1917	RCCS				Y	Y			
"RECEIVECONVERSION" na stránce 1918	RCNV				Y	Y			
"RECEIVEISOLATION" na stránce 1918 [1]	RCVISOL	Y		Y					
"RECEXIT" na stránce 1919	RCX	Y	Y	Y			Y	Y	Y
"RECEXITINIT" na stránce 1919	RCXI	Y	Y	Y			Y	Y	Y
"REPLYTOSTYLE" na stránce 1920	RTOST				Y	Y			
"RESCANINT" na stránce 1920 [1]	RINT	Y	Y				Y	Y	
"SECEXIT" na stránce 1921	SCX	Y	Y	Y			Y	Y	Y
"SECEXITINIT" na stránce 1921	SCXI	Y	Y	Y			Y	Y	Y
"SENDCHECKCOUNT" na stránce 1922	SCC	Y	Y	Y			Y	Y	Y
"SENDEXIT" na stránce 1922	SDX	Y	Y	Y			Y	Y	Y
"SENDEXITINIT" na stránce 1923	SDXI	Y	Y	Y			Y	Y	Y
"SHARECONVALLOWED" na stránce 1923	SCALD	Y	Y	Y			Y	Y	Y
"SPARSESUBS" na stránce 1924 [1]	SSUBS	Y		Y					
"SSLCIPHERSUITE" na stránce 1925	SCPHS	Y	Y	Y			Y	Y	Y
"SSLCRL" na stránce 1925	SCRL	Y	Y	Y			Y	Y	Y
"SSLFIPSREQUIRED" na stránce 1925	SFIPS	Y	Y	Y			Y	Y	Y

Tabulka 869. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
"SSLPEERNAME" na stránce 1926	SPEER	Y	Y	Y			Y	Y	Y
"SSLRESETCOUNT" na stránce 1926	SRC	Y	Y	Y			Y	Y	Y
"STATREFRESHINT" na stránce 1927 [1]	SRI	Y		Y			Y		Y
"SUBSTORE" na stránce 1927 [1]	SS	Y		Y			Y		Y
"SYNCPOINTALLGETS" na stránce 1928	SPAG	Y	Y	Y			Y	Y	Y
"TARGCLIENT" na stránce 1928	TC				Y	Y			
"TARGCLIENTMATCHING" na stránce 1929	TCM	Y	Y				Y	Y	
"TEMPMODEL" na stránce 1929	TM	Y	Y				Y	Y	
"TEMPQPREFIX" na stránce 1930	TQP	Y	Y				Y	Y	
"TEMPTOPICPREFIX" na stránce 1930	TTP	Y		Y			Y		Y
"TOPIC" na stránce 1931	TOP					Y			
"TRANSPORT" na stránce 1931	TRAN	A [2]	Y	A [2]			Y	Y	Y
"WILDCARDFORMAT" na stránce 1932	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 produktu IBM WebSphere MQ 7.0 , pokud není vlastnost PROVIDERVERSION továrny připojení nastavena na číslo verze nižší 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 ConnectionFactory nebo TopicConnection.
3. Vlastnosti CCDTURL a CHANNEL objektu nesmí být obě nastaveny současně.

Závislosti mezi vlastnostmi objektů produktu 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
- Ukončovací inicializační řetězce

Vlastnosti klienta

Pro připojení ke správci front jsou následující vlastnosti relevantní pouze v případě, že funkce TRANSPORT má hodnotu CLIENT:

- HOSTNAME
- PORT

- CHANNEL
- LOCALADDRESS
- CCDTURL
- CCSID
- COMPHDR
- COMPMSG
- RECEXIT
- RECEXITINIT
- SECEXIT
- SECEXITINIT
- SENDEXIT
- SENDEXITINIT
- SHARECONVALLOWED
- SSLCIPHERSUITE
- SSLCRL
- SSLFIPSREQUIRED
- SSLPEERNAME
- SSLRESETCOUNT
- APPLICATIONNAME

Hodnoty pro tyto vlastnosti nelze nastavit pomocí nástroje pro administraci, pokud má funkce TRANSPORT hodnotu BIND.

Má-li funkce TRANSPORT hodnotu CLIENT, je výchozí hodnota vlastnosti BROKERVER nastavena na hodnotu V1 a výchozí hodnota vlastnosti PORT je 1414. Pokud nastavíte hodnotu BROKERVER nebo PORT explicitně, nezmění se vaše volby na pozdější změnu na hodnotu TRANSPORT.

Vlastnosti pro připojení v reálném čase ke zprostředkovateli

Pokud má funkce TRANSPORT hodnotu DIRECT nebo DIRECTHTTP, mají význam pouze následující vlastnosti:

- BROKERVER
- CLIENTID
- DESCRIPTION
- DIRECTAUTH
- HOSTNAME
- LOCALADDRESS
- MAXBUFFSIZE
- MULTICAST (podporováno pouze pro DIRECT)
- PORT
- PROXYHOSTNAME (podporováno pouze pro DIRECT)
- PROXYPT (podporováno pouze pro DIRECT)

Má-li funkce TRANSPORT hodnotu DIRECT nebo DIRECTTTP, je výchozí hodnota vlastnosti BROKERVER nastavena na hodnotu V2a výchozí hodnota vlastnosti PORT je 1506. Pokud nastavíte hodnotu BROKERVER nebo PORT explicitně, nezmění se vaše volby na pozdější změnu na hodnotu TRANSPORT.

Ukončovací inicializační řetězce

Nenastavujte žádný z inicializačních řetězců ukončení bez zadání odpovídajícího jména ukončení. Vlastnosti inicializace uživatelské procedury jsou:

- RECEXITINIT
- SECEXITINIT
- SENDEXITINIT

Například uvedení `RECEXITINIT(myString)` bez uvedení `RECEXIT(some.exit.classname)` způsobí chybu.

Související odkazy

“TRANSPORT” na stránce 1931
Povaha připojení ke správci front nebo zprostředkovateli.

APPLICATIONNAME

Aplikace může nastavit název, který identifikuje její připojení ke správci front. Tento název aplikace se zobrazí příkazem **DISPLAY CONN MQSC/PCF** (kde pole se nazývá **APPLTAG**). nebo v zobrazení Průzkumníka IBM MQ **Připojení aplikace** (kde se pole nazývá **App name**).

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : APPLICATIONNAME

Krátký název nástroje pro administraci produktu JMS : APPNAME

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setAppName ()
- MQConnectionFactory.getAppName ()

Hodnoty

Jakýkoli platný řetězec, který není delší než 28 znaků. Delší názvy jsou upraveny tak, aby se vešly tak, že v případě potřeby odeberou úvodní názvy balíků. Je-li například třída vyvolání `com.example.MainApp`, použije se úplný název, ale pokud je třída vyvolání `com.example.dictionaryAndThesaurus.multilingual.mainApp`, použije se název `multilingual.mainApp` , protože se jedná o nejdelší kombinaci názvu třídy a názvu balíku nejvíce vpravo, který se vejde do dostupné délky.

Je-li název třídy delší než 28 znaků, je oříznut na vhodný. Například `com.example.mainApplicationForSecondTestCase` se stane `mainApplicationForSecondTest`.

V z/OSse APPNAME v:

- Režim vazeb je ignorován, pokud je nastaven a, je-li nastaven, může být nastaven pouze na mezery.
- Režim klienta lze nastavit a použít.

VÝJIMKA ASYNCEXCEPTION

Tato vlastnost určuje, zda produkt IBM MQ classes for JMS informuje modul ExceptionListener pouze v případě, že je spojení přerušeno, nebo když dojde k asynchronnímu výskytu jakékoli výjimky k volání rozhraní API produktu JMS asynchronně. Toto platí pro všechna připojení vytvořená z této ConnectionFactory , která má registrován ExceptionListener .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : ASYNCEXCEPTION

Krátký název nástroje pro administraci produktu JMS : AEX

Programový přístup

Metody Setter/Getter

- MQConnectionFactory.setAsyncExceptions ()
- MQConnectionFactory.getAsyncExceptions ()

Hodnoty

ASYNC_EXCEPTIONS_ALL

Jakákoli výjimka byla zjištěna asynchronně, mimo rozsah volání synchronního volání rozhraní API a všechny přerušené výjimky připojení jsou odeslány do ExceptionListener.

Tabulka 870. Všechny asynchronní výjimky: prostředí a související konstantní názvy

Prostředí	Hodnota
JMS nástroj administrace	ALL
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1
IBM MQ Explorer	Vše

ASYNC_EXCEPTIONS_CONNECTIONBROKEN

Do modulu ExceptionListenerse odesílají pouze výjimky označující přerušené připojení. Jakékoli další výjimky, které se vyskytly během asynchronního zpracování, se do modulu ExceptionListenerneohlašují, a proto není aplikace informována o těchto výjimkách. Toto je výchozí hodnota z IBM MQ 8.0.0 Fix Pack 2. Viz termín JMS: Změny modulu listener pro výjimky v produktu IBM MQ 8.0.

Tabulka 871. Výjimky označující přerušené připojení: prostředí a související názvy konstant

Prostředí	Hodnota
JMS nástroj administrace	SPOJENÍ PŘERUŠENO
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
IBM MQ Explorer	Připojení přerušeno

Je definována následující dodatečná konstanta:

- V produktu 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 IBM MQ classes for JMS

BROKERCCDURSUBQ

Název fronty, z níž jsou načítány zprávy trvalého odběru pro ConnectionConsumer.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci produktu JMS : BROKERCCDURSUBQ

Krátký název nástroje pro administraci produktu 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 ConnectionConsumer.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERCCSUBQ

Krátký název nástroje pro administraci produktu 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, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERCONQ

Krátký název nástroje pro administraci produktu JMS : BCON

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

Hodnoty

SYSTEM.BROKER.CONTROL.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERDURSUBQ

Je-li produkt IBM MQ classes for JMS používán v režimu migrace poskytovatele systému zpráv produktu IBM MQ , určuje tato vlastnost název fronty, z níž jsou načítány zprávy trvalého odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci produktu JMS : BROKERDURSUBQ

Krátký název nástroje pro administraci produktu JMS : BDSUB

Programový přístup

Metody Setter/getter

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

Hodnoty

SYSTEM.JMS.D.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Spouští se SYSTEM.JMS.D

Související úlohy
Konfigurace vlastnosti produktu JMS **PROVIDERVERSION**

BROKERPUBQ

Název fronty, do které jsou odesílány publikované zprávy (fronta proudu).

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERPUBQ

Krátký název nástroje pro administraci produktu 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 níž jsou odesílány zprávy publikované v rámci daného tématu.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci produktu JMS : BROKERPUBQMGR

Krátký název nástroje pro administraci produktu 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, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERQMGR

Krátký název nástroje pro administraci produktu JMS : BQM

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERSUBQ

Je-li produkt IBM MQ classes for JMS používán v režimu migrace poskytovatele systému zpráv produktu IBM MQ , určuje tato vlastnost název fronty, z níž jsou načítány zprávy z trvalého odběru.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERSUBQ

Krátký název nástroje pro administraci produktu JMS : BSUB

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

Hodnoty

SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Spouští se SYSTEM.JMS.ND

Související úlohy
Konfigurace vlastnosti produktu JMS **PROVIDERVERSION**

BROKERVER

Verze používaného zprostředkovatele.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : BROKERVER

Krátký název nástroje pro administraci produktu JMS : BVER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerverze ()
- MQConnectionFactory.getBrokerverze ()

Hodnoty

V1
To use an IBM MQ Publish/Subscribe broker, or to use a broker of IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker, or WebSphere Business Integration Message Broker in compatibility mode. Jedná se o výchozí hodnotu, pokud je funkce TRANSPORT nastavena na BIND nebo CLIENT.

V2
Chcete-li použít zprostředkovatele produktů IBM MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker nebo WebSphere Business Integration Message Broker v nativním režimu. Jedná se o výchozí hodnotu, pokud je funkce TRANSPORT nastavena na DIRECT nebo DIRECTHTTP.

nespecifikováno
Po migraci zprostředkovatele z V6 na V7nastavte tuto vlastnost tak, aby se záhlaví RFH2 nadále nepoužívala. Po migraci již tato vlastnost není relevantní.

CCDTURL

Adresa URL (Uniform Resource Locator), která identifikuje název a umístění souboru obsahujícího tabulku definic kanálů klienta a určuje, jak lze k souboru přistupovat.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : CCDTURL

Krátký název nástroje pro administraci produktu JMS : CCDT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

Hodnoty

null
Toto je výchozí hodnota.

Adresa URL (Uniform Resource Locator)

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 produktu JMS : CHAN

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

Hodnoty

SYSTEM.DEF.SVRCONN
Toto je výchozí hodnota.

Libovolný platný řetězec

CLEANUP

Úroveň úklidu pro úložiště odběrů BROKER či MIGRATE.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : CLEANUP

Krátký název nástroje pro administraci produktu JMS : CL

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setCleanupLevel ()
- MQConnectionFactory.getCleanupLevel ()

Hodnoty

Bezpečný
Použijte bezpečné vyčištění. Toto je výchozí hodnota.

ASPROP
Použijte bezpečné, silné nebo žádné vyčištění na základě vlastnosti nastavené na příkazovém řádku Java .

ŽÁDNÉ
Nepoužít žádné vyčištění.

velká
Použijte silné vyčištění.

CLEANUPINT

Interval, v milisekundách, mezi zpracováním obslužného programu čištění publikování/odběru na pozadí.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : CLEANUPINT

Krátký název nástroje pro administraci produktu 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 se zkouší v pořadí, jednou za každý pokus o opakování pokusu o opětovné připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : CONNECTIONNAMELIST

Krátký název nástroje pro administraci produktu JMS : CNLIST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setconnectionNameList ()
- MQConnectionFactory.getconnectionNameList ()

Hodnoty

Čárkou oddělený seznam HOSTNAME (PORT). Parametr HOSTNAME může být buď název DNS, nebo adresa IP.

Výchozí hodnota parametru PORT je 1414.

CLIENTRECONNECTOPTIONS

Volby regulace opětovného připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : CLIENTRECONECOPTIONS

Krátký název nástroje pro administraci produktu 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 připojen.

Byla vrácena chyba s kódem příčiny `MQRC_RECONNECT_QMID_MISMATCH` , pokud se správce front, ke kterému se aplikace pokouší připojit, jak je uvedeno v seznamu názvů připojení, liší od QMID ke správci front, k němuž se původně připojil.

Tuto hodnotu použijte v případě, že lze aplikaci znovu připojit, ale existuje afinita mezi aplikací produktu IBM MQ classes for JMS a správcem front, k němuž se nejprve navázají připojení.

Tuto hodnotu zvolte, chcete-li, aby se aplikace automaticky znovu připojila k instanci v pohotovostním režimu pro vysoce dostupného správce front.

Chcete-li tuto hodnotu použít programově, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT_Q_MGR.

ANY

Aplikace se může znovu připojit ke kterému správci front uvedenému v seznamu názvů připojení.

Volbu opětovného připojení použijte pouze v případě, že neexistuje žádná afinita mezi třídami IBM MQ pro aplikaci JMS a správcem front, se kterým na počátku navázaly spojení.

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.

VZEZ.

Zda se aplikace znovu připojí automaticky, závisí na hodnotě atributu kanálu IBM MQ DefReconnect.

Toto je výchozí hodnota.

Chcete-li použít tuto hodnotu z programu, použijte konstantu WMQConstants.WMQ_CLIENT_RECONNECT_AS_DEF.

CLIENTRECONNECTTIMEOUT

Čas ukončení opakování pokusů o opětovné připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : CLIENTRECONNECTIMEOUT

Krátký název nástroje pro administraci produktu JMS : CRT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setClientReconnectTimeout()
- MQConnectionFactory.setClientReconnectTimeout()

Hodnoty

Interval v sekundách. Předvolba 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 produktu JMS : CLIENTID

Krátký název nástroje pro administraci produktu JMS : CID

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setClientID ()
- MQConnectionFactory.getClientID ()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

CLONESUPP

Určuje, zda mohou být dvě nebo více instancí stejného odběratele trvalého tématu spuštěny souběžně.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : CLONESUPP

Krátký název nástroje pro administraci produktu JMS : CLS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setCloneSupport ()
- MQConnectionFactory. PodporagetClone()

Hodnoty

VYPNUTO
V daném okamžiku může být spuštěna pouze jedna instance trvalého odběratele tématu. Toto je výchozí hodnota.

POVOLENO
Dvě nebo více instancí stejného odběratele trvalého tématu lze spustit současně, ale každá instance musí být spuštěna v samostatném virtuálním počítači Java (JVM).

COMPHDR

Seznam technik, které lze použít pro kompresi dat záhlaví na připojení.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : COMPHDR

Krátký název nástroje pro administraci produktu JMS : HC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

Hodnoty

ŽÁDNÉ
Toto je výchozí hodnota.

SYSTÉM
Komprese hlavičky zprávy RLE se provádí.

COMPMSG

Seznam technik, které lze použít ke komprimování dat zpráv v rámci připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : COMPMSG

Krátký název nástroje pro administraci produktu 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 aplikací produktu IBM MQ classes for JMS , které používají přenos vazeb, k tomuto správci front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory.

Dlouhý název nástroje pro administraci produktu JMS : CONNOPT

Krátký název nástroje pro administraci produktu JMS : CNOPT

Programový přístup

Metody Setter/getter

- MQConnectionFactory. Volby volbysetMQConnection()
- MQConnectionFactory. Volby volbygetMQConnection()

Hodnoty

STANDARD
Povaha vazby mezi aplikací a správcem front závisí na hodnotě atributu *DefaultBindType* správce front. Hodnota STANDARD mapuje na IBM MQ *ConnectOption* MQCNO_STANDARD_BINDING.

SHARED
Aplikace a lokální agent správce front běží v samostatných jednotkách zpracování, ale sdílejí některé prostředky. Tato hodnota je mapována na IBM MQ *ConnectOption* MQCNO_SHARED_BINDING.

Izolovaný
Aplikace a lokální agent správce front běží v samostatných jednotkách provedení a nesdílejí žádné prostředky. Hodnota ISOLATED je mapována na IBM MQ *ConnectOption* MQCNO_ISOLATED_BINDING.

Rychlý
Aplikace a lokální agent správce front se spouští ve stejné jednotce provedení. Tato hodnota je mapována na IBM MQ *ConnectOption* MQCNO_FASTPATH_BINDING.

SERIALQM
Aplikace vyžaduje výlučné použití značky připojení v rámci oboru správce front. Tato hodnota je mapována na IBM MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_Q_MGR.

SERIALQSG
Aplikace vyžaduje výlučné použití značky připojení v rámci rozsahu skupiny sdílení front, do níž správce front patří. Hodnota SERIALQSG je mapována na IBM MQ *ConnectOption* MQCNO_SERIALIZE_CONN_TAG_QSG.

OMEZENÍQM
Aplikace vyžaduje sdílené použití značky připojení, existují však omezení pro sdílené použití značky připojení v rámci oboru správce front. Tato hodnota je mapována na IBM MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_Q_MGR.

RESTRICTQSG
Aplikace vyžaduje sdílené použití značky připojení, existují však omezení pro sdílené použití značky připojení v rámci oboru skupiny sdílení front, do níž správce front náleží. Tato hodnota je mapována na IBM MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_QSG.

Další informace o volbách připojení produktu IBM MQ naleznete v tématu Připojení ke správci front pomocí volání MQCONNX.

CONNTAG

Značka, kterou správce front přidružuje k prostředkům aktualizovaným aplikací v rámci jednotky práce, zatímco aplikace je připojena ke správci front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : CONNTSAG

Krátký název nástroje pro administraci produktu JMS : CNTRAG

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 oříznutá, pokud je delší než 128 bajtů.

DESCRIPTION

Popis uloženého objektu.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje administrace JMS : DESCRIPTION

Krátký název nástroje pro administraci produktu JMS : DESC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec
`

DIRECTAUTH

Určuje, zda je ověřování TLS použito v reálném čase připojení k danému zprostředkovateli.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : DIRECTAUTH

Krátký název nástroje pro administraci produktu JMS : DAUTH

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth ()

Hodnoty

ZÁKLADNÍ
Bez ověření, ověření jména uživatele, nebo ověření hesla. Toto je výchozí hodnota.

Certifikát
Ověřování pomocí certifikátu veřejného klíče.

ENCODING

Způsob, jakým jsou číselná data v těle zprávy reprezentována při odeslání zprávy do tohoto místa určení. Vlastnost uvádí znázornění binárních celých čísel, pakovaných dekadických celých čísel a čísel s pohyblivou řádovou čárkou.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : ENCODING

Krátký název nástroje pro administraci produktu JMS : ENC

Programový přístup

Metody Setter/getter

- MQDestination.setEncoding()
- MQDestination.getEncoding()

Hodnoty

Vlastnost ENCODING

Platné hodnoty, které může vlastnost produktu ENCODING použít, jsou konstruovány ze tří dílčích vlastností:

Kódování celých čísel
Buď normální, nebo obrácené

Kódování desetinných čísel
Buď normální, nebo obrácené

kódování čísel s
IEEE normální, IEEE reverzní, nebo z/OS

Vlastnost ENCODING je vyjádřena tříznakovým řetězcem 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 *celočíselné kódování*
- Druhý znak představuje *dekadické kódování*
- Třetí znak představuje *kódování s pohyblivou řádovou čárkou*

Tento parametr poskytuje sadu dvanácti možných hodnot pro vlastnost ENCODING .

Existuje další hodnota, řetězec NATIVE, který nastavuje vhodné hodnoty kódování pro platformu Java .

Následující příklady ukazují platné kombinace pro ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

EXPIRY

Doba, po jejímž uplynutí vyprší platnost zpráv v místě určení.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : EXPIRY

Krátký název nástroje pro administraci produktu JMS : EXP

Programový přístup

Metody Setter/getter

- MQDestination.setExpiry()
- MQDestination.getExpiry()

Hodnoty

Aplikace
Ukončení platnosti může být definováno aplikací JMS . Toto je výchozí hodnota.

NELIM
Nevyskytne se vypršení platnosti.

0
Nevyskytne se vypršení platnosti.

Libovolné kladné celé číslo představující vypršení platnosti v milisekundách.

FAILIFQUIESCE

Tato vlastnost určuje, zda volání určitých metod selže, pokud se správce front nachází ve stavu uvedení do klidového stavu nebo pokud se aplikace připojuje ke správci front pomocí přenosu CLIENT a kanál, který aplikace používá, byl uveden do klidového stavu, například pomocí příkazu MQSC **STOP CHANNEL** nebo **STOP CHANNEL MODE(QUIESCE)** .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, Topic, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : FAILIFQUIESCE

Krátký název nástroje pro administraci produktu JMS : FIQ

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setFailIfQuiesce()
- MQConnectionFactory.getFailIfQuiesce()

Hodnoty

YES
Volání určitých metod selže, pokud se správce front nachází ve stavu uvedení do klidového stavu nebo kanál používaný pro připojení ke správci front je uváděn do klidového stavu. Pokud aplikace zjistí některou z těchto podmínek, může dokončit její okamžitou úlohu a zavřít připojení, což umožní zastavení správce front nebo instance kanálu. Toto je výchozí hodnota.

NO
Volání metody se nezdařilo, protože správce front nebo kanál používaný pro připojení ke správci front se nachází ve stavu uvedení do klidového stavu. Zadáte-li tuto hodnotu, nebude moci aplikace zjistit, zda je správce front nebo kanál uveden do klidového stavu. Aplikace může pokračovat v provádění operací se správcem front, a zabránit tak zastavení správce front.

HOSTNAME

Pro připojení ke správci front se jedná o název hostitele nebo adresu IP systému, na kterém je správce front spuštěn, nebo v reálném čase pro připojení ke zprostředkovateli název hostitele nebo adresu IP systému, na kterém je zprostředkovatel spuštěn.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : HOSTNAME

Krátký název nástroje pro administraci produktu JMS : HOST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setHostName ()
- MQConnectionFactory.getHostNázev ()

Hodnoty

lokální hostitel
Toto je výchozí hodnota.

Libovolný platný řetězec

LOCALADDRESS

Pro připojení ke správci front určuje tato vlastnost buď lokální síťové rozhraní, které má být použito, nebo lokální port nebo rozsah lokálních portů, které mají být použity.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : LOCALADDRESS

Krátký název nástroje pro administraci produktu JMS : LA

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setLocalAdresa ()
- MQConnectionFactory.getLocalAdresa ()

Hodnoty

"" (prázdný řetězec)
Toto je výchozí hodnota.

Řetězec ve formátu [adresa_ip] [(počáteční_port [, koncový_port])]
Několik příkladů:

`192.0.2.0`

Kanál se lokálně spojí s adresou 192.0.2.0 .

`192.0.2.0(1000)`

Kanál se lokálně spojí s adresou 192.0.2.0 a bude používat port 1000.

`192.0.2.0(1000,2000)`

Kanál se lokálně spojí s adresou 192.0.2.0 a bude používat port v rozsahu 1000 až 2000.

`(1000)`

Kanál se lokálně spojí s portem 1000.

`(1000,2000)`

Kanál se lokálně spojí s portem v rozsahu 1000 až 2000.

Místo adresy IP můžete zadat název hostitele. V případě připojení v reálném čase k zprostředkovateli 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 mají hodnotu null, adresu IP nebo název hostitele.

STYL MAPNAMESTYLE

Umožňuje použít styl kompatibility pro názvy prvků MapMessage .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : MAPNAMESTYLE

Krátký název nástroje pro administraci produktu JMS : MNST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

Hodnoty

STANDARD
Je použit standardní formát pojmenování prvků com.ibm.jms.JMSMapMessage . Jedná se o výchozí hodnotu a umožňuje použití nezákonných identifikátorů Java jako názvu prvku.

Kompatibilní
Použije se starší formát com.ibm.jms.JMSMapMessage pro pojmenování prvků. Jako název prvku lze použít pouze platné identifikátory Java . To je potřeba pouze tehdy, když se odesílají zprávy mapování na aplikaci používající verzi produktu IBM MQ classes for JMS starší než 5.3.

MAXBUFFSIZE

Maximální počet přijatých zpráv, které mohou být uloženy ve vnitřní vyrovnávací paměti zpráv během čekání na zpracování aplikací. Tato vlastnost se použije pouze v případě, že hodnota TRANSPORT má hodnotu DIRECT nebo DIRECTHTTP.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : MAXBUFFSIZE

Krátký název nástroje pro administraci produktu 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 aplikace JMS může extrahovat hodnoty z polí MQMD.

Použitelné objekty

Dlouhý název nástroje pro administraci produktu JMS : MDREAD

Krátký název nástroje pro administraci produktu JMS : MDR

Programový přístup

Metody Setter/getter

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

Hodnoty

NO
Při odesílání zpráv nejsou vlastnosti JMS_IBM_MQMD* v odeslané zprávě aktualizovány tak, aby odrážely aktualizované hodnoty polí v produktu MQMD. Při příjmu zpráv nejsou dostupné žádné z vlastností JMS_IBM_MQMD* v přijaté zprávě, i když odesilatel některé či všechny tyto vlastnosti nastavil. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte False.

Ano
Při odesílání zpráv jsou všechny vlastnosti JMS_IBM_MQMD* v odeslané zprávě aktualizovány tak, aby odrážely aktualizované hodnoty polí v MQMD, včetně vlastností, které odesilatel explicitně nenastavila. Při příjmu zpráv jsou všechny vlastnosti JMS_IBM_MQMD* dostupné v přijaté zprávě, včetně vlastností, které odesilatel explicitně nenastavili.

V případě programů použijte hodnotu True.

MDWRITE

Tato vlastnost určuje, zda může aplikace JMS nastavit hodnoty polí MQMD.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : MDWRITE

Krátký název nástroje pro administraci produktu JMS : MDR

Programový přístup

Metody Setter/getter

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

Hodnoty

NO
Všechny vlastnosti JMS_IBM_MQMD* jsou ignorovány a jejich hodnoty se nezkopírují do základní struktury MQMD. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte False.

YES
Vlastnosti JMS_IBM_MQMD* jsou zpracovány. Jejich hodnoty jsou zkopírovány do podkladové struktury MQMD.

V případě programů 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 produktu JMS : MDMSGCTX

Krátký název nástroje pro administraci produktu JMS : MDCTX

Programový přístup

Metody Setter/getter

- MQDestination.setMQMDMessageContext()
- MQDestination.getMQMDMessageContext()

Hodnoty

DEFAULT
Volání rozhraní MQOPEN API a struktura MQPMO neurčují žádné explicitní volby kontextu zprávy. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQ_MDCTX_DEFAULT.

KONTEXT SET_IDENTITY_CONTEXT
Volání rozhraní MQOPEN API určuje volbu kontextu zprávy MQOO_SET_IDENTITY_CONTEXT a struktura MQPMO určuje hodnotu MQPMO_SET_IDENTITY_CONTEXT.

Pro programy použijte WMQ_MDCT_X_SET_IDENTITY_CONTEXT.

NASTAVITEL_VŠECH_KONTEXTU
Volání rozhraní MQOPEN API uvádí volbu kontextu zprávy MQOO_SET_ALL_CONTEXT a struktura MQPMO určuje MQPMO_SET_ALL_CONTEXT.

Pro programy použijte WMQ_MDCT_X_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 produktu JMS : MAXBUFFSIZE

Krátký název nástroje pro administraci produktu 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 ke zprávě MQRFH2 zprávy IBM MQ jako součásti informačního obsahu zprávy.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : WMQ_MESSAGE_BODY

Krátký název nástroje pro administraci produktu 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 nezahrnuje záhlaví MQRFH2 v závislosti na hodnotě WMQ_TARGET_CLIENT. Když přijímá, působí jako hodnota JMS.

JMS
Při odesílání produkt IBM MQ classes for JMS automaticky vygeneruje záhlaví MQRFH2 a zahrne jej do zprávy produktu IBM MQ .

Při příjmu IBM MQ classes for JMS nastavte vlastnosti zprávy JMS podle hodnot v MQRFH2 (pokud existuje). Neprezentuje MQRFH2 jako část těla zprávy JMS .

MQ
Při odesílání produkt IBM MQ classes for JMS negeneruje MQRFH2.

Při příjmu produkt IBM MQ classes for JMS prezentuje MQRFH2 jako část těla zprávy produktu JMS .

MSGRETENTION

Určuje, zda spotřebitel připojení uchovává nedoručené zprávy ve vstupní frontě.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

Dlouhý název nástroje pro administraci produktu JMS : MSGRETENTION

Krátký název nástroje pro administraci produktu JMS : MRET

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMessageRetention ()
- MQConnectionFactory.getMessageRetention ()

Hodnoty

Ano
Nedoručené zprávy zůstanou ve vstupní frontě. Toto je výchozí hodnota.

Ne
Nedoručené zprávy se řeší podle jejich dispozičních voleb.

MSGSELECTION

Určuje, zda je výběr zpráv prováděn serverem IBM MQ classes for JMS nebo zprostředkovatelem. Má-li funkce TRANSPORT hodnotu DIRECT, je výběr zprávy vždy prováděn zprostředkovatelem a hodnota MSGVÝBĚR je ignorována. Výběr zpráv zprostředkovatelem není podporován, když má BROKERVER hodnotu V1.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : MSGSELECTION

Krátký název nástroje pro administraci produktu 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 v reálném čase pro zprostředkovatele a je-li to povoleno, určete přesný způsob, jakým se výběrové vysílání používá k doručování zpráv od zprostředkovatele ke spotřebiteli zpráv. Vlastnost nemá žádný vliv na způsob, jakým Producent zpráv odesílá zprávy zprostředkovateli.

Použitelné objekty

ConnectionFactory, TopicConnectionTovárna, Téma

Dlouhý název nástroje pro administraci produktu JMS : MULTICAST

Krátký název nástroje pro administraci produktu JMS : MCAST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMulticast()
- MQConnectionFactory.getMulticast()

Hodnoty

VYPNUTO
Zprávy nejsou doručovány spotřebiteli zpráv pomocí výběrového vysílání. Jedná se o výchozí hodnotu pro objekty továrny ConnectionFactory a TopicConnection.

ASCF
Zprávy jsou doručovány spotřebiteli zpráv v souladu s nastavením výběrového vysílání pro továrnu připojení přidruženou k odběrateli zpráv. Výběrové nastavení multicast pro továrnu připojení je zaznamenáno v době vytvoření spotřebitele zprávy. Tato hodnota je platná pouze pro objekty Topic a je výchozí hodnotou pro objekty Topic.

POVOLENO
Je-li téma nakonfigurováno pro výběrové vysílání ve zprostředkovateli, zprávy se doručují spotřebiteli zpráv pomocí výběrového vysílání. Spolehlivá kvalita služeb se používá, je-li téma nakonfigurováno pro spolehlivé výběrové vysílání.

Spolehlivé
Je-li téma nakonfigurováno pro spolehlivé výběrové vysílání ve zprostředkovateli, zprávy se doručují spotřebiteli zpráv pomocí výběrového vysílání se spolehlivou kvalitou služeb. Není-li téma konfigurováno pro spolehlivé výběrové vysílání, nemůžete pro dané téma vytvořit spotřebitele zpráv.

NTR.
Je-li téma nakonfigurováno pro výběrové vysílání ve zprostředkovateli, jsou zprávy doručovány spotřebiteli zpráv pomocí výběrového vysílání. Spolehlivá kvalita služeb se nepoužívá ani v případě, že je téma nakonfigurováno pro spolehlivé výběrové vysílání.

OPTIMISTICPUBLICATION

Tato vlastnost určuje, zda produkt IBM MQ classes for JMS okamžitě vrátí řízení vydavateli, který publikoval zprávu, nebo zda vrátí řízení až poté, co dokončí veškeré zpracování přidružené k volání, a může oznámit výsledek vydavateli.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : OPTIMISTICPUBLIKACE

Krátký název nástroje pro administraci produktu 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 veškerá zpracování přidružená 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, dříve než dokončí veškeré zpracování přidružené k volání a může oznámit výsledek vydavateli. Produkt IBM MQ classes for JMS oznámí výsledek pouze v případě, že vydavatel zprávu potvrdí.

OUTCOMENOTIFICATION

Tato vlastnost určuje, zda produkt IBM MQ classes for JMS okamžitě vrátí řízení na odběratele, který právě potvrdil nebo potvrdil zprávu, nebo zda vrátí řízení až poté, co dokončí veškeré zpracování přidružené k volání a může ohlásit výsledek odběrateli.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : OUTCOMENOTIFICATION

Krátký název nástroje pro administraci produktu 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 veškerá zpracování přidružená 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 okamžitě vrátí řízení odběrateli, než dokončí veškeré zpracování přidružené k volání a může ohlásit výsledek odběrateli.

PERSISTENCE

Trvalost zpráv odeslaných do místa určení.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : PERSISTENCE

Krátký název nástroje pro administraci produktu JMS : PER

Programový přístup

Metody Setter/getter

- MQDestination.setPersistence()
- MQDestination.getPersistence()

Hodnoty

Aplikace
Perzistence je definována aplikací JMS . Toto je výchozí hodnota.

QDEF
Persistence přebírá hodnotu předvolby fronty.

PERRY
Zprávy jsou trvalé.

JINÝ
Zprávy jsou přechodné.

VYSOKÁ
Další informace o použití této hodnoty naleznete v tématu Perzistentní zprávy produktuJMS .

POLLINGINT

Pokud každý modul listener pro zprávy v rámci relace nemá ve své frontě žádnou vhodnou zprávu, je tento maximální interval, v milisekundách, který uplyne, než se každý modul listener pro zprávy znovu pokusí o získání zprávy z fronty. Pokud se často stává, že pro žádný z listenerů zpráv v rámci relace není k dispozici žádná vhodná zpráva, zvažte zvýšení hodnoty této vlastnosti. Tato vlastnost je relevantní pouze v případě, že hodnota TRANSPORT má hodnotu BIND nebo CLIENT.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : POLLINGINT

Krátký název nástroje pro administraci produktu 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 je to číslo portu, na kterém správce front naslouchá, nebo v reálném čase pro připojení k zprostředkovateli číslo portu, na kterém zprostředkovatel naslouchá připojením v reálném čase.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : PORT

Krátký název nástroje pro administraci produktu JMS : PORT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

Hodnoty

1414
Jedná se o výchozí hodnotu, pokud je funkce TRANSPORT nastavena na hodnotu CLIENT.

1506
Jedná se o výchozí hodnotu, pokud je funkce TRANSPORT nastavena na DIRECT nebo DIRECTHTTP.

Libovolné kladné celé číslo

PRIORITY

Priorita pro zprávy odeslané do místa určení.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : PRIORITY

Krátký název nástroje pro administraci produktu 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 až 9
Nejnižší na nejvyšší.

PROCESSDURATION

Tato vlastnost určuje, zda odběratel zaručuje rychlé zpracování jakékoli zprávy, kterou obdrží před vrácením řízení do produktu IBM MQ classes for JMS.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : PROCESSSDURATION

Krátký název nástroje pro administraci produktu JMS : PROCDUR

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setProcessDuration ()
- MQConnectionFactory.getProcessDuration ()

Hodnoty

NEZNÁMÉ
Odběratel nemůže poskytnout žádnou záruku, jak rychle může zpracovat jakoukoli zprávu, kterou obdrží. Toto je výchozí hodnota.

Krátký
Odběratel zaručuje, že proces rychle zpracuje všechny zprávy, které obdrží před vrácením řízení do produktu IBM MQ classes for JMS.

PROVIDERVERSION

Tato vlastnost rozlišuje mezi třemi režimy systému zpráv produktu IBM MQ : IBM MQ poskytovatele systému zpráv, normální režim, normální režim poskytovatele systému zpráv produktu IBM MQ s omezeními a režim migrace poskytovatele systému zpráv produktu IBM MQ .

Normální režim poskytovatele systému zpráv produktu IBM MQ používá všechny funkce správce front produktu IBM MQ k implementaci produktu JMS. Tento režim je optimalizovaný pro použití rozhraní API a funkčnosti produktu JMS 2.0 . Normální režim poskytovatele systému zpráv produktu IBM MQ s omezeními používá rozhraní JMS 2.0 API, ale ne nové funkce, jako jsou sdílené odběry, odložené doručení nebo asynchronní odeslání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : PROVIDERVERSION

Krátký název nástroje pro administraci produktu JMS : PVER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setProviderVerze ()
- MQConnectionFactory.getProviderVerze ()

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 prostřednictvím příkazu úrovně 800, pak se budou používat všechna rozhraní API a funkce JMS 2.0, jako např. asynchronní odesílání, odložené doručení nebo sdílené odběry.

Není-li správce front, který je uvedený v nastavení továrny na připojení, správcem front IBM MQ 8.0.0, metoda `createConnection` selže 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. Není-li správce front, který je uvedený v nastavení továrny na připojení, správcem front IBM WebSphere MQ 7.0.1 nebo novější, metoda `createConnection` selže s výjimkou `JMSFCC5008`.

Pokud se pomocí normálního režimu s omezeními připojujete ke správci front s příkazem úrovně mezi 700 a 800, můžete používat rozhraní JMS 2.0 API, ale ne funkce pro asynchronní odeslání, odložené doručení nebo sdílené odběry.

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.

Produkt IBM MQ classes for JMS používá funkce a algoritmy dodávané s produktem IBM WebSphere MQ 6.0. Chcete-li se připojit k produktu WebSphere Message Broker 6.0 nebo 6.1 pomocí protokolu IBM WebSphere MQ Enterprise Transport 6.0, musíte použít tento režim. Ve správci front IBM MQ 8.0 se můžete připojit pomocí tohoto režimu, ale žádné z nových funkcí správce front IBM MQ classes for JMS, např. dopředné čtení nebo posílání dat v proudu, se nebudou používat.

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 vlastnost **BROKERVER** nastavena na hodnotu V1 nebo výchozí hodnotu *nespecifikováno*, bude nadále používán normální režim poskytovatele systému zpráv IBM MQ a proto budou všechny operace publikování/odběru používat nové funkce verze IBM WebSphere MQ 7.0.1 nebo novější.

   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 produktu 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, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : PROXYHOSTNAME

Krátký název nástroje pro administraci produktu 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 naslouchá server proxy při použití připojení v reálném čase ke zprostředkovateli prostřednictvím serveru proxy.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : PROXYPPORT

Krátký název nástroje pro administraci produktu JMS : PPPORT

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 požádá o potvrzení od zprostředkovatele.

Když snížíte hodnotu této vlastnosti, IBM MQ classes for JMS vyžádá potvrzení častěji, proto se výkon vydavatele sníží. Když zvýšíte hodnotu, produkt IBM MQ classes for JMS trvá déle, než dojde k vyvolání výjimky, pokud se zprostředkovatel nezdaří. Tato vlastnost je relevantní pouze v případě, že hodnota TRANSPORT má hodnotu BIND nebo CLIENT.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : PROXYPPORT

Krátký název nástroje pro administraci produktu JMS : PPPORT

Programový přístup

Metody Setter/getter

MQConnectionFactory.setPubAckInterval()

MQConnectionFactory.getPubAckInterval()

Hodnoty

25
Libovolné kladné celé číslo může být výchozí hodnota.

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 produktu JMS : PUTASYNCALLOWED

Krátký název nástroje pro administraci produktu JMS : PAALD

Programový přístup

Metody Setter/getter

MQDestination.setPutAsyncAllowed()

MQDestination.getPutAsyncAllowed()

Hodnoty

NEJDEJŠÍ
Určete, zda jsou asynchronní operace vložení povoleny, odkazem na definici fronty nebo tématu. Toto je výchozí hodnota.

DEFINICE AS_Q_DEF
Určete, zda jsou asynchronní operace vložení povoleny odkazem na definici fronty.

DEFINICE AS_TOPIC_DEF
Určete, zda jsou asynchronní operace vložení povoleny odkazem na definici tématu.

NO
Asynchronní operace put nejsou povoleny.

YES
Asynchronní operace put jsou povoleny.

QMANAGER

Název správce front, s nímž má být navázáno připojení.

Pokud však vaše aplikace používá k připojení ke správci front tabulku definic kanálů klienta, 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 produktu JMS : QMANAGER

Krátký název nástroje pro administraci produktu JMS : QMGR

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setQueueManager ()
- MQConnectionFactory.getQueueManager ()

Hodnoty

"" (prázdný řetězec)
Jako výchozí hodnota může být libovolný řetězec.

QUEUE

Název cíle fronty produktu JMS . Shoduje se s názvem fronty, kterou používá správce front.

Použitelné objekty

Fronta

Dlouhý název nástroje pro administraci produktu JMS : QUEUE

Krátký název nástroje pro administraci produktu JMS : QU

Hodnoty

Libovolný řetězec
Libovolné platné jméno fronty IBM MQ .

Související pojmy
Pravidla pro pojmenování objektů IBM MQ >

READAHEADALLOWED

Tato vlastnost určuje, zda mají spotřebitelé zpráv a prohlížečové prohlížeče povoleno používat dopředné čtení k získání přechodných zpráv z tohoto místa určení do interní vyrovnávací paměti, než je přijme.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : READAHEADALLOWED

Krátký název nástroje pro administraci produktu JMS : RAALD

Programový přístup

Metody Setter/getter

- MQDestination.setReadAheadAllowed()
- MQDestination.getReadAheadAllowed()

Hodnoty

NEJDEJŠÍ
Určete, zda je dopředné čtení povoleno s odkazem na definici fronty nebo tématu. Jedná se o výchozí hodnotu v administrativních nástrojích.

Použijte parametr WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_DEST v programech.

DEFINICE AS_Q_DEF
Určete, zda je dopředné čtení povoleno s odkazem na definici fronty.

Použijte parametr WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_Q_DEF v programech.

DEFINICE AS_TOPIC_DEF
Určete, zda je dopředné čtení povoleno s odkazem na definici tématu.

V programech použijte WMQConstants.WMQ_READ_AHEAD_ALLOWED_AS_TOPIC_DEF .

NO
Čtení napřed není povoleno.

Použijte program WMQConstants.WMQ_READ_AHEAD_ALLOWED_DISABLED v programech.

YES
Čtení napřed je povoleno.

Použijte program WMQConstants.WMQ_READ_AHEAD_ALLOWED_ENABLED v programech.

READAHEADCLOSEPOLICY

Pro zprávy doručené do asynchronního modulu listener zpráv, co se stane se zprávami v interní vyrovnávací paměti dopředného čtení, když je spotřebitel zpráv uzavřen.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : READAHEADCLOSEPOLICY

Krátký název nástroje pro administraci produktu JMS : RACP

Programový přístup

Metody Setter/getter

- MQDestination.setReadAheadClosePolicy()
- MQDestination.getReadAheadClosePolicy()

Hodnoty

DORUČIT VŠE
Všechny zprávy v interní vyrovnávací paměti dopředného čtení jsou doručovány do posluchače zpráv aplikace před návratem. Jedná se o výchozí hodnotu v administrativních nástrojích.

Použijte program WMQConstants.WMQ_READ_AHEAD_DELIVERALL v programech.

AKTUÁLNÍ_DORUČENÍ
Před návratem bude dokončeno pouze aktuální vyvolání modulu listener pro zprávy, případně zanechání zpráv v interní vyrovnávací paměti dopředného čtení, které se pak vyřadí.

Použijte program WMQConstants.WMQ_READ_AHEAD_DELIVERCURRENT v programech.

RECEIVECCSID

Vlastnost místa určení, která nastavuje cílové CCSID pro převod zpráv správce front.
Hodnota je ignorována, pokud není hodnota RECEIVECONVERSION nastavena na hodnotu `WMQ_RECEIVE_CONVERSION_QMGR` .

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : RECEIVECCSID

Krátký název nástroje pro administraci produktu 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

Cílová vlastnost, která určuje, zda má správce front provést převod dat.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : RECEIVECONVERSION

Krátký název nástroje pro administraci produktu JMS : RCNV

Programový přístup

Metody Setter/Getter

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

Hodnoty

`WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG`
1 -Provést konverzi dat pouze u klienta JMS . Výchozí hodnota z až V7.0, a od, a včetně, 7.0.1.5.

`WMQConstants.WMQ_RECEIVE_CONVERSION_QMGR`
2 -Provést konverzi dat na správci front před odesláním zprávy klientovi. Výchozí (a pouze) hodnota z V7.0 na V7.0.1.4 včetně, s výjimkou případů, kdy je použita oprava APAR IC72897 .

RECEIVEISOLATION

Tato vlastnost určuje, zda odběratel může přijímat zprávy, které nebyly potvrzeny ve frontě odběratele.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : RECEIVEISOLATION

Krátký název nástroje pro administraci produktu JMS : RCVISOL

Hodnoty

POTVRZENO
Odběratel obdrží pouze ty zprávy ve frontě odběratele, které byly potvrzeny. Jedná se o výchozí hodnotu v administrativních nástrojích.

Použijte program WMQConstants.WMQ_RCVISOL_COMMITTED v programech.

NEPOTVRZENO
Odběratel může přijímat zprávy, které nebyly potvrzeny ve frontě odběratele.

Použijte program WMQConstants.WMQ_RCVISOL_UNCOMMITTED v programech.

RECEXIT

Označuje uživatelskou proceduru pro přijetí zprávy kanálu nebo posloupnost uživatelských procedur pro příjem, které mají být spuštěny v posloupnosti.

Je možné, že bude vyžadována další konfigurace, aby produkt IBM MQ classes for JMS mohl vyhledat uživatelské procedury pro příjem. Další informace naleznete v tématu Konfigurace tříd produktu IBM MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : RECEXIT

Krátký název nástroje pro administraci produktu 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, přičemž každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQReceiveExit` (pro uživatelskou proceduru pro přijetí zprávy kanálu je zapsána v produktu Java).
  - Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru pro přijetí zprávy kanálu, která není zapsána v produktu Java).

RECEXITINIT

Uživatelská data, která jsou předána uživatelským procedurám přijetí kanálu, když jsou volána.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : RECEXITINIT

Krátký název nástroje pro administraci produktu JMS : RCXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

Hodnoty

null
Řetězec obsahující jednu nebo více položek dat uživatele oddělených čárkami. Toto je výchozí hodnota.

REPLYTOSTYLE

Určuje, jak je vytvořeno pole JMSReplyTo v přijaté zprávě.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : REPLYTOSTYLE

Krátký název nástroje pro administraci produktu JMS : RTOST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setReplyToStyle()
- MQConnectionFactory.getReplyToStyle()

Hodnoty

DEFAULT
Ekvivalentní proměnné `MQMD`.

RFH2
Použijte hodnotu zadanou v záhlaví RFH2 . Je-li v odesílající aplikaci nastavena hodnota JMSReplyTo , použijte tuto hodnotu.

MQMD
Použijte zadanou hodnotu MQMD. Toto chování je ekvivalentní výchozímu chování produktu 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 svůj vlastní název v MQMD. Pokud tento parametr nastavíte na hodnotu `MQMD`, bude fronta odpovědí, kterou používáte, na přijímajícím správci front. Nastavíte-li tento parametr na hodnotu RFH2, bude fronta odpovědí, kterou používáte, ve správci front uvedeném v záhlaví RFH2 odeslané zprávy, jak byla původně nastavena odesílající aplikací.

Pokud hodnota parametru JMSReplyTo nastavená odesílající aplikací obsahuje název správce front, je hodnota tohoto parametru nedůležitá, protože oba prvky 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 prohledá frontu IBM MQ pro vhodné zprávy v pořadí určeném atributem `MsgDeliverySequence` fronty.

Poté, co produkt IBM MQ classes for JMS najde vhodnou zprávu a doručí ji spotřebiteli, produkt IBM MQ classes for JMS obnoví hledání další vhodné zprávy od své aktuální pozice ve frontě. Produkt IBM MQ

classes for JMS pokračuje v hledání fronty tímto způsobem, dokud nedosáhne konce fronty, nebo dokud nevyprší časový interval v milisekundách určený hodnotou této vlastnosti. V každém případě se IBM MQ classes for JMS vrátí na začátek fronty, aby pokračovalo hledání, a začíná nový časový interval.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : RESCANINT

Krátký název nástroje pro administraci produktu JMS : RINT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setRescanInterval ()
- MQConnectionFactory.getRescanInterval ()

Hodnoty

5000
Libovolné kladné celé číslo může být výchozí hodnota.

SECEXIT

Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.

Je možné, že bude vyžadována další konfigurace, aby produkt IBM MQ classes for JMS vyhledal uživatelské procedury zabezpečení. Další informace naleznete v tématu Konfigurace tříd produktu IBM MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SECEXIT

Krátký název nástroje pro administraci produktu 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, přičemž každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQSecurityExit` (pro uživatelskou proceduru zabezpečení kanálu napsanou v produktu Java).
  - Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru zabezpečení kanálu, která není zapsána v produktu 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 produktu JMS : SECEXITINIT

Krátký název nástroje pro administraci produktu JMS : SCXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

Hodnoty

null
Jako výchozí hodnota může být libovolný řetězec.

SENDCHECKCOUNT

Počet volání odesílání, která umožňují mezi kontrolou asynchronních chyb vložení, v rámci jedné relace JMS bez transakce.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SENDCHECKCOUNT

Krátký název nástroje pro administraci produktu JMS : SCC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSendCheckCount()
- MQConnectionFactory.getSendCheckCount()

Hodnoty

null
Jako výchozí hodnota může být libovolný řetězec.

SENDEXIT

Identifikuje uživatelskou proceduru pro odeslání zprávy kanálu nebo posloupnost uživatelských procedur odeslání, které mají být spuštěny v posloupnosti.

Je možné, že bude vyžadována další konfigurace, aby produkt IBM MQ classes for JMS mohl vyhledat uživatelské procedury odeslání. Další informace naleznete v tématu Konfigurace tříd produktu IBM MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SENDEXIT

Krátký název nástroje pro administraci produktu 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, přičemž každá položka je buď:
  - Název třídy, která implementuje rozhraní `WMQSendExit` (pro uživatelskou proceduru pro odeslání zprávy kanálu je zapsána v souboru Java).
  - Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru pro 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 produktu JMS : SENDEXITINIT

Krátký název nástroje pro administraci produktu JMS : SDXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

Hodnoty

null
Výchozí hodnota může být libovolný řetězec obsahující jednu nebo více položek dat uživatele 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, TopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SPARSESUBS

Krátký název nástroje pro administraci produktu JMS : SSUBS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSparseOdběry ()
- MQConnectionFactory.getSparseOdběry ()

Hodnoty

NO
Odběry přijímají časté odpovídající zprávy. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte hodnotu false.

YES
Odběry přijímají zřídka odpovídající zprávy. Tato hodnota vyžaduje, aby byla fronta odběru otevřena pro procházení.

Pro programy použijte hodnotu true.

SSLCIPHERSUITE

Sada CipherSuite , která má být použita pro připojení TLS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SSLCIPHERSUITE

Krátký název nástroje pro administraci produktu JMS : SCPHS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory.getSSLCipherSuite ()

Hodnoty

null
Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti TLS pro objekty JMS.

SSLCRL

Servery CRL, které mají zkontrolovat odvolání certifikátu TLS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SSLCRL

Krátký název nástroje pro administraci produktu 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 naleznete v tématu Vlastnosti TLS pro objekty JMS.

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" . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C certificate a měli by si být vědomi jakýchkoli 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

Pro TLS je to kostra *rozlišujícího názvu* , která se musí shodovat s tou, kterou poskytuje správce front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SSLPEERNAME

Krátký název nástroje pro administraci produktu JMS : SPEER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLPeerNázev ()
- MQConnectionFactory.getSSLPeerNázev ()

Hodnoty

null
Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti TLS pro objekty JMS.

SSLRESETCOUNT

Pro TLS je celkový počet bajtů odeslaných a přijatých připojením před opětovným získáním tajného klíče, který se používá pro šifrování.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SSLRESETCOUNT

Krátký název nástroje pro administraci produktu JMS : SRC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLResetCount ()
- MQConnectionFactory.getSSLResetCount ()

Hodnoty

0
Nula nebo jakékoli kladné celé číslo menší nebo rovné 999, 999, 999. Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti TLS pro objekty JMS.

STATREFRESHINT

Interval (v milisekundách) mezi aktualizacemi transakce s dlouhou dobou zpracování, která zjišťuje, zda odběratel ztratil připojení ke správci front.

Tato vlastnost je relevantní pouze v případě, že hodnota SUBSTORE má hodnotu QUEUE.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : STATREFRESHINT

Krátký název nástroje pro administraci produktu JMS : SRI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

Hodnoty

60000
Libovolné kladné celé číslo může být výchozí hodnota. Další informace naleznete v tématu Vlastnosti TLS pro objekty JMS.

SUBSTORE

Kde IBM MQ classes for JMS ukládá trvalá data týkající se aktivních odběrů.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : SUBSTORE

Krátký název nástroje pro administraci produktu JMS : SS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSubscriptionStore ()

- MQConnectionFactory.getSubscriptionStore ()

Hodnoty

BROKER
Chcete-li uchovávat podrobnosti odběrů, použijte úložiště odběrů na základě zprostředkovatele. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_BROKER.

MIGRATE
Přenést informace o odběru z úložiště odběrů založeného na odběru do úložiště odběrů na bázi zprostředkovatele.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_MIGRATE.

QUEUE
Podrobnosti odběrů lze uchovávat v úložišti odběrů založeném na frontách.

Pro programy použijte WMQConstants.WMQ_SUBSTORE_QUEUE.

SYNCPOINTALLGETS

Tato vlastnost určuje, zda mají být v rámci synchronizačního bodu provedeny všechny operace typu get.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : SYNCPOINTALLGETS

Krátký název nástroje pro administraci produktu JMS : SPAG

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

Hodnoty

Ne
Toto je výchozí hodnota.

Ano

TARGCLIENT

Tato vlastnost určuje, zda má být použit formát IBM MQ RFH2 k výměně informací s cílovými aplikacemi.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci produktu JMS : TARGCLIENT

Krátký název nástroje pro administraci produktu JMS : TC

Programový přístup

Metody Setter/getter

- MQDestination.setTargetClient()

- MQDestination.getTargetClient()

Hodnoty

JMS
Cílem zprávy je aplikace JMS . Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_CLIENT_JMS_COMPLIANT.

MQ
Cílem zprávy je aplikace typu non-JMS IBM MQ .

Pro programy použijte WMQConstants.WMQ_CLIENT_NONJMS_MQ.

TARGCLIENTMATCHING

Tato vlastnost určuje, zda má zpráva odpovědi odeslaná do fronty označené v poli záhlaví JMSReplyTo příchozí zprávy záhlaví MQRFH2 pouze v případě, že má příchozí zpráva záhlaví MQRFH2 .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : TARGCLIENTMATCHING

Krátký název nástroje JMS Administration Tool: TCM

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTargetClientMatching()
- MQConnectionFactory.getTargetClientMatching()

Hodnoty

YES
Pokud příchozí zpráva nemá záhlaví MQRFH2 , vlastnost TARGCLIENT objektu fronty odvozeného z pole záhlaví JMSReplyTo zprávy se odešle do produktu MQ. Má-li zpráva záhlaví MQRFH2 , je vlastnost TARGCLIENT nastavena na hodnotu JMS . Jedná se o výchozí hodnotu 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 vytvářeny dočasné fronty produktu JMS .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : TEMPMODEL

Krátký název nástroje pro administraci produktu JMS : TM

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTemporaryModel ()
- MQConnectionFactory.getTemporaryModel ()

Hodnoty

SYSTEM.DEFAULT.MODEL.QUEUE
Jako výchozí hodnota může být libovolný řetězec.

TEMPQPREFIX

Předpona, která se používá k vytvoření názvu dynamické fronty produktu IBM MQ .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci produktu JMS : TEMPQPREFIX

Krátký název nástroje pro administraci produktu JMS : TQP

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

Hodnoty

" " (prázdný řetězec)
Použitá předpona je `CSQ.*` na systémech z/OS a `AMQ.*` na všech ostatních platformách. Jedná se o výchozí hodnoty.

předpona fronty
Předpona fronty je libovolný řetězec, který vyhovuje pravidlům pro vytváření obsahu pole *DynamicQName* v deskriptoru objektu IBM MQ (struktura MQOD), ale poslední nemezerový 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, jen '' TEMP / *unique_id* ''. Zadání neprázdné hodnoty TEMPTOPICPREFIX umožňuje definovat specifické modelové fronty pro vytvoření spravovaných front pro odběratele do dočasných témat vytvořených pod tímto připojením.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : TEMPTOPICPREFIX

Krátký název nástroje pro administraci produktu JMS : TTP

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

Hodnoty

Jakýkoli řetězec, který není null, obsahuje pouze platné znaky pro řetězec tématu IBM MQ . Výchozí hodnota je '' '' (prázdný řetězec).

TOPIC

Název místa určení tématu produktu JMS . Tato hodnota je používána správcem front jako řetězec tématu publikování nebo odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci produktu JMS : TOPIC

Krátký název nástroje pro administraci produktu JMS : TOP

Hodnoty

Libovolný řetězec

Řetězec, který tvoří platný řetězec tématu IBM MQ . Při použití IBM MQ jako poskytovatele systému zpráv s produktem WebSphere Application Serverzadejte hodnotu, která odpovídá názvu, pod kterým je toto téma známé pro účely administrace v rámci produktu WebSphere Application Server.

Související pojmy

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 produktu JMS : TRAN

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTransportTyp ()
- MQConnectionFactory.getTransportTyp ()

Hodnoty

BIND

Pro připojení ke správci front v režimu vazeb. Jedná se o výchozí hodnotu 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

V případě připojení v reálném čase ke zprostředkovateli, který nepoužívá tunelování HTTP.

Pro programy použijte WMQConstants.WMQ_CM_DIRECT_TCPIP.

DIRECTTTP
V případě připojení v reálném čase ke zprostředkovateli pomocí tunelování HTTP. Podporován je pouze protokol HTTP 1.0 .

Pro programy použijte WMQConstants.WMQ_CM_DIRECT_HTTP.

Související pojmy
“Závislosti mezi vlastnostmi objektů produktu IBM MQ classes for JMS” na stránce 1883
Platnost některých vlastností závisí na konkrétních hodnotách jiných vlastností.

WILDCARDFORMAT

Tato vlastnost určuje, která verze syntaxe zástupných znaků má být použita.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci produktu JMS : WILDCARDFORMAT

Krátký název nástroje pro administraci produktu JMS : WCFMT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setWildCardFormat()
- MQConnectionFactory.getWildCardFormat()

Hodnoty

POUZE TOPIC
Rozpoznává pouze zástupné znaky na úrovni tématu, které jsou použity ve zprostředkovateli verze 2. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_WILDCARD_TOPIC_ONLY.

POUZE CHAR_ONLY
Rozlišuje pouze zástupné znaky znaků, které jsou použity ve zprostředkovateli verze 1.

Pro programy použijte WMQConstants.WMQ_WILDCARD_CHAR_ONLY.

Vlastnost ENCODING

Vlastnost ENCODING se skládá ze tří dílčích vlastností, ve dvanácti možných kombinacích.

Platné hodnoty, které může vlastnost produktu ENCODING použít, jsou konstruovány ze tří dílčích vlastností:

Kódování celých čísel
Buď normální, nebo obrácené

Kódování desetinných čísel
Buď normální, nebo obrácené

kódování čísel s
IEEE normální, IEEE reverzní, nebo z/OS

Vlastnost ENCODING je vyjádřena tříznakovým řetězcem 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 *celočíselné kódování*
- Druhý znak představuje *dekadické kódování*
- Třetí znak představuje *kódování s pohyblivou řádovou čárkou*

Tento parametr poskytuje sadu dvanácti možných hodnot pro vlastnost ENCODING .

Existuje další hodnota, řetězec NATIVE, který nastavuje vhodné hodnoty kódování pro platformu Java .

Následující příklady ukazují platné kombinace pro ENCODING:

```
ENCODING(NNR)
ENCODING(NATIVE)
ENCODING(RR3)
```

Vlastnosti TLS pro objekty JMS

Povolte šifrování protokolu TLS (Transport Layer Security) s použitím vlastnosti SSLCIPHERSUITE. Charakteristiky šifrování TLS pak můžete změnit pomocí několika dalších vlastností.

Zadáte-li funkci TRANSPORT (CLIENT), můžete povolit šifrovanou komunikaci TLS pomocí vlastnosti SSLCIPHERSUITE. Nastavte tuto vlastnost na platnou sadu CipherSuite poskytovanou poskytovatelem JSSE; musí odpovídat hodnotě CipherSpec pojmenované v kanálu SVRCONN pojmenovaném vlastností CHANNEL.

Avšak CipherSpecs (určené v kanálu SVRCONN) a CipherSuites (jak je uvedeno v objektech ConnectionFactory) používají různé názvy schémat, aby představovaly stejné šifrovací algoritmy TLS. Je-li v vlastnosti SSLCIPHERSUITE zadán název CipherSpec , systém JMSAdmin vydá varování a mapuje CipherSpec na ekvivalentní sadu CipherSuite. Seznam CipherSpecs rozpoznaných produktem IBM MQ a JMSAdmin naleznete v tématu TLS CipherSpecs a CipherSuites v produktu IBM MQ classes for JMS .

Pokud vyžadujete připojení pro použití sady CipherSuite , která je podporována poskytovatelem prostředí JSSE FIPS produktu IBM Java (IBMJSSEFIPS), nastavte vlastnost SSLFIPSREQUIRED továrny na připojení na hodnotu YES. Výchozí hodnota této vlastnosti je NO, což znamená, že připojení může používat všechny podporované CipherSuite. Je-li SSLCIPHERSUITE nenastavená, je vlastnost ignorována.

Parametr SSLPEERNAME odpovídá formátu parametru SSLPEER, který lze nastavit v definicích kanálů. Jedná se o seznam dvojic název-hodnota 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 kontroluje identifikační certifikát prezentovaný serverem při připojování. Aby bylo připojení úspěšné, musí mít certifikát Common Name začínající QMGR., a musí mít alespoň dva názvy organizační jednotky, první z nich je IBM a druhý WEBSPHERE. Při kontrole se nerozlišují velká a malá písmena.

Pokud parametr SSLPEERNAME není nastaven, žádná taková kontrola se neprovede. SSLPEERNAME je ignorován, pokud není nastavena hodnota SSLCIPHERSUITE.

Vlastnost SSLCRL uvádí nula nebo více serverů CRL (Certificate Revocation List). Použití této vlastnosti vyžaduje prostředí JVM v umístění Java 2 v1.4. Jedná se o seznam položek oddělených mezerami:

```
ldap:// hostname:[ port ]
```

volitelně následováno jedním/. Je-li vynechán parametr *port* , předpokládá se výchozí port LDAP 389. Při připojování je certifikát TLS představený serverem kontrolován proti zadaným serverům CRL. Další informace o zabezpečení CRL viz Zabezpečení IBM MQ .

Není-li SSLCRL nastaven, žádná taková kontrola se neprovede. SSLCRL je ignorován, pokud SSLCIPHERSUITE není nastaveno.

Vlastnost SSLRESETCOUNT představuje celkový počet bajtů odeslaných a přijatých připojením před opětovným získáním tajného klíče, který je použit pro šifrování. Počet odeslaných bajtů je číslo před šifrováním a počet přijatých bajtů je číslo po dešifrování. Počet bajtů obsahuje také 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 použitím tajného klíče, který je znovu vyjednán po 4 MB dat, zadejte pro správce JMSAdmin následující příkaz:

```
ALTER CF(my.cf) SSLRESETCOUNT(4194304)
```

Je-li hodnota parametru SSLRESETCOUNT rovna nule, což je výchozí hodnota, nebude tajný klíč nikdy znovu vyjednáván. Vlastnost SSLRESETCOUNT je ignorována, není-li nastavena hodnota 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 produktem XMS.

.NETRozhraní

Tento oddíl popisuje rozhraní třídy .NET a jejich vlastnosti a metody.

Následující tabulka shrnuje rozhraní, která jsou definována v rámci oboru názvů produktu `IBM.XMS` .

Tabulka 872. Souhrn rozhraní třídy .NET

Rozhraní	**Popis**
“IBytesMessage” na stránce 1936	Bajtová zpráva je zpráva, jejíž tělo tvoří proud bajtů.
“IConnection” na stránce 1946	Objekt připojení představuje aktivní připojení aplikace k serveru systému zpráv.
“IConnectionFactory” na stránce 1948	Aplikace používá továrnu připojení k vytvoření připojení.
“Data IConnectionMetaData” na stránce 1950	Objekt datového objektu ConnectionMetaposkytuje informace o připojení.
“Instinace” na stránce 1950	Cíl je místo, kam aplikace odesílá zprávy, nebo je to zdroj, ze kterého aplikace přijímá zprávy, nebo obojí.
“ExceptionListener” na stránce 1951	Aplikace používá modul listener pro výjimky k asynchronnímu upozornění na problém s připojením.
“Výjimka IllegalState” na stránce 1952	XMS vyvolá tuto výjimku, pokud aplikace volá metodu v nesprávném nebo nevhodném čase, nebo pokud XMS není v odpovídajícím stavu pro požadovanou operaci.
“InitialContext” na stránce 1952	Aplikace používá objekt InitialContext k vytvoření objektů z definic objektů, které jsou načteny z úložiště spravovaných objektů.

Tabulka 872. Souhrn rozhraní třídy .NET (pokračování)	
Rozhraní	**Popis**
“Výjimka IDException InvalidClient” na stránce 1954	XMS vyvolá tuto výjimku, pokud se aplikace pokusí nastavit identifikátor klienta pro připojení, ale identifikátor klienta je neplatný nebo je již používán.
“Výjimka InvalidDestination” na stránce 1954	XMS vyvolá tuto výjimku, pokud aplikace určuje, že místo určení není platné.
“Výjimka InvalidSelector” na stránce 1955	XMS vyvolá tuto výjimku, pokud aplikace poskytuje výraz selektoru zpráv, jehož syntaxe není platná.
“IMapMessage” na stránce 1955	Mapová zpráva je zpráva, jejíž tělo tvoří sadu dvojic název-hodnota, kde má každá hodnota přidružený datový typ.
“IMessage” na stránce 1964	Objekt zpráv představuje zprávu, kterou aplikace odesílá nebo přijímá. IMessage je nadtřída pro třídy zpráv, jako např. IMapMessage.
“IMessageConsumer” na stránce 1969	Aplikace používá spotřebitele zpráv k přijetí zpráv odeslaných do místa určení.
“MessageEOFException” na stránce 1972	XMS vyvolá tuto výjimku, pokud XMS narazí na konec proudu bajtů zpráv, když aplikace čte tělo bajtové zprávy.
“Výjimka MessageFormat” na stránce 1972	XMS vyvolá tuto výjimku, pokud XMS narazí na zprávu s formátem, který není platný.
“IMessageListener (delegát)” na stránce 1972	Aplikace používá modul listener pro zprávy k asynchronnímu příjmu zpráv.
“MessageNotReadableException” na stránce 1973	XMS vyvolá tuto výjimku, pokud se aplikace pokusí číst tělo zprávy, která je pouze pro zápis.
“MessageNotWritableException” na stránce 1973	XMS vyvolá tuto výjimku, pokud se aplikace pokusí o zápis do těla zprávy, která je jen pro čtení.
“IMessageProducer” na stránce 1973	Aplikace používá producenta zpráv k odesílání zpráv do místa určení.
“IObjectMessage” na stránce 1979	Zpráva objektu je zpráva, jejíž tělo obsahuje serializovaný objekt Java nebo .NET .
“IPropertyContext” na stránce 1979	IPropertyContext je abstraktní nadtřída, která obsahuje metody pro získání a nastavení vlastností. Tyto metody jsou děděny jinými třídami.
“IQueueBrowser” na stránce 1988	Aplikace používá prohlížeč front k procházení zpráv ve frontě, aniž by je odebíráte.
“Žadatel” na stránce 1990	Aplikace používá žadatele k odeslání zprávy požadavku a následné čekání na odpověď a přijetí odpovědi.
“Výjimka ResourceAllocation” na stránce 1991	XMS vyvolá tuto výjimku, pokud XMS nemůže přidělit prostředky požadované metodou.

Tabulka 872. Souhrn rozhraní třídy .NET (pokračování)	
Rozhraní	**Popis**
"SecurityException" na stránce 1991	XMS vyvolá tuto výjimku, je-li identifikátor uživatele a heslo poskytnuté pro ověření aplikace odmítnuty. XMS také vyvolá tuto výjimku, pokud kontrola oprávnění selže a zabrání dokončení metody.
"ISession" na stránce 1992	Relace je jednovláknový kontext pro odesílání a příjem zpráv.
"IStreamMessage" na stránce 2002	Proudová zpráva je zpráva, jejíž tělo tvoří proud hodnot, kde má každá hodnota přidružený datový typ.
"ITextMessage" na stránce 2010	Textová zpráva je zpráva, jejíž tělo tvoří řetězec.
"TransactionInProgressException" na stránce 2011	XMS vyvolá tuto výjimku, pokud aplikace požádá o operaci, která není platná, protože probíhá transakce.
"TransactionRolledBackException" na stránce 2012	XMS vyvolá tuto výjimku, pokud aplikace volá Session.commit() k potvrzení aktuální transakce, ale transakce se pak vrátí zpět.
XMSC	Pro produkt .NETjsou názvy vlastností a hodnoty produktu XMS definovány v této třídě jako veřejné konstanty. Další podrobnosti viz "Vlastnosti objektů XMS" na stránce 2014.
"Výjimka XMSException" na stránce 2012	Pokud XMS zjistí chybu během zpracování volání metody .NET , příkaz XMS vyvolá výjimku. Výjimkou je objekt, který zapouzdřuje informace o chybě. Existují různé typy výjimek XMS a objekt XMSException je pouze jedním typem výjimky. Třída XMSException je však nadtřídou jiných tříd výjimek XMS . XMS generuje objekt XMSException v situacích, kdy není vhodný žádný z jiných typů výjimek.
"XMSFactoryFactory" na stránce 2013	Pokud aplikace nepoužívá spravované objekty, použijte tuto třídu k vytvoření továren připojení, front a témat.

Definice každé metody uvádí kódy výjimek, které XMS může vrátit, pokud zjistí chybu během zpracování volání metody. Každý kód výjimky je představován svou pojmenovanou konstantou, která má odpovídající výjimku.

IBytesMessage

Bajtová zpráva je zpráva, jejíž tělo tvoří proud bajtů.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IBytesMessage
```

.NET vlastnosti

BodyLength -Získat délku těla

Rozhraní:

```
Int64 BodyLength
    {
      get;
    }
```

Získejte délku těla zprávy v bajtech, je-li tělo zprávy jen pro čtení.

Vrácená hodnota je délka celého těla bez ohledu na to, kde je kurzor pro čtení zprávy momentálně umístěn.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

Metody

ReadBoolean -Čtení hodnoty typu Boolean

Rozhraní:

```
Boolean ReadBoolean();
```

Přečtěte si logickou hodnotu z proudu zpráv v bajtech.

Parametry:
Není

vrátí:
Logická hodnota, která je přečtená.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadSignedByte-Read Byte

Rozhraní:

```
Int16   ReadSignedByte();
```

Přečíst další bajt z proudu zpráv bajtů jako podepsané 8bitové celé číslo.

Parametry:
Není

vrátí:
Bajt, který je čten.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes -Čtení bajtů

Rozhraní:

```
Int32   ReadBytes(Byte[] array);
Int32   ReadBytes(Byte[] array, Int32 length);
```

Čtení pole bajtů z proudu bajtů zpráv, které začíná od aktuální pozice kurzoru.

Parametry:

array (výstup)
Vyrovnávací paměť, která má obsahovat pole bajtů, které se čtou. Pokud je počet bajtů zbývajících k přečtení z proudu před voláním větší nebo roven délce vyrovnávací paměti, vyrovnávací paměť je zaplněna. Jinak bude vyrovnávací paměť částečně vyplněna všemi zbývajícími bajty.

Určíte-li na vstupu ukazatel null, metoda přeskočí bajty, aniž by je četla. Pokud je počet bajtů zbývajících k přečtení z proudu před voláním větší nebo rovnou délce vyrovnávací paměti, počet vynechaných bajtů je roven délce vyrovnávací paměti. Jinak budou všechny zbývající bajty vynechány. Kurzor zůstává na další pozici, aby bylo možné číst v proudu bajtů zpráv.

délka (vstup)
Délka vyrovnávací paměti v bajtech

vrátí:
Počet bajtů, které jsou načteny do vyrovnávací paměti. Je-li vyrovnávací paměť částečně vyplněna, hodnota je menší než délka vyrovnávací paměti, což znamená, že zde již nejsou žádné další bajty, které mají být čteny. Pokud zbývají před voláním žádné bajty, které mají být přečteny z proudu, hodnota je XMSC_END_OF_STREAM.

Uvedete-li na vstupu ukazatel null, metoda nevrátí žádnou hodnotu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

ReadChar -Čtení znaků

Rozhraní:

```
Char    ReadChar();
```

Přečtěte si další 2 bajty z bajtového proudu zpráv jako znak.

Parametry:
Není

vrátí:
Znak, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble -Přečíst číslo s pohyblivou řádovou čárkou s dvojitou přesností

Rozhraní:

```
Double  ReadDouble();
```

Přečtěte si dalších 8 bajtů z proudu zpráv v bajtech jako číslo s pohyblivou řádovou čárkou a dvojitou přesností.

Parametry:
Není

vrátí:
Číslo s plovoucí řádovou čárkou a dvojitou přesností, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat -Čtení čísla s pohyblivou řádovou čárkou

Rozhraní:

```
Single  ReadFloat();
```

Přečtěte si následující 4 bajty z bajtového proudu zpráv jako číslo s pohyblivou řádovou čárkou.

Parametry:
Není

vrátí:
Číslo v plovoucí řádové čárce, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt -Přečíst celé číslo

Rozhraní:

```
Int32   ReadInt();
```

Přečtěte si další 4 bajty z bajtového proudu zpráv jako 32bitové celé číslo se znaménkem.

Parametry:
Není

vrátí:
Celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong -Číst dlouhé celé číslo

Rozhraní:

```
Int64   ReadLong();
```

Přečtěte si dalších 8 bajtů z bajtového proudu zpráv jako podepsané 64bitové celé číslo.

Parametry:
Není

vrátí:
Dlouhé celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadShort -Přečíst krátké celé číslo

Rozhraní:

```
Int16   ReadShort();
```

Přečtěte si další 2 bajty z bajtového proudu zpráv jako 16bitové celé číslo se znaménkem.

Parametry:
Není

vrátí:
Krátké celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte -Čtení nepodepsaných bajtů

Rozhraní:

```
Byte    ReadByte();
```

Přečíst další bajt z proudu bajtů zpráv jako 8bitové celé číslo typu unsigned.

Parametry:
Není

vrátí:
Bajt, který je čten.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadUnsignedShort-Read Unsigned Short Integer

Rozhraní:

```
Int32   ReadUnsignedShort();
```

Přečtěte si další 2 bajty z proudu zpráv o bajtech jako 16bitové celé číslo bez znaménka.

Parametry:
Není

vrátí:
Krátké celé číslo bez znaménka, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadUTF -Čtení řetězce UTF

Rozhraní:

```
String  ReadUTF();
```

Přečíst řetězec, zakódovaný v UTF-8, z bajtového proudu zpráv.

Poznámka: Před voláním funkce ReadUTF() se ujistěte, že kurzor ve vyrovnávací paměti ukazuje na začátek toku bajtů zpráv.

Parametry:
Není

vrátí:
Řetězcový objekt zapouzdřující řetězec, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

Reset-Reset

Rozhraní:

```
void Reset();
```

Umístěte tělo zprávy do režimu jen pro čtení a přesuňte kurzor na začátek bajtového proudu zpráv.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotReadableException

WriteBoolean -Zapsat logickou hodnotu

Rozhraní:

```
void  WriteBoolean(Boolean value);
```

Zapište logickou hodnotu do bajtového proudu zpráv.

Parametry:

hodnota (vstup)
Logická hodnota, která má být zapsána.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteByte -Zápis bajtu

Rozhraní:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Napište bajt do bajtového proudu zpráv.

Parametry:

hodnota (vstup)
Bajt, který se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Zápis bajtů

Rozhraní:

```
void  WriteBytes(Byte[] value);
```

Zapisovat pole bajtů do proudu bajtů zprávy.

Parametry:

hodnota (vstup)
Pole bajtů, které mají být zapsány.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Zápis částečných bajtů do pole

Rozhraní:

```
void  WriteBytes(Byte[] value, int offset, int length);
```

Napište dílčí pole bajtů do proudu zpráv o bajtech, jak je definováno uvedenou délkou.

Parametry:

hodnota (vstup)
Pole bajtů, které mají být zapsány.

offset (vstup)
Počáteční bod pro pole bajtů, které mají být zapsány.

délka (vstup)
Počet bajtů, které se mají zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteChar -Zápis znaku

Rozhraní:

```
void  WriteChar(Char value);
```

Napište znak do toku bajtů zpráv jako 2 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Znak, který má být zapsán.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteDouble -Write Double Precision Floating Point Number

Rozhraní:

```
void  WriteDouble(Double value);
```

Převeďte číslo s pohyblivou řádovou čárkou a dvojitou přesností na dlouhé celé číslo a zapište dlouhé celé číslo na proud bajtů zpráv jako 8 bajtů, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Číslo s plovoucí řádovou čárkou s dvojitou přesností, které se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteFloat -Zápis čísla s pohyblivou řádovou čárkou

Rozhraní:

```
void  WriteFloat(Single value);
```

Převeďte číslo s pohyblivou řádovou čárkou na celé číslo a zapište celé číslo do proudu bajtů zpráv jako 4 bajty, bajt s velkým pořadovým číslem jako první a první.

Parametry:

hodnota (vstup)
Číslo plovoucí řádové čárky, které se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteInt -Zapsat celé číslo

Rozhraní:

```
void  WriteInt(Int32 value);
```

Zapište celé číslo do proudu bajtů zpráv jako 4 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteLong -zapište dlouhé celé číslo

Rozhraní:

```
void  WriteLong(Int64 value);
```

Zapište dlouhé celé číslo na proud bajtů zpráv jako 8 bajtů, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Dlouhé celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteObject -Zapsat objekt

Rozhraní:

```
void  WriteObject(Object value);
```

Napište uvedený objekt do bajtového proudu zpráv.

Parametry:

hodnota (vstup)
Objekt, který má být zapsán, což musí být odkaz na primitivní typ.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteShort -zapište krátké celé číslo

Rozhraní:

```
void  WriteShort(Int16 value);
```

Napište krátké celé číslo do toku bajtů zpráv jako 2 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Krátké celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteUTF -Zápis řetězce UTF

Rozhraní:

```
void  WriteUTF(String value);
```

Napište řetězec do proudu zpráv v bajtech kódovaný v UTF-8.

Parametry:

hodnota (vstup)
Řetězcový objekt zapouzdřující řetězec, který má být zapsán.

vrátí:
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

IConnection

Objekt připojení představuje aktivní připojení aplikace k serveru systému zpráv.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnection
```

For a list of the XMS defined properties of a Connection object, see “Vlastnosti připojení” na stránce 2015.

.NET vlastnosti

ClientID -Získání a nastavení ID klienta

Rozhraní:

```
String ClientID
      {
        get;
        set;
      }
```

Získejte a nastavte identifikátor klienta pro připojení.

Identifikátor klienta může být buď předkonfigurován administrátorem v rámci ConnectionFactory, nebo přiřazen nastavením ClientID.

Identifikátor klienta se používá pouze k podpoře trvalých odběrů v doméně publikování/odběru a je ignorován v dvoubodové doméně.

Pokud aplikace nastavuje identifikátor klienta pro připojení, musí to aplikace provést okamžitě po vytvoření připojení a před provedením jakékoli jiné operace na připojení. Pokud se aplikace pokusí o nastavení identifikátoru klienta po tomto bodu, volání vyvolá výjimku IllegalStateException.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState
- Výjimka IDException InvalidClient

ExceptionListener -Získat a nastavit modul listener pro výjimky

Rozhraní:

```
   ExceptionListener ExceptionListener
       {
         get;
         set;
       }
```

Získejte modul listener pro výjimky, který je registrován u připojení, a zaregistrujte modul listener pro výjimky s připojením.

Není-li modul listener pro výjimky registrován s připojením, metoda vrací hodnotu null. Je-li již modul listener pro výjimky registrován s připojením, můžete zrušit registraci zadáním hodnoty null namísto modulu listener pro výjimky.

Další informace o použití listenerů výjimek naleznete v tématu Použití listenerů zpráv a výjimek 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 uzavřeno, volání se ignoruje.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

CreateSession -Vytvořit relaci

Rozhraní:

```
    ISession CreateSession(Boolean transacted,
                        AcknowledgeMode acknowledgeMode);
```

Vytvořte relaci.

Parametry:

transkovaný (vstup)
Hodnota `True` znamená, že relace je zpracovávána. Hodnota `False` znamená, že relace se neobchoduje.

V případě připojení v reálném čase ke zprostředkovateli musí být hodnota `False`.

acknowledgeMode (vstup)
Indikuje, jak jsou potvrzovány zprávy přijaté aplikací. Hodnota musí být jednou z následujících hodnot z výčtového nástroje AcknowledgeMode :

`AcknowledgeMode.AutoAcknowledge`

`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Pro připojení v reálném čase ke zprostředkovateli musí být tato hodnota `AcknowledgeMode.AutoAcknowledge` nebo `AcknowledgeMode.DupsOkAcknowledge`

Tento parametr je ignorován, pokud relace obsahuje transakci. Další informace o režimech potvrzení naleznete v tématu Potvrzení zprávy.

vrátí:
Objekt Session

Výjimky:

- Výjimka XMSException

Start-Spustit připojení

Rozhraní:

```
void  Start();
```

Spusťte nebo restartujte doručení příchozích zpráv pro připojení. Volání je ignorováno, je-li připojení již spuštěno.

Parametry:
Není

vrátí:
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

vrátí:
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 vlastností definovaných XMS pro objekt ConnectionFactory naleznete v tématu "Vlastnosti objektu ConnectionFactory" na stránce 2016.

Metody

CreateConnection -Vytvoření továrny připojení (s použitím výchozí identity uživatele)

Rozhraní:

```
IConnection CreateConnection();
```

Vytvořte továrnu připojení s výchozími vlastnostmi.

Pokud se připojujete k IBM MQ a není nastaven parametr XMSCS_USERID, pak správce front standardně používá userID přihlášeného uživatele. Požadujete-li další ověření na úrovni připojení pro jednotlivé uživatele, můžete napsat uživatelskou proceduru pro ověření klienta, která je konfigurována 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 s použitím zadané identity uživatele.

Pokud se připojujete k IBM MQ a není nastaven parametr XMSCS_USERID, pak správce front standardně používá userID přihlášeného uživatele. Požadujete-li další ověření na úrovni připojení pro jednotlivé uživatele, můžete napsat uživatelskou proceduru pro ověření klienta, která je konfigurována v produktu IBM MQ.

Připojení se vytvoří v zastaveném režimu. Žádné zprávy se doručují až do volání aplikace **`Connection.start()`**.

Parametry:

userID (vstup)
Řetězcový objekt zapouzdřující identifikátor uživatele, který má být použit pro ověření aplikace. Zadáte-li hodnotu null, dojde k pokusu o vytvoření připojení bez ověření.

password (vstup)
Řetězcový objekt zapouzdřující heslo, které má být použito pro ověření aplikace. Zadáte-li hodnotu null, dojde k pokusu o vytvoření připojení bez ověření.

vrátí:
Objekt připojení.

Výjimky:

- Výjimka XMSException
- VÝJIMKA 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

Data IConnectionMetaData

Objekt datového objektu ConnectionMetaposkytuje informace o připojení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IConnectionMetaData
```

Seznam vlastností definovaných XMS pro objekt dat ConnectionMetanaleznete v tématu “Vlastnosti dat ConnectionMeta” na stránce 2021.

.NET vlastnosti

JMSXPropertyNames -Získat vlastnosti definované zprávy JMS

Rozhraní:

```
System.Collections.IEnumerator JMSXPropertyNames
    {
      get;
    }
```

Vrátí výčet názvů vlastností zpráv definovaných produktem JMS , které jsou podporovány připojením.

JMS vlastností definovaných zpráv není podporováno v reálném čase připojení 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

Instinace

Cíl 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 definovaných vlastností XMS pro objekt Destination lze najít v tématu “Vlastnosti místa určení” na stránce 2022.

.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 k asynchronnímu upozornění na problém s připojením.

Hierarchie dědičnosti:

Není

Pokud aplikace používá připojení pouze k asynchronnímu příjmu zpráv a k žádnému jinému účelu, pak jedinou možností, jak se aplikace může naučit o problému s připojením, je použití modulu listener pro výjimky. V jiných situacích může modul listener pro výjimky poskytnout bezprostřednější způsob, jak se naučit o problému s připojením, než čekat na další synchronní volání na XMS.

Delegát

ExceptionListener -Listener výjimek

Rozhraní:

```
public delegate void ExceptionListener(Exception ex)
```

Oznámit aplikaci problému s připojením.

Metody, které implementují tohoto delegáta, mohou být zaregistrovány spolu s připojením.

Další informace o použití listenerů výjimek naleznete v tématu Použití listenerů zpráv a výjimek v produktu .NET.

Parametry:

exception (vstup)
Ukazatel na výjimku vytvořenou příkazem XMS.

vrátí:
Void

Výjimka IllegalState

XMS vyvolá tuto výjimku, pokud aplikace volá metodu v nesprávném 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, GetLinkedException

InitialContext

Aplikace používá objekt InitialContext k vytvoření objektů z definic objektů, které jsou načteny z úložiště spravovaných objektů.

Hierarchie dědičnosti:
Není

.NET vlastnosti

Prostředí-Získat 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 potřebné k vytvoření připojení k úložišti spravovaných objektů jsou poskytovány konstruktoru v tabulce Hashtable prostředí.

Výjimky:

- Výjimka XMSException

Metody

Prostředí AddTo-přidání nové vlastnosti do prostředí

Rozhraní:

```
Object AddToEnvironment(String propName, Object propVal);
```

Přidejte novou vlastnost do prostředí.

Parametry:

propName (vstup)
Řetězcový objekt zapouzdřující název vlastnosti, která má být přidána.

propVal (vstup)
Hodnota vlastnosti, která má být přidána.

vrátí:
Původní hodnota vlastnosti.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Zavřít-Zavřít tento kontext

Rozhraní:

```
void Close()
```

Zavřete tento kontext.

Parametry:
Není

vrátí:
Není

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Vyhledat-Vyhledat objekt v počátečním kontextu

Rozhraní:

```
Object Lookup(String name);
```

Vytvoření objektu z definice objektu, která se načítá z úložiště spravovaných objektů.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název administrovaného objektu, který má být načten. Název může být buď jednoduchý název, nebo komplexní název. Další podrobnosti naleznete v tématu Načítání spravovaných objektů.

vrátí:
Buďto objekt IConnectionFactory , nebo hodnota IDestination, v závislosti na typu načítaného objektu. Pokud má funkce přístup k adresáři, ale nemůže najít požadovaný objekt, je vrácena hodnota null.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

RemoveFromEnvironment-Odebrat vlastnost z prostředí

Rozhraní:

```
Object RemoveFromEnvironment(String propName);
```

Odeberte vlastnost z prostředí.

Parametry:

propName (vstup)
Řetězcový objekt zapouzdřující název vlastnosti, která má být odebrána.

vrátí:
Objekt, který byl odebrán.

Výjimky:

- Výjimky jsou specifické pro používanou adresářovou službu.

Výjimka IDException InvalidClient

XMS vyvolá tuto výjimku, pokud se aplikace pokusí nastavit identifikátor klienta pro připojení, ale identifikátor klienta je neplatný nebo je již používán.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidClientIDException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedException

Výjimka InvalidDestination

XMS vyvolá tuto výjimku, pokud aplikace určuje, že místo určení 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, GetLinkedException

Výjimka InvalidSelector

XMS vyvolá tuto výjimku, pokud aplikace poskytuje výraz selektoru zpráv, jehož syntaxe není platná.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.InvalidSelectorException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedException

IMapMessage

Mapová zpráva je zpráva, jejíž tělo tvoří sadu dvojic název-hodnota, kde má každá hodnota přidružený datový typ.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IMapMessage
```

Když aplikace získá hodnotu páru název-hodnota, lze hodnotu převést pomocí XMS na jiný datový typ. Další informace o tomto formuláři implicitní konverze naleznete v informacích o mapách zpráv v části Tělo zprávy XMS.

.NET vlastnosti

MapNames -Získat názvy map

Rozhraní:

```
System.Collections.IEnumerator MapNames
{
  get;
}
```

Získejte výčet názvů v těle zprávy mapování.

Výjimky:

- Výjimka XMSException

Metody

GetBoolean -Získat logickou hodnotu

Rozhraní:

```
Boolean GetBoolean(String name);
```

Získejte logickou hodnotu identifikovanou názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje logickou hodnotu.

vrátí:
Logická hodnota načtená z těla zprávy mapování.

Výjimky:

- Výjimka XMSException

GetByte -Získat bajt

Rozhraní:

```
Byte    GetByte(String name);
    Int16   GetSignedByte(String name);
```

Získejte bajt identifikovaný názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje bajt.

vrátí:
Bajt načtený z těla zprávy mapy. Na bajtu se neprovádí žádná konverze dat.

Výjimky:

- Výjimka XMSException

GetBytes -získání bajtů

Rozhraní:

```
Byte[]  GetBytes(String name);
```

Získejte pole bajtů identifikovaných názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje pole bajtů.

vrátí:
Počet bajtů v poli.

Výjimky:

- Výjimka XMSException

GetChar -Získání znaku

Rozhraní:

```
Char    GetChar(String name);
```

Získejte znak identifikovaný názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje znak.

vrátí:
Znak načtený z těla zprávy mapy.

Výjimky:

- Výjimka XMSException

GetDouble -Získat číslo s pohyblivou řádovou čárkou dvojité přesnosti

Rozhraní:

```
Double  GetDouble(String name);
```

Získejte číslo s plovoucí řádovou čárkou a dvojitou přesnost, které jsou identifikovány názvem z textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje číslo s plovoucí řádovou čárkou a dvojitou přesností.

vrátí:
Číslo s plovoucí řádovou čárkou a 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 textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje číslo s pohyblivou řádovou čárkou.

vrátí:
Číslo s plovoucí řádovou čárkou načtené z textu zprávy mapy.

Výjimky:

- Výjimka XMSException

GetInt -Získat celé číslo

Rozhraní:

```
Int32   GetInt(String name);
```

Získejte celé číslo identifikované názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje celé číslo.

vrátí:
Celé číslo načtené z těla zprávy mapování.

Výjimky:

- Výjimka XMSException

GetLong -Získat dlouhé celé číslo

Rozhraní:

```
Int64   GetLong(String name);
```

Získejte dlouhé celé číslo identifikované názvem z textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje dlouhé celé číslo.

vrátí:
Dlouhé celé číslo načtené z těla zprávy mapování.

Výjimky:

- Výjimka XMSException

GetObject -Získat objekt

Rozhraní:

```
Object  GetObject(String name);
```

Získejte odkaz na hodnotu páru název-hodnota z textu zprávy mapy. Dvojice název-hodnota je identifikována názvem.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název dvojice název-hodnota.

vrátí:
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 textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje krátké celé číslo.

vrátí:
Krátké celé číslo načtené z těla zprávy mapování.

Výjimky:

- Výjimka XMSException

GetString -Získat řetězec

Rozhraní:

```
String  GetString(String name);
```

Získejte řetězec identifikovaný názvem z těla zprávy mapování.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje řetězec v těle zprávy mapování.

vrátí:
Řetězcový objekt zapouzdřující řetězec načtený z textu zprávy mapy. Je-li vyžadována konverze dat, bude tato hodnota řetězcem po převodu.

Výjimky:

- Výjimka XMSException

ItemExists -Kontrola názvu-dvojice hodnot existuje.

Rozhraní:

```
Boolean ItemExists(String name);
```

Zkontrolujte, zda tělo zprávy mapování obsahuje dvojici název-hodnota s určeným názvem.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název dvojice název-hodnota.

vrátí:

- `True`, pokud text zprávy mapy obsahuje dvojici název-hodnota s uvedeným názvem.
- `False`, pokud tělo zprávy mapování 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 mapy zpráv.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje logickou hodnotu v těle zprávy mapování.

hodnota (vstup)
Logická hodnota, která má být nastavena.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetByte -Nastavit bajt

Rozhraní:

```
void  SetByte(String name, Byte value);
void  SetSignedByte(String name, Int16 value);
```

Nastavte bajt v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název pro identifikaci bajtu v těle zprávy mapování.

hodnota (vstup)
Bajt, který má být nastaven.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetBytes -Nastavit bajty

Rozhraní:

```
void  SetBytes(String name, Byte[] value);
```

Nastavení pole bajtů v těle mapy zpráv.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, aby identifikoval pole bajtů v těle zprávy mapování.

hodnota (vstup)
Pole bajtů, které mají být nastaveny.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetChar -Nastavit znak

Rozhraní:

```
void  SetChar(String name, Char value);
```

Nastavení 2-bajtového znaku v těle zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující jméno, aby identifikoval znak v textu zprávy mapy.

hodnota (vstup)
Znak, který má být nastaven.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetDouble -Nastavit číslo s pohyblivou řádovou čárkou dvojité přesnosti

Rozhraní:

```
void  SetDouble(String name, Double value);
```

Nastavte číslo s plovoucí řádovou čárkou a dvojitou přesností v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název pro identifikaci čísla s plovoucí řádovou čárkou a dvojitou přesností v textu zprávy mapy.

hodnota (vstup)
Je třeba nastavit číslo s plovoucí řádovou čárkou a dvojitou přesností.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetFloat -Nastavit počet čísel s pohyblivou řádovou čárkou

Rozhraní:

```
void  SetFloat(String name, Single value);
```

Nastavte číslo s pohyblivou řádovou čárkou v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje číslo plovoucí řádové čárky v těle zprávy mapování.

hodnota (vstup)
Číslo plovoucí řádové čárky, které má být nastaveno.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetInt -Nastavit celé číslo

Rozhraní:

```
void  SetInt(String name, Int32 value);
```

Nastavit celé číslo v těle mapy zpráv.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje celé číslo v těle zprávy mapování.

hodnota (vstup)
Celé číslo, které má být nastaveno.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetLong -Nastavit dlouhé celé číslo

Rozhraní:

```
void  SetLong(String name, Int64 value);
```

Nastavte dlouhé celé číslo v těle mapy zpráv.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, aby identifikoval dlouhé celé číslo v těle zprávy mapování.

hodnota (vstup)
Dlouhé celé číslo, které má být nastaveno.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetObject -Nastavit objekt

Rozhraní:

```
void  SetObject(String name, Object value);
```

Nastavte hodnotu, která musí být primitivním typem XMS , v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující jméno, aby identifikoval hodnotu v textu zprávy mapy.

hodnota (vstup)
Pole bajtů obsahující hodnotu, která má být nastavena.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetShort -Nastavit krátké celé číslo

Rozhraní:

```
void  SetShort(String name, Int16 value);
```

Nastavte krátké celé číslo v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, aby identifikoval krátké celé číslo v těle zprávy mapování.

hodnota (vstup)
Krátké celé číslo, které má být nastaveno.

vrátí:
Void

Výjimky:

- Výjimka XMSException

SetString -Nastavit řetězec

Rozhraní:

```
void  SetString(String name, String value);
```

Nastavte řetězec v textu zprávy mapy.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název, aby identifikoval řetězec v těle zprávy mapování.

hodnota (vstup)
Řetězcový objekt zapouzdřující řetězec, který má být nastaven.

vrátí:
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

IMessage

Objekt zpráv představuje zprávu, kterou aplikace odesílá nebo přijímá. IMessage je nadtřída pro třídy zpráv, jako např. IMapMessage.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
```

Seznam polí záhlaví zprávy produktu JMS v objektu zprávy naleznete v tématu Pole záhlaví zprávy XMS. Seznam definovaných vlastností objektu Message (JMS) naleznete v tématu Vlastnosti zprávy definované službou JMS. Seznam definovaných vlastností objektu Message Object IBM najdete v tématu IBMdefinované vlastnosti zprávy. Seznam vlastností JMS_IBM_MQMD* pro objekt Message Object naleznete v tématu “Vlastnosti JMS_IBM_MQMD*” na stránce 2025 .

Zprávy jsou odstraněny programem pro uvolnění paměti. Když je zpráva odstraněna, uvolní prostředky, které použil.

.NET vlastnosti

ID GetJMSCorrelation-Získat a nastavit JMSCorrelationID

Rozhraní:

```
String JMSCorrelationID
{
  get;
  set;
}
```

Získejte a nastavte identifikátor korelace zprávy jako objekt typu String.

Výjimky:

- Výjimka XMSException

JMSDeliveryMode -Získat a nastavit JMSDeliveryMode

Rozhraní:

```
DeliveryMode JMSDeliveryMode
{
  get;
  set;
}
```

Získejte a nastavte režim doručení zprávy.

Režim doručení zprávy je jedna z následujících hodnot:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

V případě nově vytvořené zprávy, která nebyla odeslána, je režim doručení `DeliveryMode.Trvalý`, s výjimkou připojení v reálném čase k zprostředkovateli, pro který je režim doručení `DeliveryMode.NonPersistent`. Pro přijatou zprávu metoda vrací režim doručení, který byl nastaven voláním IMessageProducer.send (), když byla zpráva odeslána, pokud přijímající aplikace nezmění režim doručení nastavením JMSDeliveryMode.

Výjimky:

- Výjimka XMSException

JMSDestination-Získání a nastavení JMSDestination

Rozhraní:

```
IDestination JMSDestination
{
  get;
  set;
}
```

Získejte a nastavte cíl zprávy.

Místo určení je nastaveno voláním IMessageProducer.send (), když je zpráva odeslána. Hodnota JMSDestination je ignorována. Místo určení JMSDestination však můžete použít ke změně místa určení přijaté zprávy.

Pro nově vytvořenou zprávu, která nebyla odeslána, vrací metoda objekt místa určení s hodnotou Null, pokud odesílající aplikace nenastavuje místo určení nastavením JMSDestination. Pro přijatou zprávu metoda vrací objekt Destination pro cíl, který byl nastaven voláním funkce IMessageProducer.send (), když byla zpráva odeslána, pokud přijímající aplikace nezmění cíl nastavením JMSDestination.

Výjimky:

- Výjimka XMSException

JMSExpiration-Získat a nastavit JMSExpiration

Rozhraní:

```
Int64 JMSExpiration
{
  get;
  set;
}
```

Získejte a nastavte čas vypršení platnosti zprávy.

Doba vypršení platnosti je nastavena voláním funkce IMessageProducer.send () při odeslání zprávy. Jeho hodnota se vypočítá tak, že se přidá čas k životu, jak je uvedeno odesílající aplikací, do času odeslání zprávy. Čas vypršení platnosti je vyjádřen v milisekundách od 00:00:00 GMT k 1. lednu 1970.

Pro nově vytvořenou zprávu, která nebyla odeslána, je doba vypršení platnosti 0, pokud odesílající aplikace nenastavuje jinou dobu vypršení platnosti nastavením JMSExpiration. Pro přijatou zprávu metoda vrátí čas vypršení platnosti, který byl nastaven voláním funkce IMessageProducer.send (), když byla zpráva odeslána, pokud přijímající aplikace nezmění čas vypršení platnosti nastavením JMSExpiration.

Je-li doba života rovna 0, volání IMessageProducer.send () nastaví čas vypršení platnosti na 0, aby označoval, že zpráva nevyprší.

Produkt XMS vyřadí zprávy s vypršenou platností a nedoručuje je do aplikací.

Výjimky:

- Výjimka XMSException

JMSMessageID -Získat a nastavit JMSMessageID

Rozhraní:

```
String JMSMessageID
{
  get;
  set;
}
```

Získejte a nastavte identifikátor zprávy jako řetězcový objekt zapouzdřující identifikátor zprávy.

Identifikátor zprávy je nastaven voláním funkce IMessageProducer.send (), když je zpráva odeslána. Pro přijatou zprávu vrací metoda identifikátor zprávy, který byl nastaven voláním IMessageProducer.send (), když byla zpráva odeslána, 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;
}
```

Získejte a nastavte prioritu zprávy.

Priorita je nastavena voláním funkce IMessageProducer.send (), když je zpráva odeslána. Hodnota je celé číslo v rozsahu 0, nejnižší priorita, k 9, nejvyšší priorita.

Pro nově vytvořenou zprávu, která nebyla odeslána, je priorita 4 , pokud odesílající aplikace nenastaví jinou prioritu nastavením parametru JMSPriority. Pro přijatou zprávu metoda vrací prioritu, která byla nastavena voláním funkce IMessageProducer.send (), když byla zpráva odeslána, pokud přijímající aplikace nezmění prioritu nastavením parametru JMSPriority.

Výjimky:

- Výjimka XMSException

JMSRedelivered-Get a Set JMSRedelivered

Rozhraní:

```
Boolean JMSRedelivered
{
  get;
  set;
}
```

Získejte informace o tom, zda je zpráva znovu doručena, a uveďte, zda je zpráva znovu doručena. Indikace je nastavena voláním IMessageConsumer.receive (), když je přijata zpráva.

Tato vlastnost má následující hodnoty:

- `True`, pokud je zpráva znovu doručena.
- `False`, pokud zpráva nebyla znovu doručena.

Pro připojení v reálném čase ke zprostředkovateli je hodnota vždy `False`.

Indikace opětovného doručení sady JMSRedelivered před odesláním zprávy je ignorována voláním funkce IMessageProducer.send () při odeslání zprávy a je ignorována a nahrazena voláním IMessageConsumer.receive (), když je zpráva přijata. Můžete však použít JMSReverse, abyste změnili indikaci 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 cílovým objektem pro místo určení, kam má být odeslána odpověď na zprávu. Cílový objekt s hodnotou Null znamená, že se neočekává žádná odpověď.

Výjimky:

- Výjimka XMSException

JMSTimstamp-Získat a nastavit JMSTimstamp

Rozhraní:

```
Int64 JMSTimestamp
{
  get;
  set;
}
```

Získejte a nastavte čas, kdy byla zpráva odeslána.

Časové razítko nastavuje volání IMessageProducer.send (), když je zpráva odeslána a je vyjádřena v milisekundách od 00:00:00 GMT k 1. lednu 1970.

Pro nově vytvořenou zprávu, která nebyla odeslána, je časové razítko 0, pokud odesílající aplikace nenastavuje jiné časové razítko nastavením JMSTimstamp. Pro přijatou zprávu vrací metoda časové razítko, které bylo nastaveno voláním IMessageProducer.send (), když byla zpráva odeslána, pokud přijímající aplikace nemění časové razítko nastavením JMSTimstamp.

Výjimky:

- Výjimka XMSException

Notes:

1. Je-li časová značka nedefinovaná, 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 požadována konverze dat, je tato hodnota typ po převodu.

Výjimky:

- Výjimka XMSException

PropertyNames -Získat vlastnosti

Rozhraní:

```
System.Collections.IEnumerator PropertyNames
{
  get;
}
```

Získejte výčet názvů vlastností pro zprávu.

Výjimky:

- Výjimka XMSException

Metody

Potvrdit-Potvrdit

Rozhraní:

```
void  Acknowledge();
```

Potvrdit tuto zprávu a všechny dříve nepotvrzené zprávy přijaté relací.

Aplikace může volat tuto metodu, je-li režim potvrzení relace AcknowledgeMode.ClientAcknowledge. Volání metody jsou ignorována, pokud má relace nějaký jiný režim potvrzení nebo se jedná o transakci.

Zprávy, které byly přijaty, ale nebyly potvrzeny, mohou být znovu doručeny.

Další informace o potvrzení zpráv viz ../com.ibm.mq.dev.doc/xms_cmesack.dita#xms_cmesack.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

ClearBody -Vymazat tělo

Rozhraní:

```
void  ClearBody();
```

Vyčistěte 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 najdete v tématu Tělo zprávy XMS.

Aplikace může kdykoli vymazat tělo zprávy, bez ohledu na to, v jakém stavu se tělo nachází. Je-li tělo zprávy jen pro čtení, jediný způsob, jak může aplikace zapisovat do těla, je aplikace, aby bylo tělo první.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

ClearProperties -Vymazat vlastnosti

Rozhraní:

```
void  ClearProperties();
```

Vymažte vlastnosti zprávy. Pole záhlaví a tělo zprávy nejsou vymazány.

Pokud aplikace vymaže vlastnosti zprávy, vlastnosti budou přístupné pro čtení a zápis.

Aplikace může kdykoli vymazat vlastnosti zprávy, bez ohledu na to, ve kterém stavu jsou vlastnosti uvedeny. Jsou-li vlastnosti zprávy určeny pouze ke čtení, lze do aplikace zapisovat pouze vlastnosti, aby bylo možné vlastnosti nejprve vymazat.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

PropertyExists -Kontrola vlastnosti existuje

Rozhraní:

```
Boolean PropertyExists(String propertyName);
```

Zkontrolujte, zda má zpráva vlastnost s určeným názvem.

Parametry:

propertyName (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:

- `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 přijetí zpráv odeslaných do místa určení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageConsumer
```

Seznam definovaných vlastností XMS objektu MessageConsumer naleznete v tématu “Vlastnosti objektu MessageConsumer” na stránce 2029.

.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 zaregistrujte modul listener zpráv se spotřebitelem zpráv.

Není-li u spotřebitele zpráv zaregistrován žádný modul listener pro zprávy, parametr MessageListener má hodnotu Null. Pokud je modul listener pro zprávy již registrován se spotřebitelem zpráv, můžete registraci zrušit tím, že místo toho uvedete hodnotu null.

Další informace o použití listenerů zpráv naleznete v tématu Použití listenerů zpráv a výjimek v .NET.

Výjimky:

- Výjimka XMSException

MessageSelector -Získat selektor zpráv

Rozhraní:

```
String MessageSelector
{
  get;
}
```

Získejte selektor zpráv pro spotřebitele zpráv. Návratová hodnota je řetězcový objekt zapouzdřující výraz selektoru zpráv. Je-li požadována konverze 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 zavřen, volání se ignoruje.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

Přijmout-Přijmout

Rozhraní:

```
IMessage Receive();
```

Přijmout další zprávu pro spotřebitele zpráv. Volání čeká po neomezenou dobu na zprávu, nebo dokud nebude spotřebitel zpráv uzavřen.

Parametry:
Není

vrátí:
Ukazatel na objekt zprávy. Je-li spotřebitel zpráv zavřen, zatímco volání čeká na zprávu, metoda vrátí ukazatel na objekt Message Object s hodnotou Null.

Výjimky:

- Výjimka XMSException

Příjem-příjem (s intervalem čekání)

Rozhraní:

```
IMessage Receive(Int64 delay);
```

Přijmout další zprávu pro spotřebitele zpráv. Volání čeká pouze určené období pro zprávu, nebo dokud není spotřebitel zpráv uzavřen.

Parametry:

delay (vstup)
Doba (v milisekundách), po kterou volání čeká na zprávu. Uvedete-li interval čekání 0, volání čeká nekonečně dlouhou dobu na zprávu.

vrátí:
Ukazatel na objekt zprávy. Pokud během intervalu čekání nepřijde žádná zpráva, nebo pokud je spotřebitel zpráv zavřen, zatímco volání čeká na zprávu, metoda vrátí ukazatel na objekt Message Object s hodnotou null, ale nevyvolá žádnou výjimku.

Výjimky:

- Výjimka XMSException

ReceiveNoČekání-Přijmout s nečekanou dobou čekání

Rozhraní:

```
IMessage ReceiveNoWait();
```

Přijmout další zprávu pro spotřebitele zpráv, je-li k dispozici okamžitě.

Parametry:
Není

vrátí:
Ukazatel na objekt zprávy. Není-li okamžitě k dispozici žádná zpráva, metoda vrátí ukazatel na objekt Message Object 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 narazí na konec proudu bajtů 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, GetLinkedException

Výjimka MessageFormat

XMS vyvolá tuto výjimku, pokud XMS narazí na zprávu s formátem, který není platný.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.MessageFormatException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedException

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 zaregistrovány spolu s připojením.

Další informace o použití listenerů zpráv naleznete v tématu Použití listenerů zpráv a výjimek v produktu .NET.

Parametry:

mesg (vstup)
Objekt Zpráva.

vrátí:
Void

MessageNotReadableException

XMS vyvolá tuto výjimku, pokud se aplikace pokusí číst tělo zprávy, která 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, GetLinkedException

MessageNotWritableException

XMS vyvolá tuto výjimku, pokud se aplikace pokusí o zápis 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, GetLinkedException

IMessageProducer

Aplikace používá producenta zpráv k odesílání zpráv do místa určení.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessageProducer
```

Seznam definovaných vlastností XMS objektu MessageProducer viz “Vlastnosti objektu MessageProducer” na stránce 2029.

.NET vlastnosti

DeliveryMode -Získat a nastavit výchozí režim doručení

Rozhraní:

```
DeliveryMode DeliveryMode
{
  get;
  set;
}
```

Získejte a nastavte výchozí režim doručení pro zprávy odeslané producentem zpráv.

Výchozí režim doručení má jednu z následujících hodnot:

`DeliveryMode.Persistent`
`DeliveryMode.NonPersistent`

V případě připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

Výchozí hodnota je `DeliveryMode.Persistent`, kromě připojení v reálném čase pro zprostředkovatele, jehož výchozí hodnota je `DeliveryMode.NonPersistent`.

Výjimky:

- Výjimka XMSException

Cíl-Získat cíl

Rozhraní:

```
IDestination Destination
{
  get;
}
```

Získejte místo určení pro producenta zpráv.

Parametry:
Není

vrátí:
Cílový objekt. Pokud producent zpráv nemá místo určení, metoda vrátí objekt místa určení s hodnotou Null.

Výjimky:

- Výjimka XMSException

ID DisableMsg-Získání a nastavení příznaku Zakázat ID zprávy

Rozhraní:

```
Boolean DisableMessageID
{
  get;
  set;
}
```

Získejte informace o tom, zda přijímající aplikace vyžaduje zahrnutí identifikátorů zpráv do zpráv odeslaných producentem zpráv, a indikuje, zda přijímající aplikace vyžaduje zahrnutí identifikátorů zpráv do zpráv odeslaných producentem zpráv.

V případě připojení ke správci front nebo v reálném čase připojení k zprostředkovateli je tento příznak ignorován. Při připojení ke sběrnici pro integraci služeb je tento parametr dodržován.

ID DisabledMsgmá následující hodnoty:

- `True`, pokud přijímající aplikace nevyžaduje zahrnutí identifikátorů zpráv do zpráv odeslaných producentem zpráv.
- `False`, pokud přijímající aplikace vyžaduje, aby byly do zpráv odeslaných producentem zpráv zahrnuty identifikátory zpráv.

Výjimky:

- Výjimka XMSException

DisableMsgTS-Získat a nastavit příznak zablokování časového razítka

Rozhraní:

```
Boolean DisableMessageTimestamp
{
  get;
  set;
}
```

Získejte informace o tom, zda přijímající aplikace vyžaduje zahrnutí časových razítek do zpráv odeslaných producentem zpráv, a indikuje, zda přijímající aplikace vyžaduje zahrnutí časových razítek do zpráv odeslaných producentem zpráv.

V reálném čase připojení k zprostředkovateli je tento příznak ignorován. V případě připojení ke správci front nebo připojení ke sběrnici pro integraci služeb je tento příznak uznán.

DisableMsgTS má následující hodnoty:

- `True`, pokud přijímající aplikace nevyžaduje, aby byla časová razítka zahrnuta do zpráv odeslaných producentem zpráv.
- `False`, pokud přijímající aplikace vyžaduje, aby byla časová razítka zahrnuta do zpráv odeslaných producentem zpráv.

vrátí:

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 zpráv je celé číslo v rozsahu `0`, nejnižší priorita, až `9`, nejvyšší priorita.

V reálném čase připojení k zprostředkovateli je priorita zprávy ignorována.

Výjimky:

- Výjimka XMSException

TimeToLive-Získat a nastavit výchozí dobu na život

Rozhraní:

```
Int64 TimeToLive
{
  get;
```

```
  set;
}
```

Získejte a nastavte výchozí dobu, po kterou existuje zpráva, než vyprší platnost.

Čas je měřen od doby, kdy producent zpráv odešle zprávu a je výchozím nastavením času života v milisekundách. Hodnota 0 znamená, že platnost zprávy nikdy nevyprší.

Pro připojení v reálném čase ke zprostředkovateli je tato hodnota vždy 0.

Výjimky:

- Výjimka XMSException

Metody

Zavřít-Producent zprávy o zavření

Rozhraní:

```
void  Close();
```

Zavřete výrobce zpráv.

Pokud se aplikace pokusí zavřít producenta zpráv, který je již zavřen, volání se ignoruje.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

Odeslat-Odeslat

Rozhraní:

```
void Send(IMessage msg) ;
```

Odešle zprávu na místo určení, které bylo určeno při vytvoření producenta zprávy. Odeslat zprávu s použitím výchozího režimu doručení producenta zpráv, priority a času života.

Parametry:

msg (vstup)
Objekt Zpráva.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination

Odeslat-Odeslat (určení režimu doručení, priority a doby životnosti)

Rozhraní:

```
void Send(IMessage msg,
          DeliveryMode deliveryMode,
```

```
          Int32 priority,
          Int64 timeToLive);
```

Odešle zprávu na místo určení, které bylo určeno při vytvoření producenta zprávy. Odeslat zprávu s použitím určeného režimu doručení, priority a času k životu.

Parametry:

msg (vstup)
Objekt Zpráva.

deliveryMode (vstup)
Režim doručení pro zprávu, který musí mít jednu z následujících hodnot:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

V případě připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

priority (vstup)
Priorita zprávy. Hodnota může být celé číslo v rozsahu `0`, pro nejnižší prioritu, do 9, pro nejvyšší prioritu. V reálném čase připojení k zprostředkovateli je tato hodnota ignorována.

timeToLive (vstup)
Doba životnosti zprávy v milisekundách. Hodnota 0 znamená, že platnost zprávy nikdy nevyprší. V případě připojení v reálném čase ke zprostředkovateli musí být hodnota 0.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat
- Výjimka InvalidDestination
- Výjimka IllegalState

Odeslat-Odeslat (na určené místo určení)

Rozhraní:

```
void Send(IDestination dest, IMessage msg) ;
```

Odešlete zprávu do zadaného cíle, pokud používáte producenta zpráv, pro který nebylo určeno místo určení, když byl vytvořen producent zpráv. Odeslat zprávu s použitím výchozího režimu doručení producenta zpráv, priority a času života.

Obvykle určujete cíl při vytváření producenta zpráv, ale pokud ne, musíte zadat cíl pokaždé, když odešlete zprávu.

Parametry:

dest (vstup)
Cílový objekt.

msg (vstup)
Objekt Zpráva.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka MessageFormat

- Výjimka InvalidDestination

Odeslat-Odeslat (na určené místo určení, určení režimu doručení, priority a času životnosti).

Rozhraní:

```
void Send(IDestination dest,
          IMessage msg,
          DeliveryMode deliveryMode,
          Int32 priority,
          Int64 timeToLive) ;
```

Odešlete zprávu do zadaného cíle, pokud používáte producenta zpráv, pro který nebylo určeno místo určení, když byl vytvořen producent zpráv. Odeslat zprávu s použitím určeného režimu doručení, priority a času k životu.

Obvykle určujete cíl při vytváření producenta zpráv, ale pokud ne, musíte zadat cíl pokaždé, když odešlete zprávu.

Parametry:

dest (vstup)
Cílový objekt.

msg (vstup)
Objekt Zpráva.

deliveryMode (vstup)
Režim doručení pro zprávu, který musí mít jednu z následujících hodnot:

```
DeliveryMode.Persistent
DeliveryMode.NonPersistent
```

V případě připojení v reálném čase ke zprostředkovateli musí být hodnota `DeliveryMode.NonPersistent`.

priority (vstup)
Priorita zprávy. Hodnota může být celé číslo v rozsahu 0, pro nejnižší prioritu, do 9, pro nejvyšší prioritu. V reálném čase připojení k zprostředkovateli je tato hodnota ignorována.

timeToLive (vstup)
Doba životnosti zprávy v milisekundách. Hodnota 0 znamená, že platnost zprávy nikdy nevyprší. V případě připojení v reálném čase ke zprostředkovateli musí být hodnota 0.

vrátí:
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 obsahuje serializovaný objekt 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();
```

Získejte a nastavte 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 nadtřída, která obsahuje metody pro získání a nastavení vlastností. Tyto metody jsou děděny jinými třídami.

Hierarchie dědičnosti:

Není

Metody

Vlastnost GetBoolean-Získat logickou vlastnost

Rozhraní:

```
Boolean GetBooleanProperty(String property_name);
```

Získejte hodnotu logické vlastnosti s určeným názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetByte-Získat bajtovou vlastnost

Rozhraní:

```
Byte    GetByteProperty(String property_name) ;
Int16   GetSignedByteProperty(String property_name) ;
```

Získejte hodnotu vlastnosti byte identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetBytesProperty-Get Byte Array Property

Rozhraní:

```
Byte[]  GetBytesProperty(String property_name) ;
```

Získejte hodnotu vlastnosti pole bajtů identifikované pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Počet bajtů v poli.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetChar-získání vlastnosti znaku

Rozhraní:

```
Char    GetCharProperty(String property_name) ;
```

Získejte hodnotu dvoubajtové znakové vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetDouble-získání vlastnosti Double Precision Floating Point

Rozhraní:

```
Double  GetDoubleProperty(String property_name) ;
```

Získejte hodnotu vlastnosti s pohyblivou řádovou čárkou s dvojitou přesností určenou názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetFloat-získání vlastnosti s pohyblivou řádovou čárkou

Rozhraní:

```
Single  GetFloatProperty(String property_name) ;
```

Získejte hodnotu vlastnosti s pohyblivou řádovou čárkou identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetInt-vlastnost GetInt

Rozhraní:

```
Int32   GetIntProperty(String property_name) ;
```

Získejte hodnotu celočíselné vlastnosti identifikované pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetLong-Získat vlastnost long integer

Rozhraní:

```
Int64   GetLongProperty(String property_name) ;
```

Získejte hodnotu vlastnosti typu long integer identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
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 pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
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

Vlastnost GetShort-získání vlastnosti krátkých celých čísel

Rozhraní:

```
Int16   GetShortProperty(String property_name) ;
```

Získejte hodnotu vlastnosti krátkého celého čísla identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Hodnota vlastnosti.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost GetString-vlastnost GetString

Rozhraní:

```
String  GetStringProperty(String property_name) ;
```

Získejte hodnotu řetězcové vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

vrátí:
Řetězcový objekt zapouzdřující řetězec, který je hodnotou vlastnosti. Je-li vyžadována konverze dat, bude tato hodnota řetězcem po převodu.

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException

Vlastnost SetBoolean-Nastavit logickou vlastnost

Rozhraní:

```
void SetBooleanProperty( String property_name,  Boolean value) ;
```

Nastavte hodnotu logické vlastnosti identifikované pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetByte-Nastavit vlastnost Byte

Rozhraní:

```
void SetByteProperty( String property_name,  Byte value) ;
      void SetSignedByteProperty( String property_name,  Int16 value) ;
```

Nastavte hodnotu vlastnosti byte identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

SetBytesVlastnost-Nastavit vlastnost bajtového pole

Rozhraní:

```
void SetBytesProperty( String property_name, Byte[] value ) ;
```

Nastavte hodnotu vlastnosti bajtového pole identifikovaného pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti, která je polem bajtů.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetChar-Nastavit vlastnost znaku

Rozhraní:

```
void SetCharProperty( String property_name,  Char value) ;
```

Nastavte hodnotu dvoubajtové znakové vlastnosti identifikované názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetDouble-Set Double Precision Floating Point Property

Rozhraní:

```
void SetDoubleProperty( String property_name,  Double value) ;
```

Nastavte hodnotu vlastnosti s plovoucí řádovou čárkou a dvojitou přesností určenou názvem.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetFloat-nastavení vlastnosti s pohyblivou řádovou čárkou

Rozhraní:

```
void SetFloatProperty( String property_name,  Single value) ;
```

Nastavte hodnotu vlastnosti s plovoucí desetinnou čárkou určenou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
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 identifikované pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetLong-Nastavit vlastnost long integer

Rozhraní:

```
void SetLongProperty( String property_name,  Int64 value) ;
```

Nastavte hodnotu vlastnosti typu long integer identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost objektu SetObject-Nastavit vlastnost objektu

Rozhraní:

```
void SetObjectProperty( String property_name,  Object value) ;
```

Nastavte hodnotu a datový typ vlastnosti identifikované pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

objectType (vstup)
Hodnota vlastnosti, která musí být jedna 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.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetShort-Nastavit krátký celočíselný majetek

Rozhraní:

```
void SetShortProperty( String property_name,  Int16 value) ;
```

Nastavte hodnotu vlastnosti krátkého celého čísla identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Hodnota vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

Vlastnost SetString-Nastavit vlastnost řetězce

Rozhraní:

```
void SetStringProperty( String property_name,  String value);
```

Nastavte hodnotu vlastnosti řetězce identifikovanou pomocí názvu.

Parametry:

název_vlastnosti (vstup)
Řetězcový objekt zapouzdřující název vlastnosti.

hodnota (vstup)
Řetězcový objekt zapouzdřující řetězec, který je hodnotou vlastnosti.

vrátí:
Void

Kontext podprocesu:
Určeno podtřídou

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

IQueueBrowser

Aplikace používá prohlížeč front k procházení zpráv ve frontě, aniž by je odebíráte.

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 řetězcový objekt zapouzdřující výraz selektoru zpráv. Je-li požadována konverze dat, tato hodnota je výraz selektoru zpráv po převodu. Pokud prohlížeč front nemá selektor zpráv, metoda vrátí řetězcový objekt s hodnotou null.

Výjimky:

- Výjimka XMSException

Fronta-Získat frontu

Rozhraní:

```
IDestination Queue
{
  get;
}
```

Získejte frontu přidruženou k prohlížeči fronty jako cílový objekt reprezentující frontu.

Výjimky:

- Výjimka XMSException

Metody

Zavření-Zavření prohlížeče fronty

Rozhraní:

```
void  Close();
```

Zavřete prohlížeč front.

Pokud se aplikace pokusí zavřít prohlížeč front, který je již zavřen, volání se ignoruje.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException

GetEnumerator -Získání zpráv

Rozhraní:

```
IEnumerator GetEnumerator();
```

Získejte seznam zpráv ve frontě.

Tato metoda vrací výčtový nástroj, který zapouzdří seznam objektů zpráv. Pořadí objektů zpráv je stejné jako pořadí, ve kterém budou zprávy načítány z fronty. Aplikace pak může použít výčtový nástroj k procházení každé zprávy postupně.

výčtový nástroj se dynamicky aktualizuje, protože zprávy jsou vloženy do fronty a odebrány z fronty. Pokaždé, když aplikace zavolá IEnumerator.MoveNext() k procházení další zprávy ve frontě, zpráva bude odrážet aktuální obsah fronty.

Pokud aplikace volá tuto metodu více než jednou pro prohlížeč front, každý volání vrátí nový výčtový nástroj. Aplikace proto může používat více než jeden výčtový nástroj k procházení zpráv ve frontě a k udržování více pozic ve frontě.

Parametry:
Není

vrátí:
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 následné čekání na odpověď a přijetí odpovědi.

Hierarchie dědičnosti:
Není

Konstruktory

Žadatel-Vytvořit žadatele

Rozhraní:

```
Requestor(ISession sess, IDestination dest);
```

Vytvořte žadatele.

Parametry:

sess (vstup)
Objekt relace. Relace nesmí být zpracovávána a musí mít jeden z následujících režimů potvrzení:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

dest (vstup)
Cílový objekt reprezentující místo určení, kam může aplikace odesílat zprávy požadavků.

Kontext podprocesu:
Relace přidružená k žadateli

Výjimky:

- Výjimka XMSException

Metody

Zavřít-Zavřený žadatel

Rozhraní:

```
void  Close();
```

Zavřete žadatele.

Pokud se aplikace pokusí zavřít žadatele, který je již zavřen, volání se ignoruje.

Poznámka: Když aplikace uzavře žadatele, přidružená relace se nezavře také. V tomto ohledu se produkt XMS chová odlišně v porovnání s platformou JMS.

Parametry:
Není

vrátí:
Void

Kontext podprocesu:
Libovolný

Výjimky:

- Výjimka XMSException

Požadavek-odezva požadavku

Rozhraní:

```
IMessage Request(IMessage requestMessage);
```

Odešlete zprávu s požadavkem a poté vyčkejte a obdržíte odpověď od aplikace, která přijímá zprávu požadavku.

Volání této metody bloků, dokud není přijata odpověď nebo dokud relace neskončí, podle toho, co nastane dříve.

Parametry:

requestMessage (vstup)
Objekt Message shrnující zprávu požadavku.

vrátí:
Ukazatel na objekt Message shrnující 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, GetLinkedException

SecurityException

XMS vyvolá tuto výjimku, je-li identifikátor uživatele a heslo poskytnuté pro ověření aplikace odmítnuty. 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, GetLinkedException

ISession

Relace je jednovláknový kontext pro odesílání a příjem zpráv.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.ISession
```

Seznam definovaných vlastností objektu relace XMS naleznete v tématu “Vlastnosti relace” na stránce 2029.

.NET vlastnosti

AcknowledgeMode -Získat režim potvrzení

Rozhraní:

```
AcknowledgeMode AcknowledgeMode
    {
      get;
    }
```

Získejte režim potvrzení pro relaci.

Režim potvrzení je zadán při vytvoření relace.

V případě, že relace není prováděna, je režim potvrzení jedním z následujících hodnot:

`AcknowledgeMode.AutoAcknowledge`
`AcknowledgeMode.ClientAcknowledge`
`AcknowledgeMode.DupsOkAcknowledge`

Další informace o režimech potvrzení naleznete v tématu Potvrzení zprávy.

Relace, která transakci obsahuje, nemá žádný režim potvrzení. Pokud relace obsahuje transakci, metoda vrátí místo toho metodu `AcknowledgeMode.SessionTransacted` .

Výjimky:

- Výjimka XMSException

Transacted-Určit, zda se transakce provádí

Rozhraní:

```
Boolean Transacted
    {
      get;
    }
```

Určete, zda relace obsahuje transakci.

Uzpracovávána transakce je:

- Je splněn, je-li relace zpracovávána.

- False, pokud relace není transakcemi.

V případě připojení v reálném čase ke zprostředkovateli metoda vždy vrací hodnotu False.

Výjimky:

- Výjimka XMSException

Metody

Zavřít-Zavřít relaci

Rozhraní:

```
void Close();
```

Zavřete relaci. Je-li relace zpracovávána, transakce s probíhaní probíhají bude odvolána.

Pokud se aplikace pokusí zavřít relaci, která je již zavřena, volání se ignoruje.

Parametry:
Není

vrátí:
Void

Kontext podprocesu:
Libovolný

Výjimky:

- Výjimka XMSException

Potvrdit-Potvrdit

Rozhraní:

```
void Commit();
```

Potvrdit všechny zprávy zpracované v aktuální transakci.

Relace musí být relací transakcí.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState
- TransactionRolledBackException

CreateBrowser -Vytvoření prohlížeče front

Rozhraní:

```
IQueueBrowser CreateBrowser(IDestination queue) ;
```

Vytvořte prohlížeč front pro uvedenou frontu.

Parametry:

queue (vstup)
Cílový objekt reprezentující frontu.

vrátí:
Objekt QueueBrowser .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateBrowser -Vytvoření prohlížeče front (se selektorem zpráv)

Rozhraní:

```
IQueueBrowser CreateBrowser(IDestination queue, String selector) ;
```

Vytvořte prohlížeč front pro uvedenou frontu pomocí selektoru zpráv.

Parametry:

queue (vstup)
Cílový objekt reprezentující frontu.

selektor (vstup)
Řetězcový objekt zapouzdřující výraz selektoru zpráv. Do prohlížeče fronty se doručí pouze zprávy s vlastnostmi, které se shodují s výrazem selektoru zpráv.

Objekt s hodnotou null String znamená, že pro prohlížeč front neexistuje žádný selektor zpráv.

vrátí:
Objekt QueueBrowser .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

Zpráva CreateBytesZpráva-Vytvořit bajtovou zprávu

Rozhraní:

```
IBytesMessage CreateBytesMessage();
```

Vytvořte bajtovou zprávu.

Parametry:
Není

vrátí:
Objekt BytesMessage .

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState(relace je zavřena)

CreateConsumer -Vytvořit odběratele

Rozhraní:

```
IMessageConsumer CreateConsumer(IDestination dest) ;
```

Vytvořte spotřebitele zpráv pro určené místo určení.

Parametry:

dest (vstup)
Cílový objekt.

vrátí:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateConsumer -Vytvoření odběratele (s selektorem zpráv)

Rozhraní:

```
IMessageConsumer CreateConsumer(IDestination dest,
                                String selector) ;
```

Vytvořte spotřebitele zpráv pro určené místo určení pomocí selektoru zpráv.

Parametry:

dest (vstup)
Cílový objekt.

selektor (vstup)
Řetězcový objekt zapouzdřující výraz selektoru zpráv. Do konzumentu zpráv jsou doručeny pouze zprávy s vlastnostmi, které se shodují s výrazem selektoru zpráv.

Objekt s hodnotou null String znamená, že pro spotřebitele zpráv neexistuje žádný selektor zpráv.

vrátí:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateConsumer -Vytvoření odběratele (s 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 cílem je téma, určete, zda spotřebitel zpráv přijímá zprávy publikované pomocí vlastního připojení.

Parametry:

dest (vstup)
Cílový objekt.

selektor (vstup)
Řetězcový objekt zapouzdřující výraz selektoru zpráv. Do konzumentu zpráv jsou doručeny pouze zprávy s vlastnostmi, které se shodují s výrazem selektoru zpráv.

Objekt s hodnotou null String 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 přijme zprávy publikované svým vlastním připojením. Výchozí hodnota je false.

vrátí:
Objekt MessageConsumer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

CreateDurableOdběratel-Vytvořit trvalý odběratel

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 Trvalý odběratelé.

Parametry:

dest (vstup)
Cílový objekt reprezentující téma. Téma nesmí být dočasné téma.

odběr (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje trvalý odběr. Název musí být jedinečný v rámci identifikátoru klienta pro připojení.

vrátí:
Objekt MessageConsumer reprezentující trvalého odběratele.

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateDurableOdběratel-Vytvořit trvalý odběratel (se selektorem zpráv a lokálním příznakem zprávy)

Rozhraní:

```
IMessageConsumer CreateDurableSubscriber(IDestination dest,
                                         String subscription,
                                         String selector,
                                         Boolean noLocal) ;
```

Vytvořte trvalého odběratele pro určené téma pomocí selektoru zpráv a určením, zda trvalý odběratel přijímá zprávy publikované pomocí vlastního připojení.

Tato metoda není platná pro připojení v reálném čase ke zprostředkovateli.

Další informace o trvalých odběratelích naleznete v tématu Trvalý odběratelé.

Parametry:

dest (vstup)
Cílový objekt reprezentující téma. Téma nesmí být dočasné téma.

odběr (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje trvalý odběr. Název musí být jedinečný v rámci identifikátoru klienta pro připojení.

selektor (vstup)
Řetězcový objekt zapouzdřující výraz selektoru zpráv. Do trvalého odběratele jsou doručovány pouze zprávy s vlastnostmi, které se shodují s výrazem selektoru zpráv.

Řetězcový objekt s hodnotou Null znamená, že pro trvalého odběratele neexistuje žádný selektor zpráv.

noLocal (vstup)
Hodnota True znamená, že trvalý odběratel nepřijímá zprávy publikované svým vlastním připojením. Hodnota False znamená, že trvalý odběratel přijímá zprávy publikované svým vlastním připojením. Výchozí hodnota je false.

vrátí:
Objekt MessageConsumer reprezentující trvalého odběratele.

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination
- Výjimka InvalidSelector

Zpráva CreateMap-Vytvořit zprávu mapování

Rozhraní:

```
IMapMessage CreateMapMessage();
```

Vytvořte zprávu mapování.

Parametry:
Není

vrátí:
Objekt MapMessage .

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState(relace je zavřena)

CreateMessage -Vytvořit zprávu

Rozhraní:

```
IMessage CreateMessage();
```

Vytvořte zprávu, která nemá tělo.

Parametry:
Není

vrátí:
Objekt Zpráva.

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState(relace je zavřena)

Zpráva CreateObject-Vytvoření zprávy objektu

Rozhraní:

```
IObjectMessage CreateObjectMessage();
```

Vytvořte zprávu objektu.

Parametry:
Není

vrátí:
Objekt ObjectMessage .

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState(relace je zavřena)

CreateProducer -Vytvoření producenta

Rozhraní:

```
IMessageProducer CreateProducer(IDestination dest) ;
```

Vytvořte producenta zpráv pro odesílání zpráv do zadaného místa určení.

Parametry:

dest (vstup)
Cílový objekt.

Určíte-li objekt místa určení s hodnotou Null, bude producent zpráv vytvořen bez místa určení. V takovém případě musí aplikace určit místo určení vždy, když k odeslání zprávy použije producenta zpráv.

vrátí:
Objekt MessageProducer .

Výjimky:

- Výjimka XMSException
- Výjimka InvalidDestination

CreateQueue -Vytvořit frontu

Rozhraní:

```
IDestination CreateQueue(String queue) ;
```

Vytvořte objekt místa určení, který bude reprezentovat frontu na serveru systému zpráv.

Tato metoda nevytvoří frontu na serveru systému zpráv. Frontu musíte vytvořit dříve, než bude moci aplikace volat tuto metodu.

Parametry:

queue (vstup)
Řetězcový objekt zapouzdřující název fronty nebo zapouzdření identifikátoru URI (Uniform Resource Identifier), který identifikuje frontu.

vrátí:
Cílový objekt reprezentující frontu.

Výjimky:

- Výjimka XMSException

Zpráva CreateStream-Vytvořit proudovou zprávu

Rozhraní:

```
IStreamMessage CreateStreamMessage();
```

Vytvořte zprávu proudu.

Parametry:
Není

vrátí:
Objekt StreamMessage .

Výjimky:

- Výjimka XMSException
- VÝJIMKA XMS_ILLEGAL_STATE_EXCEPTION

CreateTemporaryQueue-Vytvoření dočasné fronty

Rozhraní:

```
IDestination CreateTemporaryQueue() ;
```

Vytvořte dočasnou frontu.

Obor dočasné fronty je připojení. Pouze relace vytvořené připojením mohou použít dočasnou frontu.

Dočasná fronta zůstává, dokud není explicitně odstraněna, nebo dokud se připojení neukončí, podle toho, co nastane dříve.

Další informace o dočasných frontách najdete v tématu Dočasná místa určení.

Parametry:
Není

vrátí:
Cílový objekt reprezentující dočasnou frontu.

Výjimky:

- Výjimka XMSException

Téma CreateTemporary-Vytvořit dočasné téma

Rozhraní:

```
IDestination CreateTemporaryTopic() ;
```

Vytvořte dočasné téma.

Rozsah dočasného tématu je připojení. Pouze relace vytvořené připojením mohou používat dočasné téma.

Dočasné téma zůstává do okamžiku, kdy je explicitně odstraněno, nebo pokud připojení skončí, podle toho, co nastane dříve.

Další informace o dočasných tématech naleznete v tématu Dočasná místa určení.

Parametry:
Není

vrátí:
Cílový objekt reprezentující dočasné téma.

Výjimky:

- Výjimka XMSException

CreateTextZpráva-Vytvořit textovou zprávu

Rozhraní:

```
ITextMessage CreateTextMessage();
```

Vytvořte textovou zprávu s prázdným tělem.

Parametry:
Není

vrátí:
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 (vstup)
Objekt typu String, který zapouzdřuje text, aby inicializoval tělo textové zprávy.

Není

vrátí:
Objekt TextMessage .

Výjimky:

- Výjimka XMSException

CreateTopic -Vytvořit téma

Rozhraní:

```
IDestination CreateTopic(String topic) ;
```

Vytvoření cílového objektu pro znázornění tématu.

Parametry:

topic (vstup)
Řetězcový objekt zapouzdřující název tématu nebo zapouzdření identifikátoru URI (Uniform Resource Identifier), který identifikuje dané téma.

vrátí:
Cílový objekt reprezentující téma.

Výjimky:

- Výjimka XMSException

Obnova-obnova

Rozhraní:

```
void Recover();
```

Obnovte relaci. Doručování zpráv je zastaveno a pak restartováno s nejstarší nepotvrzenou zprávou.

Relace nesmí být relací s transakcemi.

Další informace o obnově relace naleznete v tématu Potvrzení zprávy.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

Odvolání-odvolání

Rozhraní:

```
void Rollback();
```

Odvolejte všechny zprávy zpracované v aktuální transakci.

Relace musí být relací transakcí.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- Výjimka IllegalState

Zrušit odběr-Zrušit odběr

Rozhraní:

```
void Unsubscribe(String subscription);
```

Odstraňte trvalý odběr. Server systému zpráv odstraní záznam o trvalém odběru, který spravuje, a neodešle další zprávy trvalému odběrateli.

Aplikace nemůže odstranit trvalý odběr za následujících okolností:

- Existuje aktivní spotřebitel zpráv pro trvalý odběr
- Spotřebovaná zpráva je součástí nevyřízené transakce.
- Spotřebovaná zpráva nebyla potvrzena

Tato metoda není platná pro připojení v reálném čase ke zprostředkovateli.

Parametry:

odběr (vstup)
Řetězcový objekt zapouzdřující název, který identifikuje trvalý odběr.

vrátí:
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

Proudová zpráva je zpráva, jejíž tělo tvoří proud hodnot, kde má každá hodnota přidružený datový typ. Obsah těla se zapisuje a čte postupně.

Hierarchie dědičnosti:

```
IBM.XMS.IPropertyContext
   |
   +----IBM.XMS.IMessage
           |
           +----IBM.XMS.IStreamMessage
```

Když aplikace načte hodnotu z proudu zpráv, může ji převést hodnota XMS do jiného datového typu. Další informace o této formě implicitní konverze najdete v tématu Tělo zprávy XMS.

Metody

ReadBoolean -Čtení hodnoty typu Boolean

Rozhraní:

```
Boolean ReadBoolean();
```

Přečtěte si logickou hodnotu z proudu zpráv.

Parametry:
Není

vrátí:
Logická hodnota, která je přečtená.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadByte -Čtení bajtů

Rozhraní:

```
Int16   ReadSignedByte();
    Byte    ReadByte();
```

Přečtěte si 8bitové celé číslo z proudu zpráv.

Parametry:
Není

vrátí:
Bajt, který je čten.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadBytes -Čtení bajtů

Rozhraní:

```
Int32   ReadBytes(Byte[] array);
```

Přečtěte si pole bajtů z proudu zpráv.

Parametry:

array (vstup)
Vyrovnávací paměť obsahující pole bajtů, které se čtou, a délku vyrovnávací paměti v bajtech.

Je-li 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, je vyrovnávací paměť zaplněna částí pole a vnitřní kurzor označuje pozici dalšího bajtu, který se má přečíst. Následující volání do readBytes() čte bajty z pole od aktuální pozice kurzoru.

Uvedete-li na vstupu ukazatel null, přeskočí volání přes pole bajtů, aniž by jej četl.

vrátí:
Počet bajtů, které jsou načteny do vyrovnávací paměti. Je-li vyrovnávací paměť částečně vyplněna, hodnota je menší než délka vyrovnávací paměti, což znamená, že v poli, které zbývá přečíst, nejsou žádné další bajty. Pokud před voláním nezbývají žádné bajty, které by bylo možné přečíst z pole, hodnota je `XMSC_END_OF_BYTEARRAY`.

Uvedete-li na vstupu ukazatel null, metoda nevrátí žádnou hodnotu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadChar -Čtení znaků

Rozhraní:

```
Char    ReadChar();
```

Čtení 2-bajtového znaku z proudu zpráv.

Parametry:
Není

vrátí:
Znak, který se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadDouble -Přečíst číslo s pohyblivou řádovou čárkou s dvojitou přesností

Rozhraní:

```
Double  ReadDouble();
```

Přečtěte si 8bajtové číslo s pohyblivou řádovou čárkou o dvojnásobné přesnosti z proudu zpráv.

Parametry:
Není

vrátí:
Číslo s plovoucí řádovou čárkou a dvojitou přesností, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadFloat -Čtení čísla s pohyblivou řádovou čárkou

Rozhraní:

```
Single  ReadFloat();
```

Přečtěte si 4bajtové číslo s pohyblivou řádovou čárkou z proudu zpráv.

Parametry:
Není

vrátí:
Číslo v plovoucí řádové čárce, které se čte.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadInt -Přečíst celé číslo

Rozhraní:

```
Int32   ReadInt();
```

Čtení 32bitového celého čísla z proudu zpráv.

Parametry:
Není

vrátí:
Celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadLong -Číst dlouhé celé číslo

Rozhraní:

```
Int64   ReadLong();
```

Přečtěte si 64bitové celé číslo se znaménkem, které je ze zprávy proudu.

Parametry:
Není

vrátí:
Dlouhé celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadObject -Číst objekt

Rozhraní:

```
Object  ReadObject();
```

Přečtěte si hodnotu z proudu zpráv a vraťte její datový typ.

Parametry:
Není

vrátí:
Hodnota, která je jedním z následujících typů objektů:

```
Boolean
Byte
Byte[]
Char
Double
Single
Int32
Int64
Int16
String
```

Výjimky:
Výjimka XMSException

ReadShort -Přečíst krátké celé číslo

Rozhraní:

```
Int16   ReadShort();
```

Přečíst podepsané 16bitové celé číslo z proudu zpráv.

Parametry:
Není

vrátí:
Krátké celé číslo, které je přečteno.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

ReadString -Číst řetězec

Rozhraní:

```
String  ReadString();
```

Číst řetězec z proudu zpráv. Je-li to nutné, produkt XMS převede znaky v řetězci na lokální kódovou stránku.

Parametry:
Není

vrátí:
Řetězcový objekt zapouzdřující řetězec, který se čte. Je-li požadován převod dat, je tento řetězec po převodu.

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

Reset-Reset

Rozhraní:

```
void  Reset();
```

Umístěte tělo zprávy do režimu jen pro čtení a přesuňte kurzor na začátek toku zpráv.

Parametry:
Není

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotReadableException
- MessageEOFException

WriteBoolean -Zapsat logickou hodnotu

Rozhraní:

```
void  WriteBoolean(Boolean value);
```

Napište logickou hodnotu do proudu zpráv.

Parametry:

hodnota (vstup)
Logická hodnota, která má být zapsána.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteByte -Zápis bajtu

Rozhraní:

```
void  WriteByte(Byte value);
void  WriteSignedByte(Int16 value);
```

Napište bajt do proudu zpráv.

Parametry:

hodnota (vstup)
Bajt, který se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteBytes -Zápis bajtů

Rozhraní:

```
void  WriteBytes(Byte[] value);
```

Zapisovat pole bajtů do proudu zpráv.

Parametry:

hodnota (vstup)
Pole bajtů, které mají být zapsány.

délka (vstup)
Počet bajtů v poli.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteChar -Zápis znaku

Rozhraní:

```
void  WriteChar(Char value);
```

Napište znak do proudu zpráv jako 2 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Znak, který má být zapsán.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteDouble -Write Double Precision Floating Point Number

Rozhraní:

```
void  WriteDouble(Double value);
```

Převeďte číslo s pohyblivou řádovou čárkou a dvojitou přesností na dlouhé celé číslo a zapište dlouhé celé číslo do toku zpráv jako 8 bajtů, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Číslo s plovoucí řádovou čárkou s dvojitou přesností, které se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteFloat -Zápis čísla s pohyblivou řádovou čárkou

Rozhraní:

```
void  WriteFloat(Single value);
```

Převeďte číslo s plovoucí řádovou čárkou na celé číslo a zapište celé číslo do proudu zpráv jako 4 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Číslo plovoucí řádové čárky, které se má zapsat.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteInt -Zapsat celé číslo

Rozhraní:

```
void  WriteInt(Int32 value);
```

Zapište celé číslo do proudu zpráv jako 4 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteLong -zapište dlouhé celé číslo

Rozhraní:

```
void  WriteLong(Int64 value);
```

Zapište dlouhé celé číslo do toku zpráv jako 8 bajtů, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Dlouhé celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteObject -Zapsat objekt

Rozhraní:

```
void  WriteObject(Object value);
```

Zadejte hodnotu do proudu zpráv se zadaným datovým typem.

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.

vrátí:
Void

Výjimky:

- Výjimka XMSException

WriteShort -zapište krátké celé číslo

Rozhraní:

```
void  WriteShort(Int16 value);
```

Napište krátké celé číslo do toku zpráv jako 2 bajty, bajt s velkým pořadovým číslem jako první.

Parametry:

hodnota (vstup)
Krátké celé číslo, které má být zapsáno.

vrátí:
Void

Výjimky:

- Výjimka XMSException
- MessageNotWritableException

WriteString -Řetězec zápisu.

Rozhraní:

```
void  WriteString(String value);
```

Napište řetězec do proudu zpráv.

Parametry:

hodnota (vstup)
Řetězcový objekt zapouzdřující řetězec, který má být zapsán.

vrátí:
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 tvoří 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.

Je-li to nutné, 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 požádá o 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, GetLinkedException

TransactionRolledBackException

XMS vyvolá tuto výjimku, pokud aplikace volá Session.commit() k potvrzení aktuální transakce, ale transakce se pak vrátí zpět.

Hierarchie dědičnosti:

```
IBM.XMS.XMSException
   |
   +----IBM.XMS.XMSException
           |
           +----IBM.XMS.TransactionRolledBackException
```

Zděděné vlastnosti a metody

Níž uvedené metody jsou zděděny z rozhraní XMSException:

GetErrorCode, GetLinkedException

Výjimka XMSException

Pokud XMS zjistí chybu během zpracování volání metody .NET , příkaz XMS vyvolá výjimku. Výjimkou je objekt, který zapouzdřuje informace o chybě.

Hierarchie dědičnosti:

```
System.Exception
   |
   +----IBM.XMS.XMSException
```

Existují různé typy výjimek XMS a objekt XMSException je pouze jedním typem výjimky. Třída XMSException je však nadtřídou jiných tříd výjimek XMS . XMS generuje objekt XMSException v situacích, kdy není vhodný žádný z jiných typů výjimek.

.NET vlastnosti

ErrorCode -Získat kód chyby

Rozhraní:

```
public String ErrorCode
{
  get {return errorCode_;}
}
```

Získejte kód chyby.

Výjimky:

- Výjimka XMSException

LinkedException -Získat propojené výjimky

Rozhraní:

```
public Exception LinkedException
{
  get { return linkedException_;}
```

```
  set { linkedException_ = value;}
}
```

Získejte další výjimku v řetězci výjimek.

Pokud v řetězci nejsou žádné další výjimky, vrací metoda hodnotu null.

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-Načíst metadata

Rozhraní:

```
IConnectionMetaData MetaData
```

Získejte metadata, která odpovídají typu připojení objektu XMSFactoryFactory .

Výjimky:
Není

Metody

CreateConnectionTovárna-Vytvořit továrnu připojení

Rozhraní:

```
IConnectionFactory CreateConnectionFactory();
```

Vytvořte objekt ConnectionFactory deklarovaného typu.

Parametry:
Není

vrátí:
Objekt ConnectionFactory .

Výjimky:

- Výjimka XMSException

CreateQueue -Vytvořit frontu

Rozhraní:

```
IDestination CreateQueue(String name);
```

Vytvořte objekt místa určení, který bude reprezentovat frontu na serveru systému zpráv.

Tato metoda nevytvoří frontu na serveru systému zpráv. Frontu musíte vytvořit dříve, než bude moci aplikace volat tuto metodu.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název fronty nebo zapouzdření identifikátoru URI (Uniform Resource Identifier), který identifikuje frontu.

vrátí:
Cílový objekt reprezentující frontu.

Výjimky:

- Výjimka XMSException

CreateTopic -Vytvořit téma

Rozhraní:

```
IDestination CreateTopic(String name);
```

Vytvoření cílového objektu pro znázornění tématu.

Parametry:

name (vstup)
Řetězcový objekt zapouzdřující název tématu nebo zapouzdření identifikátoru URI (Uniform Resource Identifier), který identifikuje dané téma.

vrátí:
Cílový objekt reprezentující téma.

Výjimky:

- Výjimka XMSException

GetInstance -Získání instance třídy 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 které objekt ConnectionFactory produkuje připojení:

- XMSC.CT_WPM
- XMSC.CT_RTT
- XMSC.CT_WMQ

vrátí:
Objekt XMSFactoryFactory vyhrazený pro deklarovaný typ připojení.

Výjimky:

- Výjimka NotSupported

Vlastnosti objektů XMS

Tato sekce dokumentuje vlastnosti objektu definované produktem XMS.

Tato sekce obsahuje informace o následujících typech objektů:

- “Vlastnosti připojení” na stránce 2015

- “Vlastnosti objektu ConnectionFactory” na stránce 2016
- “Vlastnosti dat ConnectionMeta” na stránce 2021
- “Vlastnosti místa určení” na stránce 2022
- “Vlastnosti objektu InitialContext” na stránce 2023
- “Vlastnosti zprávy” na stránce 2024
- “Vlastnosti objektu MessageConsumer” na stránce 2029
- “Vlastnosti objektu MessageProducer” na stránce 2029
- “Vlastnosti relace” na stránce 2029

Popis každého typu objektu uvádí vlastnosti objektu uvedeného typu a poskytuje krátký popis každé vlastnosti.

Tato sekce také poskytuje definici každé vlastnosti (viz “Definice vlastností” na stránce 2029).

Pokud aplikace definuje své vlastní vlastnosti objektů popsaných v této sekci, nezpůsobí chybu, ale může dojít k nepředvídatelným výsledkům.

Poznámka: Názvy a hodnoty vlastností v této sekci jsou zobrazeny ve tvaru XMSC.*NAME*, což je formulář používaný pro jazyky C a C + +. Avšak v produktu .NETmůže být název vlastnosti XMSC.*NAME* nebo XMSC_*NAME*, v závislosti na tom, jak ji používáte:

- Pokud určujete vlastnost, název vlastnosti musí být ve tvaru XMSC.*NAME* , jak je uvedeno v následujícím příkladu:

  ```
  cf.SetStringProperty(XMSC.WMQ_CHANNEL, "DOTNET.SVRCONN");
  ```

- Pokud zadáváte řetězec, musí být název vlastnosti ve tvaru XMSC_*NAME* tak, jak je uvedeno v následujícím příkladu:

  ```
  cf.SetStringProperty("XMSC_WMQ_CHANNEL", "DOTNET.SVRCONN");
  ```

  V produktu .NETjsou názvy vlastností a hodnoty zadány jako konstanty ve třídě XMSC. Tyto konstanty identifikují řetězce a budou použity libovolnou aplikací produktu XMS .NET . Používáte-li tyto předdefinované konstanty, názvy a hodnoty vlastností jsou ve formátu XMSC.*NAME*, takže byste například mohli použít XMSC.USERID, spíše než XMSCS_USERID.

Datové typy jsou také ve formuláři použitém pro C/C + +. V části Datové typy pro .NETmůžete najít odpovídající hodnoty pro .NET .

Vlastnosti připojení

Přehled vlastností objektu připojení s odkazy na podrobnější referenční informace.

Tabulka 873. Vlastnosti připojení

Název vlastnosti	**Popis**
“XMSC_WMQ_RESOLVE_QUEUE_MANAGER” na stránce 2063	Tato vlastnost se používá k získání názvu správce front, ke kterému je připojen.
“XMSC_WMQ_RESOLVE_QUEUE_MANAGER_ID” na stránce 2063	Tato vlastnost je naplněna identifikátorem ID správce front po připojení.
XMSC_WPM_CONNECTION_PROTOCOL	Komunikační protokol použitý pro připojení k jádru systému zpráv. Tato vlastnost je jen pro čtení.
NÁZEV_HOSTITEL_XMSC	Název hostitele nebo adresa IP systému, který obsahuje jádro systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.
XMSCS_WPM_ME_NAME	Název jádra systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

Tabulka 873. Vlastnosti připojení (pokračování)	
Název vlastnosti	**Popis**
XCSMS_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 určené jen pro čtení, které jsou odvozeny od vlastností továrny připojení, která byla použita k vytvoření připojení. Tyto vlastnosti se odvozují nejen od vlastností továrny připojení, které byly nastaveny v době vytvoření připojení, ale také z výchozích hodnot vlastností, které nebyly nastaveny. Vlastnosti zahrnují pouze ty vlastnosti, které jsou relevantní pro typ serveru systému zpráv, ke kterému je aplikace připojena. Názvy vlastností jsou stejné jako názvy vlastností továrny připojení.

Vlastnosti objektu ConnectionFactory

Přehled vlastností objektu ConnectionFactory s odkazy na podrobnější referenční informace.

Tabulka 874. Vlastnosti objektu ConnectionFactory	
Název vlastnosti	**Popis**
“XMSC_ASYNC_EXCEPTIONS” na stránce 2039	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.
V 9.2.4 “XMSC_WMQ_BALANCING_APPLICATION_TYPE” na stránce 2047	Typ volby vyvážení
V 9.2.4 “XMSC_WMQ_BALANCING_OPTIONS” na stránce 2048	Volby vyvážení nastavené vydáním aplikace
V 9.2.4 “XMSC_WMQ_BALANCING_TIMEOUT” na stránce 2048	Časový limit, po jehož uplynutí může opětovné vyvážení přerušit aktivitu aplikace.
ID XMSCS_CLIENT_ID	Identifikátor klienta pro účely připojení.
XMSC_CONNECTION_TYPE	Typ serveru systému zpráv, ke kterému se aplikace připojuje.
HESLO_HESLO	Heslo, které lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv.
“XMSC_RTC_BROKER_PING_INTERVAL” na stránce 2044	Časový interval v milisekundách, po který XMS .NET kontroluje připojení k serveru systému zpráv v reálném čase ke zjištění jakékoli aktivity.
XMSCS_TRT_CONNECTION_PROTOCOL	Komunikační protokol použitý pro připojení v reálném čase ke zprostředkovateli.
NÁZEV_HOSTITELE XMSCS_RTT_NAME	Název nebo adresa IP hostitele systému, na kterém je spuštěn zprostředkovatel.

Tabulka 874. Vlastnosti objektu ConnectionFactory (pokračování)	
Název vlastnosti	**Popis**
ADRESA_XCST_RT_LOCAL_ADDRESS	Název hostitele nebo adresa IP lokálního síťového rozhraní, které má být použito pro připojení v reálném čase ke zprostředkovateli.
XMSCS_RTT_MULTICAST	Nastavení výběrového vysílání pro továrnu připojení nebo cíl.
VÝPORT XMSCS_RTT_PORT	Číslo portu, na kterém zprostředkovatel naslouchá příchozím požadavkům.
ID UŽIVATELE XMSC	Identifikátor uživatele, který lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv.
XMSCS_WMQ_BROKER_CONTROLQ	Název řídicí fronty používané zprostředkovatelem. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
XMSCS_WMQ_BROKER_PUBQ	Název fronty monitorované zprostředkovatelem, kde aplikace odesílají zprávy, které publikují. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
XMSCS_WMQ_BROKER_QMGR	Název správce front, ke kterému je zprostředkovatel připojen. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
XMSCS_WMQ_BROKER_SUBQ	Název fronty odběratele pro spotřebitele netrvalých zpráv. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.

Tabulka 874. Vlastnosti objektu ConnectionFactory (pokračování)	
Název vlastnosti	**Popis**
XMSCS_WMQ_BROKER_VERSION	Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
"XMSC_WMQ_CCDTURL" na stránce 2050	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.
XMSCS_WMQ_CHANNEL	Název kanálu, který se má použít pro připojení.
"XMSC_WMQ_CLIENT_RECONNECT_OPTIONS" na stránce 2051	Tato vlastnost uvádí volby nového připojení klienta pro nová připojení vytvořená touto továrnou
"XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT" na stránce 2051	Tato vlastnost určuje dobu trvání (v sekundách), po kterou se připojení klienta pokouší znovu navázat spojení.
XMSC_WMQ_CONNECTION_MODE	Režim, pomocí kterého se aplikace připojuje ke správci front.
"XMSC_WMQ_CONNECTION_NAME_LIST" na stránce 2052	Tato vlastnost určuje hostitele, ke kterému se klient pokusí znovu připojit po přerušení připojení.
XMSCS_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.
NÁZEV_HOSTITEL_XMSC	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. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
XMSCS_WMQ_MSG_BATCH_SIZE	Maximální počet zpráv, které mají být načteny z fronty v jedné dávce, v případě asynchronního doručování zpráv. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.

Tabulka 874. Vlastnosti objektu ConnectionFactory (pokračování)	
Název vlastnosti	**Popis**
XMSCS_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. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
“XMSC_WMQ_PROVIDER_VERSION” na stránce 2060	Verze, vydání, úroveň modifikace a opravný balík správce front, ke kterému se tato aplikace hodlá připojovat.
PORT XMSCS_WMQ_PORT	Číslo portu, na kterém správce front naslouchá příchozím požadavkům.
INTER_INTERVALOVÝ_ČASOVÝ_ADRESA_XMSC	Počet zpráv publikovaných vydavatelem, než klient XMS vyžádá potvrzení od zprostředkovatele. **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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
“XMSC_WMQ_PUT_ASYNC_ALLOWED” na stránce 2056	Tato vlastnost určuje, zda producenti zpráv mohou používat k odesílání zpráv do tohoto místa určení asynchronní operace put.
XMSCS_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 .
XMSCS_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.
XMSCS_WMQ_RECEIVE_EXIT_INIT	Uživatelská data, která jsou předána uživatelské proceduře pro přijetí zprávy kanálu při volání.
XMSCS_WMQ_SECURITY_EXIT	Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.
XMSCS_WMQ_SECURITY_EXIT_INIT	Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.
“XMSC_WMQ_SEND_CHECK_COUNT” na stránce 2065	Počet povolených odeslaných volání mezi kontrolou asynchronních chyb vložení během jedné netransakční relace XMS.
XCS_WMQ_SEND_EXIT	Identifikuje uživatelskou proceduru pro odeslání zprávy kanálu.

Tabulka 874. Vlastnosti objektu ConnectionFactory (pokračování)	
Název vlastnosti	**Popis**
XMSCS_WMQ_SEND_EXIT_INIT	Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.
"XMSC_WMQ_SHARE_CONV_ALLOWED" na stránce 2065	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.
XMSCS_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.
XMSCS_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.
XMSCS_WMQ_CRYPTO_HW	Podrobnosti konfigurace šifrovacího hardwaru připojeného k systému klienta.
POŽAD_ZÁSOBNÍ_CHYBA_XMSC	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.
XMSCS_WMQ_SSL_KEY_REPOSITORY	Umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty.
XMSCS_WMQ_SSL_KEY_RESETCOUNT	Hodnota KeyResetCount představuje celkový počet nešifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL, než je znovu vyjednán tajný klíč.
XMSCS_WMQ_SSL_PEER_NAME	Název typu peer, které se použije při připojení SSL ke správci front.
XMSCS_WMQ_SYNCINT_ALL_GETS	Určuje, zda musí být všechny zprávy načteny z front v rámci řízení synchronizačního bodu.
"XMSC_WMQ_TARGET_CLIENT" na stránce 2072	
XMSCS_WMQ_TEMP_Q_PREFIX	Předpona použitá k vytvoření názvu dynamické fronty IBM MQ , která se vytvoří, když aplikace vytvoří XMS dočasnou frontu.
TEPLOT_TEPLOTA XMSCS_WMQ__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.

Tabulka 874. Vlastnosti objektu ConnectionFactory (pokračování)	
Název vlastnosti	**Popis**
Model XMSCS_WMQ_TEMPORARY_MODEL	Název modelové fronty IBM MQ , ze které se vytvoří dynamická fronta, když aplikace vytvoří XMS dočasnou frontu.
XMSCS_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.
ADRESA_DOP_SUB_V_ADRESÁŘ_XMSC	Název jádra systému zpráv, v nichž jsou spravovány všechny trvalé odběry pro připojení nebo cíl.
XMSCS_WPM_LOCAL_ADDRESS	Pro připojení ke sběrnici SIBus tato vlastnost určuje rozhraní lokální sítě, lokálního portu nebo rozsahu lokálních portů, nebo obojí.
XMSCS_WPM_NON_PERSISTENT_MAP	Úroveň spolehlivosti přechodných zpráv, které se odesílají pomocí připojení.
XMSCS_WPM_PERSISTENT_MAP	Úroveň spolehlivosti trvalých zpráv, které se odesílají pomocí připojení.
XMSCS_WPM_PROVIDER_ENDPOINTS	Sekvence jednoho nebo více adres koncového bodu zaváděcích serverů.
XMSCS_WPM_TARGET_GROUP	Název cílové skupiny jader systému zpráv.
XMSC_WPM_TARGET_TARGET_SIGNIFICANCE	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.
XMSCS_WPM_TARGET_TYPE	Typ cílové skupiny jader systému zpráv.
XMSCS_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.
XMSCS_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 875. Vlastnosti dat ConnectionMeta	
Název vlastnosti	**Popis**
XMSCS_JMS_MAJOR_VERZE	Číslo hlavní verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
XMSCS_JMS_MINOR_VERSION	Číslo vedlejší verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
XMSCS_JMS_VERSION	Identifikátor verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.
VÝZNAM_VELIKO_XMSC	Číslo verze klienta XMS . Tato vlastnost je jen pro čtení.
XMSCS_MINOR_VERSION	Číslo vydání klienta XMS . Tato vlastnost je jen pro čtení.

Tabulka 875. Vlastnosti dat ConnectionMeta (pokračování)	
Název vlastnosti	**Popis**
NÁZEV_ADRESÁŘ_XMSC	Poskytovatel klienta XMS . Tato vlastnost je jen pro čtení.
VERZE XMSCS_VERSION	Identifikátor verze rozhraní cliXMSent. Tato vlastnost je jen pro čtení.

Vlastnosti místa určení

Přehled vlastností objektu Destination, s odkazy na podrobnější referenční informace.

Tabulka 876. Vlastnosti místa určení	
Název vlastnosti	**Popis**
REŽIM XMSCS_DELIVERY_MODE	Režim doručení zpráv odeslaných do cíle.
PRIORITA XMSCSC	Priorita zpráv odeslaných do cíle.
XMSCS_RTT_MULTICAST	Nastavení výběrového vysílání pro továrnu připojení nebo cíl.
XMSCS_TIME_TO_LIVE	Doba životnosti zpráv odeslaných do cíle.
XMSCS_WMQ_BROKER_VERSION	Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl.
CCSID XMSCS_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.
DÍL_UPO_WMQTMQ XMSC	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 vlastnost XMSCS_WMQ_PROVIDER_VERSION dané továrny připojení není nastavena na číslo verze nižší než 7.
XMSCS_WMQ_ENCODING	Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do místa určení.
XMSCS_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_MESSAGE_BODY” na stránce 2054	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 2055	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 2055	Tato vlastnost určuje, zda může aplikace XMS extrahovat hodnoty polí MQMD.
“XMSC_WMQ_MQMD_WRITE_ENABLED” na stránce 2056	Tato vlastnost určuje, zda může aplikace XMS nastavit hodnoty polí MQMD.

Tabulka 876. Vlastnosti místa určení (pokračování)	
Název vlastnosti	**Popis**
“XMSC_WMQ_READ_AHEAD_ALLOWED” na stránce 2057	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 2057	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 2062	Vlastnost místa určení, která nastavuje cílové CCSID pro převod zpráv správce front. Hodnota je ignorována, pokud není hodnota XMSCS_WMQ_RECEIVE_CONVERSION nastavena na hodnotu WMQ_RECEIVE_CONVERSION_QMGR.
“XMSC_WMQ_RECEIVE_CONVERSION” na stránce 2062	Cílová vlastnost, která určuje, zda bude převod dat prováděn správcem front.
CLIENT_WMQ_TARGET_CLIENT	Určuje, zda zprávy odeslané do cíle obsahují záhlaví MQRFH2.
TEPLOT_TEPLOTA XMSCS_WMQ__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.
XMSCS_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.
XMSCS_WPM_TOPIC_SPACE	Název prostoru témat, který obsahuje dané téma.

Vlastnosti objektu InitialContext

Přehled vlastností objektu InitialContext s odkazy na podrobnější referenční informace.

Tabulka 877. Vlastnosti objektu InitialContext	
Název vlastnosti	**Popis**
Adresa URL XMSCS_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.

Tabulka 877. Vlastnosti objektu InitialContext (pokračování)	
Název vlastnosti	**Popis**
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.
Adresa XMSCS_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 878. Vlastnosti zprávy	
Název vlastnosti	**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_EXCEPTION_MESSAGE	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.
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.
ČAS SOUBORU 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.

Tabulka 878. Vlastnosti zprávy (pokračování)	
Název vlastnosti	**Popis**
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.
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.
JMS_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.
JMS_USERID	Identifikátor uživatele přidružený k aplikaci, která odeslala zprávu.

Vlastnosti JMS_IBM_MQMD*

IBM Message Service Client for .NET umožňuje klientským aplikacím číst/zapisovat pole MQMD pomocí rozhraní API. Umožňuje také přístup k datům zprávy produktu MQ . Standardně je přístup k produktu MQMD zakázán a musí být explicitně povolen aplikací s použitím vlastností cíle XMSCS_WMQ_MQMD_WRITE_ENABLED a XMSCS_WMQ_MQMD_READ_ENABLED. Tyto dvě vlastnosti jsou na sobě nezávislé.

Všechna pole MQMD s výjimkou StrucId a verze jsou odkryty jako další vlastnosti objektu zprávy a mají předponu JMS_IBM_MQMD.

Vlastnosti JMS_IBM_MQMD* mají vyšší prioritu než jiné vlastnosti, jako např. JMS_IBM* popsané v předchozí tabulce.

Odesílání zpráv

Všechna pole MQMD s výjimkou StrucId a verze jsou reprezentovány. Tyto vlastnosti se odkazují pouze na pole MQMD, kde se vlastnost vyskytuje jak v deskriptoru MQMD, tak v záhlaví MQRFH2 , verze v MQRFH2 není nastavena ani extrahována. Je možné nastavit libovolnou z těchto vlastností, kromě JMS_IBM_MQMD_BackoutCount. Jakákoli hodnota nastavená pro parametr JMS_IBM_MQMD_BackoutCount je ignorována.

Má-li vlastnost maximální délku a zadáte příliš dlouhou hodnotu, bude tato hodnota zkrácena.

Pro určité vlastnosti musíte také nastavit vlastnost XMSCS_WMQ_MQMD_MESSAGE_CONTEXT na objektu Destination. Aby tato vlastnost mohla nabýt účinnosti, musí aplikace běžet s příslušným oprávněním kontextu. Pokud nenastavíte hodnotu XMSCS_WMQ_MQM_MESSAGE_CONTEXT na příslušnou hodnotu, bude hodnota vlastnosti ignorována. Pokud jste nastavili XMSER_WMQ_MQMD_MESSAGE_CONTEXT na odpovídající hodnotu, ale nemáte dostatečné oprávnění ke kontextu pro správce front, bude vydána výjimka. Vlastnosti vyžadující specifické hodnoty parametru XMSCS_WMQ_MQMD_MESSAGE_CONTEXT jsou následující.

Následující vlastnosti vyžadují, aby byl objekt XMSCS_WMQ_MQMD_MESSAGE_CONTEXT nastaven na hodnotu XMSCS_WMQ_MDCTX_SET_IDENTITY_CONTEXT nebo XMSCS_WMQ_MDCTX_SET_ALL_CONTEXT:

- JMS_IBM_MQMD_UserIdentifier .
- JMS_IBM_MQMD_AccountingToken .
- JMS_IBM_MQMD_ApplIdentityData

Následující vlastnosti vyžadují, aby byl objekt XMSCS_WMQ_MQMD_MESSAGE_CONTEXT nastaven na hodnotu XMSCS_WMQ_MDCTX_SET_ALL_CONTEXT:

- JMS_IBM_MQMD_PutApplTyp
- Název JMS_IBM_MQMD_PutAppl
- JMS_IBM_MQMD_PutDate .
- JMS_IBM_MQMD_PutTime .
- JMS_IBM_MQMD_ApplOriginData

Příjem zpráv

Všechny tyto vlastnosti jsou k dispozici v přijaté zprávě, je-li vlastnost XMSCS_WMQ_MQMD_READ_ENABLED nastavena na hodnotu true, a to bez ohledu na skutečné vlastnosti, které vytváří daná sada aplikací. Aplikace nemůže upravit vlastnosti přijaté zprávy, pokud nejsou nejprve vymazány všechny vlastnosti, podle specifikace JMS. Obdrženou zprávu lze předat bez úpravy vlastností.

Poznámka: Obdrží-li vaše aplikace zprávu z vlastnosti XMSCS_WMQ_MQMD_READ_ENABLED nastavenou na hodnotu true a předá ji cíli s hodnotou XMSCS_WMQ_MQMD_WRITE_ENABLED nastavenou na hodnotu true, budou mít tyto výsledky ve všech hodnotách polí MQMD přijaté zprávy do předané zprávy. Tabulka vlastností

Tabulka 879. Vlastnosti objektu zprávy reprezentující pole MQMD

Vlastnost	**Popis**	**Typ**
ZPRÁVA JMS_IBM_MQMD_REPORT	Volby pro zprávy sestav	System.Int32
JMS_IBM_MQMD_MSGTYPE	Typ zprávy	System.Int32
VYPRŠENÍ PLATNOSTI JMS_IBM_MQMD_EXPIRY	doba trvání zprávy	System.Int32
ZPĚTNÁ VAZBA JMS_IBM_MQMD_FEEDBACK	Zpětná vazba nebo kód příčiny	System.Int32

Tabulka 879. Vlastnosti objektu zprávy reprezentující pole MQMD (pokračování)		
Vlastnost	**Popis**	**Typ**
KÓDOVÁNÍ JMS_IBM_MQMD_ENCODING	Číselný kódování dat zprávy	System.Int32
JMS_IBM_MQMD_CODEDCHARSETID	Identifikátor znakové sady dat zprávy	System.Int32
FORMÁT JMS_IBM_MQMD_FORMAT	Název formátu dat zprávy	System.String
PRIORITA JMS_IBM_MQMD_PRIORITY **Poznámka:** Přiřadíte-li hodnotu JMS_IBM_MQMD_PRIORITY, která není v rozsahu 0-9, tato hodnota porušuje specifikaci JMS.	Priorita zprávy	System.Int32
PERZISTENTNÍ PERZISTENCE JMS_IBM_MQMD_	Trvalost zpráv	System.Int32
MSGID JMS_IBM_MQMD_MSGID **Poznámka:** Ve specifikaci platformy JMS je uvedeno, že ID zprávy musí být nastaveno poskytovatelem rozhraní JMS a že musí být buď jedinečné, nebo musí mít hodnotu null. Přiřadíte-li hodnotu JMS_IBM_MQMD_MSGID, bude tato hodnota zkopírována do hodnoty JMSMessageID. Takže není nastaven poskytovatelem JMS a nemusí být jedinečná: tato hodnota porušuje specifikaci JMS.	Identifikátor zprávy	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci platformy JMS.
JMS_IBM_MQMD_CORRELID **Poznámka:** Přiřadíte-li 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 platformy 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
ROZLIŠUJÍCÍ TOKEN JMS_IBM_MQMD_ACCOUNTINGTOKEN	Token evidence	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci platformy JMS.
JMS_IBM_MQMD_APPLIDENTITYDATA	Údaje o žádosti vztahující se k totožnosti	System.String
JMS_IBM_MQMD_PUTAPPLTYPE	Typ aplikace, která vložila zprávu	System.Int32
JMS_IBM_MQMD_PUTAPPLNAME	Název aplikace, která vložila zprávu	System.String
JMS_IBM_MQMD_PUTDATE	Datum, kdy byla zpráva vložena	System.String

Tabulka 879. Vlastnosti objektu zprávy reprezentující pole MQMD (pokračování)		
Vlastnost	**Popis**	**Typ**
ČAS PUTTIME JMS_IBM_MQMD_PUTTIME	Čas, kdy byla zpráva vložena	System.String
JMS_IBM_MQMD_APPLORIGINDATA	Údaje o žádosti vztahující se k původu	System.String
JMS_IBM_MQMD_GROUPID	Identifikátor skupiny	Bajtové pole **Poznámka:** Použití vlastností bajtového pole ve zprávě porušuje specifikaci platformy JMS.
JMS_IBM_MQMD_MSGSEQNUMBER	Pořadové číslo lokální zprávy v rámci skupiny	System.Int32
POSUNUTÍ JMS_IBM_MQMD_OFFSET	Posunutí dat ve fyzické zprávě od začátku logické zprávy	System.Int32
PARAMETRY JMS_IBM_MQMD_MSGFLAGS	Příznaky zprávy	System.Int32
JMS_IBM_MQMD_ORIGINALLENGTH	Délka původní zprávy	System.Int32

Další podrobnosti viz MQMD .

Příklady

Tento příklad vede k vložení zprávy do fronty nebo tématu s 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 třeba nastavit hodnotu XMSCS_WMQ_MQMD_MESSAGE_CONTEXT. Další informace o použití příkazu XMSCS_WMQ_MQM_MESSAGE_CONTEXT viz Vlastnosti objektu zprávy.

Podobně můžete extrahovat obsah polí MQMD nastavením hodnoty XMSCS_WMQ_MQMD_READ_ENABLED na hodnotu true před přijetím zprávy a poté použitím metod get zprávy, jako je například vlastnost getString. Všechny přijaté vlastnosti jsou jen pro čtení.

Tento příklad vede k hodnotě pole MQMD.ApplIdentityData pole zprávy bylo získáno 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 880. Vlastnosti objektu MessageConsumer

Název vlastnosti	**Popis**
XMSCS_IS_SUBSCRIPTION_MULTICAST	Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí WebSphere MQ Multicast Transport. Tato vlastnost je jen pro čtení.
VÝJIM_PODSÍTY_XMSCS_ISLEABLE_MULTICAST	Označuje, zda jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služby. Tato vlastnost je jen pro čtení.

Postupujte podle části .Vlastnosti prostředí NET pro IMessageConsumer jsou uvedeny v dalších podrobnostech.

Vlastnosti objektu MessageProducer

Přehled vlastností objektu MessageProducer s odkazy na podrobnější referenční informace.

Viz .Vlastnosti prostředí NET IMessageProducer pro více podrobností.

Vlastnosti relace

Přehled vlastností objektu Relace s odkazy na podrobnější referenční informace.

Viz .Další podrobnosti lze najít v tématu NET properties of ISession .

Definice vlastností

Tato sekce obsahuje definici každé vlastnosti objektu.

Každá definice vlastnosti obsahuje následující informace:

- Datový typ vlastnosti.
- Typy objektů, které mají vlastnost
- Pro vlastnost Destination zadejte název, který lze použít v identifikátoru URI (Uniform Resource Identifier).
- Podrobnější popis vlastnosti
- Platné hodnoty vlastnosti
- Výchozí hodnota vlastnosti.

Vlastnosti, jejichž názvy začínají na jednu z následujících předpon, jsou relevantní pouze pro daný 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 produktu 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 produktu WebSphere . Názvy vlastností jsou definovány jako pojmenované konstanty v hlavičkovém souboru `xmsc_wpm.h`.

Nejsou-li ve svých definicích uvedeny jinak, jsou zbývající vlastnosti 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 zprávy XMS.

Není-li ve své definici uvedeno jinak, je každá vlastnost relevantní jak v doménách typu point-to-point, tak i v doméně odběru.

Aplikace může získat a nastavit hodnotu libovolné vlastnosti, pokud tato vlastnost není označena jako jen pro čtení.

JMS_IBM_CHARACTER_SET

Datový typ:
System.Int32

Vlastnost:
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 přepíše jakékoli CCSID uvedené pro cíl vlastností XMSCS_WMQ_CCSID .

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_Encoding

Datový typ:
System.Int32

Vlastnost:
Zpráva

Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do zamýšleného místa určení. Tato vlastnost potlačuje veškerá kódování určená pro místo určení pomocí vlastnosti XMSC_WMQ_ENCODING . Vlastnost uvádí znázornění binárních celých čísel, pakovaných dekadických celých čísel a čísel s pohyblivou řádovou čárkou.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **Encoding** deskriptoru zpráv.

Aplikace může pro nastavení vlastnosti použít následující pojmenované konstanty:

pojmenovaná konstanta	Význam
MQENC_INTEGER_NORMAL	Normální celočíselné kódování
MQENC_INTEGER_REVERSED	Obrácené celočíselné kódování
MQENC_DECIMAL_NORMAL	Normální pakované kódování desetinných čísel
MQENC_DECIMAL_REVERSED	Obrácené pakované kódování desetinných čísel
MQENC_FLOAT_IEEEE_NORMAL	Normální kódování čísel s plovoucí desetinnou čárkou

pojmenovaná konstanta	Význam
MQENC_FLOAT_IEEE_OBRÁCENÝ	Obrácené kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_S390	Kódování čísel s pohyblivou řádovou čárkou z/OS
MQENC_NATIVE	Kódování nativního počítače

Chcete-li vytvořit hodnotu vlastnosti, může aplikace přidat tři z těchto konstant takto:

- Konstanta, jejíž název začíná řetězcem MQENC_INTEGER, určuje reprezentaci binárních celých čísel.
- Konstanta, jejíž název začíná řetězcem MQENC_DECIMAL, určuje reprezentaci pakovaných dekadických celých čísel.
- Konstanta, jejíž název začíná řetězcem MQENC_FLOAT, určuje reprezentaci čísel s pohyblivou řádovou čárkou.

Alternativně může aplikace nastavit vlastnost na hodnotu MQENC_NATIVE, jejíž hodnota závisí na prostředí.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_EXCEPTIONMESSAGE

Datový typ:
Řetězec

Vlastnost:
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 připojuje ke sběrnici pro integraci služeb a přijímá zprávu z místa určení výjimek.

NÁZEV SOUBORU JMS_IBM_EXCEPTION

Datový typ:
Řetězec

Vlastnost:
Zpráva

Název cíle, ve kterém byla zpráva před odesláním zprávy do cíle výjimek.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke sběrnici pro integraci služeb a přijímá zprávu z místa určení výjimek.

JMS_IBM_EXCEPTIONREASON

Datový typ:
System.Int32

Vlastnost:
Zpráva

Kód příčiny označující příčinu, proč byla zpráva odeslána do cíle výjimek.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke sběrnici pro integraci služeb a přijímá zprávu z místa určení výjimek.

JMS_IBM_EXCEPTIONTIMESTAMP

Datový typ:
System.Int64

Vlastnost:
Zpráva

Čas, kdy byla zpráva odeslána do cíle výjimek.

Čas je vyjádřen v milisekundách od 00:00:00 GMT k 1. lednu 1970.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke sběrnici pro integraci služeb a přijímá zprávu z místa určení výjimek.

JMS_IBM_FEEDBACK

Datový typ:
System.Int32

Vlastnost:
Zpráva

Kód, který označuje chování zprávy sestavy.

Platné hodnoty této vlastnosti jsou kódy zpětné vazby a kódy příčiny, které lze zadat v poli **Feedback** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

FORMÁT JMS_IBM_FORMAT

Datový typ:
Řetězec

Vlastnost:
Zpráva

Chování dat aplikace ve zprávě.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **Format** deskriptoru zpráv.

Ve výchozím nastavení 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

Vlastnost:
Zpráva

Označuje, zda je zpráva poslední zprávou ve skupině zpráv.

Nastavte vlastnost na hodnotu true, pokud se jedná o poslední zprávu ve skupině zpráv. Jinak nastavte vlastnost na hodnotu false, nebo vlastnost nenastavujte. Ve výchozím nastavení není vlastnost nastavena.

Hodnota true odpovídá příznaku stavu MQMF_LAST_MSG_IN_GROUP, který lze zadat v poli **MsgFlags** deskriptoru zpráv.

Tato vlastnost je ignorována v doméně publikování/odběru a není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_MSGTYPE

Datový typ:
System.Int32

Vlastnost:
Zpráva

Typ zprávy.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
MQM_DATAGRAM	Zpráva je taková, která nevyžaduje odpověď.
POŽADAVEK MQMT_REQUEST	Zpráva je taková, která vyžaduje odpověď.
MQMT_REPLY	Zpráva je zpráva odpovědi.
SESTAVA MQMT_REPORT	Zpráva je zpráva hlášení.

Tyto hodnoty odpovídají typům zpráv, které lze zadat v poli **MsgType** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_PUTAPPLTYPE

Datový typ:
System.Int32

Vlastnost:
Zpráva

Typ aplikace, která odeslala zprávu.

Platnými hodnotami vlastnosti jsou typy aplikací, které lze zadat v poli **PutApplType** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_PUTDATE

Datový typ:
Řetězec

Vlastnost:
Zpráva

Datum, kdy byla zpráva odeslána.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **PutDate** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

ČAS SOUBORU JMS_IBM_PUTTIME

Datový typ:
Řetězec

Vlastnost:
Zpráva

Čas, kdy byla zpráva odeslána.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které lze zadat v poli **PutTime** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není relevantní, když se aplikace připojuje ke sběrnici pro integraci služeb.

JMS_IBM_REPORT_COA

Datový typ:
System.Int32

Vlastnost:
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	Požadavek 'potvrdit při příchodu' zprávy hlášení, bez dat aplikace z původní zprávy obsažené ve zprávě sestavy.
MQRO_COA_WITH_DATA	Požadavek 'potvrdit při příchodu' zprávy hlášení, s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COA_WITH_FULL_DATA	Požadavek 'potvrdit při příchodu' zprávy hlášení se všemi daty aplikace z původní zprávy, která je zahrnuta ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zpráv. Další informace o těchto volbách viz Sestava (MQLONG).

Ve výchozím nastavení není vlastnost nastavena.

JMS_IBM_REPORT_COD

Datový typ:
System.Int32

Vlastnost:
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	Požadavek 'potvrdit na doručení' zprávy hlášení, bez dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_CED_WITH_DATA	Požadavek 'potvrdit na doručení' zprávy hlášení, s prvními 100 bajty dat aplikace z původní zprávy zahrnuté ve zprávě sestavy.
MQRO_COD_WITH_FULL_DATA	Požadavek 'potvrdit na doručovací' zprávy hlášení se všemi daty aplikace z původní zprávy zahrnuté ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

SKÓDOVÁ_ZPRÁVA JMS_IBM_REPORT_DISCARD_MSG

Datový typ:
System.Int32

Vlastnost:
Zpráva

Požadavek, aby byla zpráva vyřazena, pokud ji nelze doručit do zamýšleného cíle.

Nastavte vlastnost na hodnotu MQRO_DISCARD_MSG na žádost, aby byla zpráva vyřazena, pokud ji nelze doručit do zamýšleného místa určení. Pokud vyžadujete, aby byla zpráva vložena do fronty nedoručených

zpráv nebo byla odeslána do místa určení výjimek, nenastavujte vlastnost. Ve výchozím nastavení není vlastnost nastavena.

Hodnota MQRO_DISCARD_MSG odpovídá volbě sestavy, která může být zadána v poli **Report** deskriptoru zpráv.

VÝJIMKA JMS_IBM_REPORT_EXCEPTION

Datový typ:
System.Int32

Vlastnost:
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
VÝJIMKA MQRO_EXCEPTION	Vyžádejte zprávy o výjimce, a to bez dat aplikace z původní zprávy obsažené ve zprávě sestavy.
MQRO_EXCEPTION_WIT_DATA	Vyžádejte si zprávy o výjimce požadavku s prvními 100 bajty dat aplikace z původní zprávy obsažené ve zprávě sestavy.
MQRO_EXCEPTION_WITH_FULL_DATA	Vyžádat zprávy o zprávě výjimky se všemi daty aplikace z původní zprávy obsažené ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

JMS_IBM_REPORT_EXPIRATION

Datový typ:
System.Int32

Vlastnost:
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	Vyžádejte zprávy o vypršení platnosti požadavku, bez dat aplikace z původní zprávy obsažené ve zprávě sestavy.
MQRO_EXPIRATION_WITH_DATA	Vyžádat zprávy o vypršení platnosti požadavku, s prvními 100 bajty dat aplikace z původní zprávy obsažené ve zprávě sestavy.
MQRO_EXPIRATION_WITH_FULL_DATA	Vyžádat zprávy o vypršení platnosti požadavku se všemi daty aplikace z původní zprávy obsažené ve zprávě sestavy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zpráv.

Ve výchozím nastavení není vlastnost nastavena.

JMS_IBM_REPORT_NAN

Datový typ:
System.Int32

Vlastnost:
Zpráva

Požadavek na zprávy sestavy negativního oznámení na akci.

Nastavte vlastnost na hodnotu MQRO_NAN tak, aby bylo možné požadovat zprávy s oznámením o negativních akcích. Pokud nevyžadujete zprávy sestavy upozornění na negativní akci, nenastavujte tuto vlastnost. Ve výchozím nastavení není vlastnost nastavena.

Hodnota MQRO_NAN odpovídá volbě sestavy, která může být zadána v poli **Report** deskriptoru zpráv.

JMS_IBM_REPORT_PAN

Datový typ:
System.Int32

Vlastnost:
Zpráva

Požadavek na zprávy sestavy pozitivního oznámení na akci.

Nastavte vlastnost na hodnotu MQRO_PAN a vyžádejte si zprávy s oznámením o kladných akcích. Pokud nevyžadujete zprávy sestav s kladným oznámením, nenastavujte vlastnost. Ve výchozím nastavení není vlastnost nastavena.

Hodnota MQRO_PAN odpovídá volbě sestavy, která může být zadána v poli **Report** deskriptoru zpráv.

JMS_IBM_REPORT_PASS_CORREL_ID

Datový typ:
System.Int32

Vlastnost:
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
ID_KOLEKCE_MQRO_PASS_RELACE_	Požadavek, aby identifikátor korelace jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor korelace původní zprávy.
MQRO_COPY_MSG_ID_TO_CORREL_ID	Požadujte, aby korelační identifikátor jakékoli sestavy nebo zprávy odpovědi byl stejný jako identifikátor zprávy původní zprávy.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli **Report** deskriptoru zpráv.

Výchozí hodnota vlastnosti je MQRO_COPY_MSG_ID_TO_CORREL_ID.

JMS_IBM_REPORT_PASS_MSG_ID

Datový typ:
System.Int32

Vlastnost:
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	Požadavek na vygenerování nového identifikátoru zprávy pro každou zprávu nebo zprávu s odpovědí.

Tyto hodnoty odpovídají volbám sestavy, které lze zadat v poli Sestava v deskriptoru zpráv.

Výchozí hodnota vlastnosti je MQRO_NEW_MSG_ID.

JMS_IBM_RETAIN

Datový typ:
System.Int32

Vlastnost:
Zpráva

Nastavení této vlastnosti označuje, správce front bude zprávu považovat za Zachované publikování. Pokud odběratel přijímá zprávy z témat, může okamžitě po přihlášení obdržet další zprávy nad rámec zpráv přijatých v předchozích verzích. Tyto zprávy jsou nepovinnými zachovaným publikováním pro témata přihlášená k odběru. Pro každé téma, které odpovídá odběru, existuje zachované publikování, které je zpřístupněno k doručení odběratelnému odběrateli zpráv.

Hodnota RETAIN_PUBLIKACE je jedinou platnou hodnotou této vlastnosti. Tato vlastnost standardně není nastavena.

Poznámka: Tato vlastnost je relevantní pouze v doméně publikování/odběr.

JMS_IBM_SYSTEM_MESSAGEID

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
Zpráva

Název aplikace, která odeslala zprávu.

Tato vlastnost je definovaná JMS vlastností s názvem JMS JMSXAppID. Další informace o vlastnosti naleznete v příručce *Java Message Service Specification, verze 1.1*.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

POČET DORUČENÍ JMSX_DELIVERY_COUNT

Datový typ:
System.Int32

Vlastnost:
Zpráva

Počet pokusů o doručení zprávy.

Tato vlastnost je definovaná JMS vlastností s názvem JMS JMSXDeliveryCount. Další informace o vlastnosti naleznete v příručce *Java Message Service Specification, verze 1.1*.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

JMSX_GROUPID

Datový typ:
Řetězec

Vlastnost:
Zpráva

Identifikátor skupiny zpráv, do které zpráva patří.

Tato vlastnost je definovaná JMS vlastností s názvem JMS JMSXGroupID. Další informace o vlastnosti naleznete v příručce *Java Message Service Specification, verze 1.1*.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

JMSX_GROUPSEQ

Datový typ:
System.Int32

Vlastnost:
Zpráva

Pořadové číslo zprávy v rámci skupiny zpráv.

Tato vlastnost je definovaná JMS vlastností s názvem JMS JMSXGroupSeq. Další informace o vlastnosti naleznete v příručce *Java Message Service Specification, verze 1.1*.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

ID UŽIVATELE JMSX_ID

Datový typ:
Řetězec

Vlastnost:
Zpráva

Identifikátor uživatele přidružený k aplikaci, která odeslala zprávu.

Tato vlastnost je definovaná JMS vlastností s názvem JMS JMSXUserID. Další informace o vlastnosti naleznete v příručce *Java Message Service Specification, verze 1.1*.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost není platná pro připojení v reálném čase ke zprostředkovateli.

XMSC_ASYNC_EXCEPTIONS

Datový typ:
System.Int32

Vlastnost:
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
Jakákoli výjimka byla zjištěna asynchronně, mimo rozsah volání synchronního volání rozhraní API a všechny přerušené výjimky připojení jsou odeslány do ExceptionListener.

XMSC_ASYNC_EXCEPTIONS_CONNECTIONBROKEN
Do modulu ExceptionListenerse odesílají pouze výjimky označující přerušené připojení. Jakékoli další výjimky, které se vyskytly během asynchronního zpracování, se do modulu ExceptionListenerneohlašují, a proto není aplikace informována o těchto výjimkách.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_ASYNC_EXCEPTIONS_ALL.

XMSC_CLIENT_ID

Datový typ:
Řetězec

Vlastnost:
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 k podpoře trvalých odběrů v doméně publikování/odběru a je ignorován v dvoubodové doméně. Další informace o nastavení identifikátorů klienta naleznete v tématu ConnectionFactories a objekty Connection.

Tato vlastnost není relevantní pro připojení v reálném čase ke zprostředkovateli.

XMSC_CONNECTION_TYPE

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Typ serveru systému zpráv, ke kterému se aplikace připojuje.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_CT_RTT	Připojení v reálném čase ke zprostředkovateli.
XMSC_CT_WMQ	Připojení ke správci front produktu IBM MQ .

Platná hodnota	Význam
XMSC_CT_WPM	Připojení k serveru WebSphere Application Server service integration bus.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_DELIVERY_MODE

Datový typ:
System.Int32

Vlastnost:
Místo určení

Název použitý v identifikátoru URI:

`persistence` (pro místo určení IBM MQ)
`deliveryMode` (pro výchozí místo určení poskytovatele systému zpráv produktu 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 přechodná. Výchozí režim doručení producenta zpráv nebo libovolný režim doručení zadaný v rámci volání Odeslat je ignorován. Je-li cílem fronta IBM MQ , je hodnota atributu fronty *DefPersistence* také ignorována.
XMSC_DELIVERY_PERSISTENT	Zpráva odeslaná do místa určení je trvalá. Výchozí režim doručení producenta zpráv nebo libovolný režim doručení zadaný v rámci volání Odeslat je ignorován. Je-li cílem fronta IBM MQ , je hodnota atributu fronty *DefPersistence* také ignorována.
XMSC_DELIVERY_AS_APP	Zpráva odeslaná do místa určení má režim doručení uvedený v odeslaném volání. Pokud volba Odeslat neuvádí žádný režim doručení, použije se výchozí režim doručení producenta zpráv. Je-li cílem fronta IBM MQ , hodnota atributu fronty *DefPersistence* se ignoruje.
XMSC_DELIVERY_AS_DEST	Je-li cílem fronta produktu IBM MQ , má zpráva vkládaná do fronty režim doručení určený hodnotou atributu fronty *DefPersistence*. Výchozí režim doručení producenta zpráv nebo libovolný režim doručení zadaný v rámci volání Odeslat je ignorován. Pokud cíl není frontou IBM MQ , význam je stejný jako ve formátu XMSCS_DELIVERY_AS_APP.

Výchozí hodnota je XMSCS_DELIVERY_AS_APP.

*XMSC_IC_CASH_*

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_AUTHENTICATION. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_CREDENTIALS

Datový typ:
Řetězec

Vlastnost:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_CREDENTIALS. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_PRINCIPAL

Datový typ:
Řetězec

Vlastnost:
InitialContext

Na základě Java Kontextového rozhraní SECURITY_PRINCIPAL. Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_IC_SECURITY_PROTOCOL

Datový typ:
Řetězec

Vlastnost:
InitialContext

Na základě Java kontextového rozhraní SECURITY_PROTOCOL Tato vlastnost je použitelná pouze pro kontext pojmenování COS.

XMSC_ICURL

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
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 produktu WebSphere MQ Multicast Transport. Jinak je hodnota false.

Tato vlastnost je relevantní pouze pro připojení v reálném čase ke zprostředkovateli.

XMSC_IS_SUBSCRIPTION_RELIABLE_MULTICAST

Datový typ:
System.Boolean

Vlastnost:
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 služeb. Jinak je hodnota false.

Tato vlastnost je relevantní pouze pro připojení v reálném čase ke zprostředkovateli.

XMSC_JMS_MAJOR_VERSION

Datový typ:
System.Int32

Vlastnost:
Data ConnectionMeta

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

Vlastnost:
Data ConnectionMeta

Číslo vedlejší verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.

*XMSC_JMS_*

Datový typ:
Řetězec

Vlastnost:
Data ConnectionMeta

Identifikátor verze specifikace JMS , na které je produkt XMS založen. Tato vlastnost je jen pro čtení.

XMSC_HLAVNÍ_VERZE

Datový typ:
System.Int32

Vlastnost:
Data ConnectionMeta

Číslo verze klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_MINOR_VERSION

Datový typ:
System.Int32

Vlastnost:
Data ConnectionMeta

Číslo vydání klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_PASSWORD

Datový typ:
Bajtové pole

Vlastnost:
ConnectionFactory

Heslo, které lze použít k ověření aplikace při pokusu o připojení k serveru systému zpráv. Heslo se používá s vlastností XMSCS_USERID .

Ve výchozím nastavení není vlastnost nastavena.

Připojujete-li se k produktu IBM MQ v systému Multiplatformsa nastavíte vlastnost XMSCS_USERID továrny připojení, musí se shodovat s hodnotou **userid** přihlášeného uživatele. Pokud tyto vlastnosti nenastavíte, správce front použije při výchozím nastavení **userid** přihlášeného uživatele. Požadujete-li další ověření na úrovni připojení pro jednotlivé uživatele, můžete napsat uživatelskou proceduru pro ověření klienta, která je konfigurována v produktu IBM MQ. Další informace o vytvoření uživatelské procedury pro ověření klienta naleznete v tématu Plánování ověření pro klientskou aplikaci.

Chcete-li ověřit uživatele při připojování k produktu IBM MQ for z/OS , je třeba použít uživatelskou proceduru pro zabezpečení zprávy.

XMSC_PRIORITY

Datový typ:
System.Int32

Vlastnost:
Místo určení

Název použitý v identifikátoru URI:
priority (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 producentu zpráv nebo libovolná priorita zadaná v rámci výzvy k odeslání je ignorována. Je-li cílem fronta IBM MQ , je hodnota atributu fronty **DefPriority** také ignorována.
XMSC_PRIORITY_AS_APP	Zpráva odeslaná do místa určení má prioritu určenou v rámci volání Odeslat. Pokud volání typu odeslání určuje žádnou prioritu, použije se výchozí priorita producenta zpráv. Je-li cílem fronta IBM MQ , hodnota atributu fronty **DefPriority** se ignoruje.

Platná hodnota	Význam
XMSC_PRIORITY_AS_DEST	Je-li cílem fronta IBM MQ , má zpráva uvedená ve frontě prioritu uvedenou v hodnotě atributu fronty **DefPriority**. Výchozí priorita producentu zpráv nebo libovolná priorita zadaná v rámci výzvy k odeslání je ignorována. Pokud cíl není frontou IBM MQ , význam má stejný význam jako hodnota XMSCS_PRIORITY_AS_APP.

Výchozí hodnota je XMSCS_PRIORITY_AS_APP.

WebSphere MQ Real-Time Transport a WebSphere MQ Multicast Transport nepřijímají žádné akce založené na prioritě zprávy.

XMSC_NÁZEV_POSKYTOVATELE

Datový typ:
Řetězec

Vlastnost:
Data ConnectionMeta

Poskytovatel klienta XMS . Tato vlastnost je jen pro čtení.

XMSC_RTC_BROKER_PING_INTERVAL

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Časový interval v milisekundách, po který XMS .NET kontroluje připojení k serveru systému zpráv v reálném čase ke zjištění jakékoli aktivity. Není-li zjištěna žádná aktivita, klient iniciuje příkaz ping; připojení se uzavře, pokud na příkaz ping nebude zjištěna žádná odezva.

Výchozí hodnota vlastnosti je 30000.

XMSC_RTT_CONNECTION_PROTOCOL

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Komunikační protokol použitý pro připojení v reálném čase ke zprostředkovateli.

Hodnota vlastnosti musí být XMSCS_RTT_CP_TCP, což znamená v reálném čase připojení k zprostředkovateli prostřednictvím protokolu TCP/IP. Výchozí hodnota je XMSCS_RTT_CP_TCP.

XMSC_RTT_HOST_NAME

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název nebo adresa IP hostitele systému, na kterém je spuštěn zprostředkovatel.

Tato vlastnost se používá spolu s vlastností XMSCS_RTT_PORT k identifikaci zprostředkovatele.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_RTT_LOKÁLNÍ_ADRESA

Datový typ:
Řetězec

Vlastnost:
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 vy potřebujete mít možnost určit, které rozhraní musí být použito pro připojení v reálném čase. Má-li systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Pokud má systém dvě nebo více síťových rozhraní a tato vlastnost není nastavena, je rozhraní vybráno náhodně.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_RTT_MULTICAST

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory a Destination

Název použitý v identifikátoru URI:
musikasa

Nastavení výběrového vysílání pro továrnu připojení nebo cíl. Tato vlastnost může mít pouze místo určení, které je tématem.

Aplikace používá tuto vlastnost k povolení výběrového vysílání ve spojení se zprostředkovatelem v reálném čase a, je-li povoleno výběrové vysílání, určujete přesný způsob, jakým se výběrové vysílání používá k doručování zpráv od zprostředkovatele ke spotřebiteli zpráv. Vlastnost nemá žádný vliv na způsob, jakým Producent zpráv odesílá zprávy do zprostředkovatele.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_RTT_MULTICAST_DISABLED	Zprávy nejsou doručovány spotřebiteli zpráv pomocí produktu 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 k odběrateli zpráv. Výběrové nastavení multicast pro továrnu připojení je zaznamenáno v době vytvoření připojení. Tato hodnota je platná pouze pro cílový objekt a je výchozí hodnotou pro cílový objekt.
XMSC_RTT_MULTICAST_ENABLED	Je-li téma konfigurováno pro výběrové vysílání ve zprostředkovateli, jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport. Spolehlivá kvalita služeb se používá, je-li téma nakonfigurováno pro spolehlivé výběrové vysílání.

Platná hodnota	Význam
XMSC_RTT_MULTICAST_RELIABLE	Je-li téma nakonfigurováno pro spolehlivé výběrové vysílání ve zprostředkovateli, jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport se spolehlivou kvalitou služeb. Není-li téma konfigurováno pro spolehlivé výběrové vysílání, nemůžete pro dané téma vytvořit spotřebitele zpráv.
XMSC_RTT_MULTICAST_NOT_RELIABLE	Je-li téma konfigurováno pro výběrové vysílání ve zprostředkovateli, jsou zprávy doručovány spotřebiteli zpráv pomocí produktu WebSphere MQ Multicast Transport. Spolehlivá kvalita služeb se nepoužívá ani v případě, že je téma nakonfigurováno pro spolehlivé výběrové vysílání.

XMSC_RTT_PORT

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Číslo portu, na kterém zprostředkovatel naslouchá příchozím požadavkům. Na zprostředkovateli je třeba konfigurovat uzel zpracování zpráv typu Real-timeInput nebo Real-timeOptimizedMessage Flow k naslouchání na tomto portu.

Tato vlastnost se používá s vlastností XMSCS_RTT_HOST_NAME k identifikaci zprostředkovatele.

Výchozí hodnota vlastnosti je XMSCS_RTT_DEFAULT_PORT nebo 1506.

XMSC_TIME_TO_LIVE

Datový typ:
System.Int32

Vlastnost:
Místo určení

Název použitý v identifikátoru URI:

vypršení platnosti (pro místo určení IBM MQ)
timeToLive (pro výchozí místo určení poskytovatele systému zpráv produktu WebSphere)

Doba životnosti zpráv odeslaných do cíle.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
0	Zpráva odeslaná do místa určení nikdy nevyprší platnost.
Kladné celé číslo	Zpráva odeslaná do místa určení má zadaný čas k životu v milisekundách. Výchozí doba života producenta zpráv nebo jakákoli doba, po kterou má být zadán požadavek na odeslání, je ignorována.
XMSC_TIME_TO_LIVE_AS_APP	Zpráva odeslaná do místa určení má čas, který má být zadán při volání pro odeslání. Pokud při odesílání není určen žádný čas k životu, použije se výchozí doba života producenta zpráv.

Výchozí hodnota je XMSCS_TIME_TO_LIVE_AS_APP.

XMSC_ID_UŽIVATELE

Datový typ:
Řetězec

Vlastnost:
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 .

Ve výchozím nastavení není vlastnost nastavena.

Multi Připojujete-li se k produktu IBM MQ for Multiplatformsa nastavíte vlastnost XMSCS_USERID továrny připojení, musí se shodovat s hodnotou **userid** přihlášeného uživatele. Pokud tyto vlastnosti nenastavíte, správce front použije při výchozím nastavení **userid** přihlášeného uživatele. Požadujete-li další ověření na úrovni připojení jednotlivých uživatelů, můžete napsat uživatelskou proceduru pro ověření klienta, která je konfigurována v produktu IBM MQ. Další informace o vytvoření uživatelské procedury pro ověření klienta naleznete v tématu Plánování ověření pro klientskou aplikaci.

z/OS Chcete-li ověřit uživatele při připojování k produktu IBM MQ for z/OS , je třeba použít uživatelskou proceduru pro zabezpečení zprávy.

*XMSC_*

Datový typ:
Řetězec

Vlastnost:
Data ConnectionMeta

Identifikátor verze rozhraní cliXMSent. Tato vlastnost je jen pro čtení.

V 9.2.4 *XMSC_WMQ_BALANCING_APPLICATION_TYPE*

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_BALANCING_APPLICATION_TYPE_SIMPLE	Jednoduché vyvážení; kromě těch, které jsou popsány v tématu Ovlivňování vyrovnávání aplikací v uniformách klastrů, se neuplatňují žádná specifická pravidla. Toto je výchozí hodnota.
XMSC_WMQ_BALANCING_APPLICATION_TYPE_REQUEST_REPLY	Vyrovnávání požadavek-odpověď; po každém volání MQPUT se očekává odpovídající volání MQGET pro zprávu odezvy. Vyvažování je zpožděno, dokud taková zpráva nebyla přijata, nebo byla překročena žádost o prodloužení platnosti požadavku.

Kromě toho lze tuto vlastnost nastavit v souboru `client.ini` . Pořadí předvoleb je:

1. Vlastnosti nastavené v aplikaci
2. Shoda s názvem stanza aplikace v souboru `mqclient.ini` .
3. Stanza výchozích nastavení aplikace v souboru `mqclient.ini` .

V 9.2.4 *XMSC_WMQ_BALANCING_OPTIONS*

Datový typ:
System.Int32

Vlastnost:
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 nové vyvážení aplikací i v případě, že je transakce uprostřed transakce.

Kromě toho lze tuto vlastnost nastavit v souboru `client.ini` . Pořadí předvoleb je:

1. Vlastnosti nastavené v aplikaci
2. Shoda s názvem stanza aplikace v souboru `mqclient.ini` .
3. Stanza výchozích nastavení aplikace v souboru `mqclient.ini` .

V 9.2.4 *XMSC_WMQ_BALANCING_TIMEOUT*

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_BALANCING_TIMEOUT_IMMEDIATE	Okamžitý časový limit
XMSC_WMQ_BALANCING_TIMEOUT_AS_DEFAULT	Výchozí hodnota časového limitu pro nastavení. Toto je výchozí hodnota
XMSC_WMQ_BALANCING_TIMEOUT_NEVER	Bez časového limitu

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 stanza aplikace v souboru `mqclient.ini` .
3. Stanza výchozích nastavení aplikace v souboru `mqclient.ini` .

XMSC_WMQ_BROKER_CONTROLQ

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
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

Vlastnost:
ConnectionFactory

Název správce front, ke kterému je zprostředkovatel připojen.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_SUBQ

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název fronty odběratele pro spotřebitele netrvalých zpráv.

Název fronty odběratele musí začínat následujícími znaky:

SYSTEM.JMS.ND.

Chcete-li, aby všichni odběratelé netrvalých zpráv sdíleli frontu odběratele, zadejte úplný název sdílené fronty. Fronta s uvedeným názvem musí existovat předtím, než aplikace může vytvořit netrvalý spotřebitel zpráv.

Chcete-li, aby každý spotřebitel zpráv netrvalých zpráv načítal zprávy ze své vlastní výlučné fronty odběratele, zadejte název fronty, který končí hvězdičkou (*). Poté, když aplikace vytvoří netrvanlivý spotřebitel zpráv, klient XMS vytvoří dynamickou frontu pro výhradní použití odběratelem zprávy. 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 při výchozím nastavení používá přístup prostřednictvím sdílené fronty.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_BROKER_VERSION

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory a Destination

Název použitý v identifikátoru URI:
brokerVersion

Typ zprostředkovatele použitý aplikací pro připojení nebo pro cíl. Tato 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 a odběru IBM MQ . Aplikace může tuto hodnotu také použít, pokud migrujete z IBM MQ publish/subscribe to WebSphere Message Broker , ale neměnili jste aplikaci.
XMSC_WMQ_BROKER_V2	Aplikace používá zprostředkovatele produktu IBM Integration Bus.
XMSC_WMQ_BROKER_UNSPECIFIED	Po migraci zprostředkovatele nastavte tuto vlastnost tak, aby se záhlaví RFH2 již nadále nepoužívala. Po migraci tato vlastnost již není relevantní.

Výchozí hodnota pro továrnu připojení je XMSCS_WMQ_BROKER_UNSPECIFIED, ale při výchozím nastavení není vlastnost nastavena pro místo určení. Nastavení této vlastnosti pro cíl přepíše jakoukoli hodnotu zadanou vlastností továrny připojení.

XMSC_WMQ_CCDTURL

Datový typ:
System.String

Vlastnost:
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.

Při výchozím nastavení tato vlastnost není nastavena.

XMSC_WMQ_CCSID

Datový typ:
System.Int32

Vlastnost:
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 pro jednotlivou zprávu nastavena vlastnost JMS_IBM_CHARACTER_SET , potlačí vlastnost uvedenou pro cíl touto vlastností vlastnost CCSID.

Výchozí hodnota vlastnosti je 1208.

Tato vlastnost je relevantní pouze pro zprávy odeslané do místa určení, nikoli na zprávy přijaté z cíle.

XMSC_WMQ_CHANNEL

Datový typ:
Řetězec

Vlastnost:
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.

Ve výchozím nastavení 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

Vlastnost:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CLIENTRECONECOPTIONS

Krátký název nástroje pro administraci JMS: CROPT

Tato vlastnost uvádí volby nového připojení klienta pro nová připojení vytvořená touto továrnou. Je nalezen v XMSC a je jedním z:

- WMQ_CLIENT_RECONNECT_AS_DEF (výchozí). Použijte hodnotu zadanou v souboru `mqclient.ini` . Nastavte tuto hodnotu pomocí vlastnosti **DefRecon** v sekci Kanály. Může být nastavena na jednu z následujících možností:
  1. Ano. Behaves jako volba WMQ_CLIENT_RECONNECT
  2. -NE. Předvolba. Neuvádí žádné volby opětovného připojení
  3. QMGR. Behaves jako volba WMQ_CLIENT_RECONNECT_Q_MGR
  4. Zakázáno. Behaves jako volba WMQ_CLIENT_RECONNECT_DISABLED
- WMQ_CLIENT_RECONNECT. Připojte se k libovolnému správci front uvedenému v seznamu názvů připojení.
- WMQ_CLIENT_RECONNECT_Q_MGR. Znovu naváže spojení se stejným správcem front, ke kterému je původně připojen. Vrací `MQRC_RECONNECT_QMID_MISMATCH` , pokud se správce front, ke kterému se pokouší připojit (uvedený v seznamu názvů připojení), liší od QMID ke správci front, k němuž se původně připojil.
- WMQ_CLIENT_RECONNECT_DISABLED. Opětovné připojení je zakázáno.

XMSC_WMQ_CLIENT_RECONNECT_TIMEOUT

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CLIENTRECONNECTSIMEOUT

Krátký název nástroje pro administraci JMS: CRT

Vlastnost XMSCS_WMQ_CLIENT_RECONT_TIMEOUT je platná pouze pro spravovaného klienta XMS .NET .

Tato vlastnost určuje dobu trvání (v sekundách), po kterou se připojení klienta pokouší znovu navázat spojení.

Po pokusu o nové připojení po tuto dobu dojde k selhání klienta pomocí funkce MQRC_RECONIPRI_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

Vlastnost:
ConnectionFactory

Režim, pomocí kterého se aplikace připojuje ke správci front.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_CM_VAZEB	Pro optimální výkon se používá připojení ke správci front v režimu vazeb. Tato hodnota je výchozí hodnotou pro C/C + +.
XMSC_WMQ_CM_KLIENT	Připojení ke správci front v režimu klienta za účelem zajištění plně spravovaného zásobníku. Tato hodnota je výchozí hodnotou pro .NET.
XMSCS_WMQ_CM_CLIENT_UNMANAGED (pouze pro .NET)	Připojení ke správci front, který vynutí vytvoření nespravovaného zásobníku klienta.

XMSC_WMQ_CONNECTION_NAME_LIST

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Použitelné objekty:
Dlouhý název nástroje pro administraci JMS: CONNECTIONNAMELIST

Krátký název nástroje pro administraci JMS: CNLIST

Tato vlastnost určuje hostitele, ke kterému 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.

Příklad:`127.0.0.1 (1414),host2.example.com(1400)`

Výchozí nastavení této vlastnosti je `localhost (1414)`.

XMSC_WMQ_DUR_SUBQ

Datový typ:
Řetězec

Vlastnost:
Místo určení

Název fronty odběratele pro trvalého odběratele, který přijímá zprávy z cíle. Tato vlastnost může mít pouze místo určení, které je tématem.

Název fronty odběratele musí začínat následujícími znaky:

SYSTEM.JMS.D.

Chcete-li, aby všechny trvalé odběratele sdílely frontu odběratele, zadejte úplný název sdílené fronty. Než může aplikace vytvořit trvalého odběratele, musí existovat fronta s určeným názvem.

Chcete-li, aby každý trvalý odběratel načítal zprávy ze své vlastní výlučné fronty odběratele, zadejte název fronty, který končí hvězdičkou (*). Poté, když aplikace vytvoří trvalého odběratele, vytvoří klient produktu XMS 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 při výchozím nastavení používá přístup prostřednictvím sdílené fronty.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_ENCODING

Datový typ:
System.Int32

Vlastnost:
Místo určení

Způsob reprezentace číselných dat v těle zprávy v případě, že klient produktu XMS předá zprávu do místa určení. If set for an individual message, the KÓDOVÁNÍ JMS_IBM_ property overrides the encoding specified for the destination by this property. Vlastnost uvádí znázornění binárních celých čísel, pakovaných dekadických celých čísel a čísel s pohyblivou řádovou čárkou.

Platné hodnoty vlastnosti jsou stejné jako hodnoty, které mohou být zadány v poli **Encoding** deskriptoru zpráv.

Aplikace může pro nastavení vlastnosti použít následující pojmenované konstanty:

pojmenovaná konstanta	Význam
MQENC_INTEGER_NORMAL	Normální celočíselné kódování
MQENC_INTEGER_REVERSED	Obrácené celočíselné kódování
MQENC_DECIMAL_NORMAL	Normální pakované kódování desetinných čísel
MQENC_DECIMAL_REVERSED	Obrácené pakované kódování desetinných čísel
MQENC_FLOAT_IEEE_NORMAL	Normální kódování čísel s plovoucí desetinnou čárkou
MQENC_FLOAT_IEEE_OBRÁCENÝ	Obrácené kódování čísel s pohyblivou řádovou čárkou IEEE
MQENC_FLOAT_S390	Kódování čísel s pohyblivou řádovou čárkou z/OS
MQENC_NATIVE	Kódování nativního počítače

Chcete-li vytvořit hodnotu vlastnosti, může aplikace přidat tři z těchto konstant takto:

- Konstanta, jejíž název začíná řetězcem MQENC_INTEGER, určuje reprezentaci binárních celých čísel.
- Konstanta, jejíž název začíná řetězcem MQENC_DECIMAL, určuje reprezentaci pakovaných dekadických celých čísel.
- Konstanta, jejíž název začíná řetězcem MQENC_FLOAT, určuje reprezentaci čísel s pohyblivou řádovou čárkou.

Alternativně může aplikace nastavit vlastnost na hodnotu MQENC_NATIVE, jejíž hodnota závisí na prostředí.

Výchozí hodnota vlastnosti je MQENC_NATIVE.

Tato vlastnost je relevantní pouze pro zprávy odeslané do místa určení, nikoli na zprávy přijaté z cíle.

XMSC_WMQ_FAIL_IF_QUIESCE

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory a Destination

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 selže, pokud se správce front nachází ve stavu uvedení do klidového stavu. Když aplikace zjistí, že správce front je uváděn do klidového stavu, může aplikace dokončit její okamžitou úlohu a zavřít připojení, což umožňuje zastavení správce front.
XMSC_WMQ_FIQ_NO	Žádná volání metody se nezdařila, protože správce front je ve stavu uvedení do klidového stavu. Zadáte-li tuto hodnotu, nebude moci aplikace zjistit, zda je správce front uváděn do klidového stavu. Aplikace může pokračovat v provádění operací se správcem front, a zabránit tak zastavení správce front.

Výchozí hodnota továrny připojení je XMSCS_WMQ_FIQ_YES, ale při výchozím nastavení není vlastnost nastavena pro místo určení. Nastavení této vlastnosti pro cíl přepíše jakoukoli hodnotu zadanou vlastností továrny připojení.

XMSC_WMQ_MESSAGE_BODY

Datový typ:
System.Int32

Vlastnost:
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 nahradí vlastnost XMSCS_WMQ_MESSAGE_BODY existující vlastnost XMS Destination property XMSC_WMQ_TARGET_CLIENT.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_MESSAGE_BODY_JMS
Přijmout: Příchozí typ zprávy XMS a tělo jsou určeny obsahem MQRFH2 (pokud existuje) nebo MQMD (pokud v přijaté zprávě IBM MQ není MQRFH2).

Odeslat: Tělo odchozí zprávy produktu XMS obsahuje předřazené a automaticky generované záhlaví MQRFH2 založené na vlastnostech zprávy a polích záhlaví zprávy XMS .

XMSC_WMQ_MESSAGE_BODY_MQ
Přijmout: Příchozí typ zprávy XMS je vždy ByteMessage, bez ohledu na obsah přijaté zprávy IBM MQ nebo v poli formátu pro přijatý MQMD. Tělo zprávy produktu XMS je nezměněná data zprávy vrácená voláním rozhraní API poskytovatele systému zpráv. Znaková sada a kódování dat v těle zprávy je určeno poli CodedCharSetId a zakódováním v rámci polí 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 automaticky generované záhlaví IBM MQ .

XMSC_WMQ_MESSAGE_BODY_UNSPECIFIED
Přijmout: Klient XMS určí vhodnou hodnotu pro tuto vlastnost. V případě cesty příjmu je tato hodnota hodnotou vlastnosti WMQ_MESSAGE_BODY_JMS.

Odeslat: Klient XMS určí vhodnou hodnotu pro tuto vlastnost. Na cestě k odeslání je tato hodnota hodnotou vlastnosti XMSCS_WMQ_TARGET_CLIENT.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ_MESSAGE_BODY_UNSPECIFIED.

XMSC_WMQ_MQMD_MESSAGE_CONTEXT

Datový typ:
System.Int32

Vlastnost:
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_MDCEX_DEFAULT
Pro odchozí zprávy volání MQOPEN rozhraní API a struktura MQPMO neurčují žádné explicitní volby kontextu zprávy.

XMSC_WMQ_MDCTX_SET_IDENTITY_CONTEXT
Volání rozhraní MQOPEN API určuje volbu kontextu zprávy MQOO_SET_IDENTITY_CONTEXT a struktura MQPMO určuje hodnotu MQPMO_SET_IDENTITY_CONTEXT.

XMSC_WMQ_MDCTX_SET_ALL_CONTEXT
Volání rozhraní MQOPEN API uvádí volbu kontextu zprávy MQOO_SET_ALL_CONTEXT a struktura MQPMO určuje MQPMO_SET_ALL_CONTEXT.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ_MDCTX_DEFAULT.

Poznámka: Tato vlastnost není relevantní, když se aplikace připojuje k produktu WebSphere Application Server service integration bus.

Následující vlastnosti vyžadují vlastnost XMSCS_WMQ_MQM_MESSAGE_CONTEXT, která má být nastavena na hodnotu vlastnosti XMSCS_WMQ_MDCT_X_SET_IDENTITY_CONTEXT nebo vlastnost XMSCS_WMQ_MDCTX_SET_ALL_CONTEXT při odesílání zprávy za účelem dosažení požadovaného účinku:

- JMS_IBM_MQMD_USERIDENTIFIER
- ROZLIŠUJÍCÍ TOKEN JMS_IBM_MQMD_ACCOUNTINGTOKEN
- JMS_IBM_MQMD_APPLIDENTITYDATA

Následující vlastnosti vyžadují, aby vlastnost XMSCS_WMQ_MQMD_MESSAGE_CONTEXT byla nastavena na hodnotu vlastnosti XMSCS_WMQ_MDCTX_SET_ALL_CONTEXT při odesílání zprávy za účelem dosažení požadovaného efektu:

- JMS_IBM_MQMD_PUTAPPLTYPE
- JMS_IBM_MQMD_PUTAPPLNAME
- JMS_IBM_MQMD_PUTDATE
- ČAS PUTTIME JMS_IBM_MQMD_PUTTIME
- JMS_IBM_MQMD_APPLORIGINDATA

XMSC_WMQ_MQMD_READ_ENABLED

Datový typ:
System.Int32

Vlastnost:
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* v odeslané zprávě aktualizovány tak, aby odrážely aktualizované hodnoty polí v produktu MQMD.

Při příjmu zpráv nejsou žádné z vlastností JMS_IBM_MQMD* k dispozici v přijaté zprávě, i když některé nebo všechny z nich jsou nastaveny odesilatelem.

XMSC_WMQ_READ_ENABLED_YES
Při odesílání zpráv jsou všechny vlastnosti JMS_IBM_MQMD* v odeslané zprávě aktualizovány tak, aby odrážely aktualizované hodnoty polí v MQMD, včetně vlastností, které odesilatel explicitně nenastavili.

Při příjmu zpráv jsou všechny vlastnosti JMS_IBM_MQMD* k dispozici v přijaté zprávě, včetně vlastností, které odesilatel explicitně nenastavili.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ_READ_ENABLED_NO.

XMSC_WMQ_MQMD_WRITE_ENABLED

Datový typ:
System.Int32

Vlastnost:
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 se nezkopírují do základní struktury MQMD.

XMSC_WMQ_WRITE_ENABLED_YES
Vlastnosti JMS_IBM_MQMD* jsou zpracovány. Jejich hodnoty jsou zkopírovány do podkladové struktury MQMD.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ_WRITE_ENABLED_NO.

XMSC_WMQ_PUT_ASYNC_ALLOWED

Datový typ:
System.Int32

Vlastnost:
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
Určete, zda jsou asynchronní operace vložení povoleny, odkazem na definici fronty nebo tématu.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_Q_DEF
Určete, zda jsou asynchronní operace vložení povoleny odkazem na definici fronty.

XMSC_WMQ _PUT_ASYNC_ALLOWED_AS_TOPIC_DEF
Určete, zda jsou asynchronní operace vložení povoleny odkazem na definici tématu.

XMSC_WMQ _PUT_ASYNC_ALLOWED_DISABLED
Asynchronní operace put nejsou povoleny.

XMSC_WMQ _PUT_ASYNC_ALLOWED_ENABLED
Asynchronní operace put jsou povoleny.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_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

Vlastnost:
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 s odkazem na definici fronty.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_ TOPIC _DEF
Určete, zda je dopředné čtení povoleno s odkazem na definici tématu.

XMSC_WMQ_READ_AHEAD_ALLOWED_AS_DEST
Určete, zda je dopředné čtení povoleno s odkazem na definici fronty nebo tématu.

XMSC_WMQ_READ_AHEAD_ALLOWED_DISABLED
Čtení napřed není povoleno při přijímání nebo procházení zpráv.

XMSC_WMQ_READ_AHEAD_ALLOWED_ENABLED
Čtení napřed je povoleno.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ _READ_AHEAD_ALLOWED_AS_DEST.

XMSC_WMQ_READ_AHEAD_CLOSE_POLICY

Datový typ:
System.Int32

Vlastnost:
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.

Tato vlastnost je použitelná při zadávání uzavírajících voleb fronty při přijímání zpráv z místa určení a nepoužitelné při odesílání zpráv do místa určení.

Tato vlastnost je u prohlížečů fronty 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 bude dokončeno pouze aktuální vyvolání modulu listener pro zprávy, případně zanechání zpráv v interní 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 před návratem doručeny do modulu listener pro zprávy aplikace.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ _READ_AHEAD_CLOSE_POLICY_DELIVER_CURRENT.

Poznámka:

Abnormální ukončení aplikace
Všechny zprávy v vyrovnávací paměti dopředného čtení se ztratí při náhlém ukončení aplikace XMS .

Důsledky pro transakce
Čtení napřed je vypnuto, když aplikace používají transakce. Aplikace tak nevidí žádný rozdíl v chování, když používají relace zahrnující transakci.

Implikace režimů scknowledgement relace
Čtení napřed je povoleno na netransakčních relacích, když jsou režimy potvrzení buď XMSCS_AUTO_ACKNOWLEDGE nebo XMSCS_DUPS_OK_ACKNOWLEDGE. Čtení napřed je vypnuto, pokud je režim potvrzení relace XMSCS_CLIENT_ACKNOWLEDGE, bez ohledu na relace s transakcemi nebo bez transakcí.

Implikace pro prohlížeče front a selektory prohlížeče front
Prohlížeče front a selektory prohlížeče fronty použité v aplikacích produktu XMS mají výhodu v předstihu před čtením. Zavření prohlížeče fronty nesnižuje výkon, protože zpráva je stále k dispozici ve frontě pro další operace. Pro prohlížeče front a selektory prohlížečů front nejsou žádné další důsledky, kromě předností výkonu čtení napřed.

XMSC_WMQ_NÁZEV_HOSTITELE

Datový typ:
Řetězec

Vlastnost:
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. Vlastnost se používá spolu s vlastností XMSCS_WMQ_PORT k identifikaci správce front.

Výchozí hodnota vlastnosti je `localhost`.

XMSC_WMQ_LOCAL_ADDRESS

Datový typ:
Řetězec

Vlastnost:
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 má být použito pro připojení.

Poskytnutí těchto informací je nezbytné pouze v případě, že systém, na kterém je aplikace spuštěna, má dvě nebo více síťových rozhraní a vy potřebujete mít možnost určit, které rozhraní musí být pro připojení použito. Má-li systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Má-li systém dvě nebo více síťových rozhraní a nespecifikujete, které rozhraní musí být použito, je rozhraní vybráno náhodně.

nízká_port
Číslo lokálního portu, které má být použito pro připojení.

Je-li zadána také hodnota *high_port* , hodnota *low_port* je interpretována jako nejnižší číslo portu v rozsahu čísel portů.

vysoká_port
Nejvyšší číslo portu v rozsahu čísel portů. Jeden z portů v uvedeném rozsahu musí být použit pro připojení.

Maximální délka řetězce je 48 znaků.

Zde je několik příkladů platných hodnot vlastnosti:

```
PLANET NAME
9.20.4.98
JUPER (1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)
```

Ve výchozím nastavení 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

Vlastnost:
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 XMSCS_WMQ_MSEL_CLIENT.

Tato vlastnost je relevantní pouze v doméně publikování/odběru. Výběr zpráv zprostředkovatelem není podporován, je-li vlastnost XMSC_WMQ_BROKER_VERSION nastavena na hodnotu XMSC_WMQ_BROKER_V1.

XMSC_WMQ_MSG_BATCH_SIZE

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Maximální počet zpráv, které mají být načteny z fronty v jedné dávce, v případě asynchronního doručování zpráv.

Když aplikace používá asynchronní doručování zpráv za určitých podmínek, klient XMS načte dávku zpráv z fronty před odesláním každé zprávy jednotlivě do aplikace. Tato vlastnost určuje maximální počet zpráv, které mohou být v dávce.

Hodnota vlastnosti je kladné celé číslo a výchozí hodnota je 10. Zvažte nastavení této vlastnosti na jinou hodnotu pouze v případě, že máte specifický problém s výkonem, který je třeba adresovat.

Je-li aplikace připojena ke správci front v síti, může zvýšení hodnoty této vlastnosti snížit režii sítě a doby odezvy, ale zvýšit množství paměti potřebné k ukládání zpráv v systému klienta. Naopak snížení hodnoty této vlastnosti může zvýšit režii sítě a doby odezvy, ale sníží množství paměti potřebné k uložení zpráv.

XMSC_WMQ_POLLING_INTERVAL

Datový typ:
System.Int32

Vlastnost:
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 listenerů zpráv v rámci relace 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

Vlastnost:
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. Vlastnost se používá s vlastností XMSCS_WMQ_HOST_NAME k identifikaci správce front.

Výchozí hodnota vlastnosti je XMSCS_WMQ_DEFAULT_CLIENT_PORT nebo 1414.

XMSC_WMQ_PROVIDER_VERSION

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Verze, vydání, úroveň modifikace a opravný balík správce front, ke kterému se tato aplikace hodlá připojovat. Platné hodnoty této vlastnosti 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.

Hodnota 7 nebo vyšší označuje, že tato verze je určena jako IBM WebSphere MQ 7.0 ConnectionFactory pro připojení ke správci front IBM WebSphere MQ 7.0 . Hodnota, která je starší než 7 (například "6.0.2.0"), označuje, že je určen pro použití se správci front dříve než verze 7.0. Výchozí hodnota, nespecifikovaná, povoluje připojení k jakékoli úrovni správce front, určení použitelných vlastností a funkcí dostupných na základě schopností správce front.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu "unspecified".

Poznámka:

- Pokud je hodnota XMSCS_WMQ_PROVIDER_VERSION nastavena na hodnotu 6, nedojde k žádné sdílení soketu. 2. PRO
- Selhání připojení, je-li parametr XMSCS_WMQ_PROVIDER_VERSION nastaven na hodnotu 7 a na serveru SHARECNV pro kanál je nastavena hodnota 0.
- Specifické funkce produktu IBM WebSphere MQ 7.0 jsou zakázány, je-li hodnota XMSCS_WMQ_PROVIDER_VERSION nastavena na hodnotu UNSPECIFIED a hodnota SHARECNV je nastavena na hodnotu 0.

Verze klienta produktu IBM MQ také hraje hlavní roli v tom, zda klientská aplikace XMS může používat specifické funkce produktu IBM WebSphere MQ 7.0 . V následující tabulce je popsáno chování.

Poznámka: Vlastnost systému XMSCS_WMQ_OVERRIDEPROVIDERVERSION potlačuje vlastnost XMSCS_WMQ_PROVIDER_VERSION. Tuto vlastnost lze použít v případě, že nelze změnit nastavení faktorie připojení.

Tabulka 881. Klient XMS -Schopnost použít specifické funkce produktu IBM WebSphere MQ 7.0 .

#	XMSC_WMQ_PROVIDER_VERSION	IBM MQ Verze klienta	IBM WebSphere MQ 7.0 funkce
1	nespecifikováno	7	ZAP
2	nespecifikováno	6	VYP
3	7	7	ZAP
4	7	6	Výjimka
5	6	6	VYP
6	6	7	VYP

XMSC_WMQ_PUB_ACK_INTERVAL

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Počet zpráv publikovaných vydavatelem, než klient XMS vyžádá potvrzení od zprostředkovatele.

Snížíte-li hodnotu této vlastnosti, bude klient vyžadovat potvrzení příjmu častěji, a proto se výkon vydavatele sníží. 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

Vlastnost:
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.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WMQ_RECEIVE_CCSID

Vlastnost místa určení, která nastavuje cílové CCSID pro převod zpráv správce front. Hodnota je ignorována, pokud není hodnota XMSCS_WMQ_RECEIVE_CONVERSION nastavena na hodnotu WMQ_RECEIVE_CONVERSION_QMGR.

Datový typ:
Celé číslo

Hodnota:
Libovolné kladné celé číslo.

Výchozí hodnota je 1208.

Zadání hodnoty `GMO_CONVERT` ve zprávě je volitelné. Je-li zadána hodnota `GMO_CONVERT` , bude provedena konverze podle zadané hodnoty.

XMSC_WMQ_RECEIVE_CONVERSION

Cílová vlastnost, která určuje, zda bude převod dat prováděn správcem front.

Datový typ:
Celé číslo

Hodnoty:

XMSCS_WMQ_RECEIVE_CONVERSION_CLIENT_MSG (DEFAULT): Provedení konverze dat pouze na klientu XMS . Převod se vždy provádí s použitím kódové stránky 1208.

XMSCS_WMQ_RECEIVE_CONVERSION_QMGR: Provedení konverze dat na správci front před odesláním zprávy klientovi XMS .

XMSC_WMQ_RECEIVE_EXIT

Datový typ:

Řetězec

Vlastnost:

ConnectionFactory

Identifikuje uživatelskou proceduru pro přijetí zprávy kanálu, která má být spuštěna.

Hodnota vlastnosti je řetězec, který identifikuje uživatelskou proceduru pro přijetí zprávy kanálu a má následující formát:

libraryName(entryPointNázev)

kde:

- **libraryName** je úplná cesta ke spravovanému ukončení `.dll`
- `entryPointName` je název třídy kvalifikované oborem názvů

Například: `C:\MyReceiveExit.dll(MyReceiveExitNameSpace.MyReceiveExitClassName)`

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném režimu klienta. Podporovány jsou také pouze spravované ukončení.

XMSC_WMQ_RECEIVE_EXIT_INIT

Datový typ:

Řetězec

Vlastnost:

ConnectionFactory

Uživatelská data, která jsou předána uživatelské proceduře pro přijetí zprávy kanálu při volání.

Hodnota vlastnosti je řetězec. Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném klientském režimu a je nastavena vlastnost “XMSC_WMQ_RECEIVE_EXIT” na stránce 2063 .

XMSC_WMQ_RESOLVE_QUEUE_MANAGER

Datový typ:

Řetězec

Vlastnost:

ConnectionFactory

Tato vlastnost se používá k získání názvu správce front, ke kterému je připojen.

Je-li použit s tabulkou CCDT (Client Channel Definition Table), může se tento název lišit od názvu správce front určeného v Továrně připojení.

XMSC_WMQ_RESOLVE_QUEUE_MANAGER_ID

Datový typ:

Řetězec

Vlastnost:

ConnectionFactory

Tato vlastnost je naplněna identifikátorem ID správce front po připojení.

XMSC_WMQ_SECURITY_EXIT

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Identifikuje uživatelskou proceduru pro zabezpečení zprávy kanálu.

Hodnota vlastnosti je řetězec, který identifikuje uživatelskou proceduru zabezpečení kanálu a má následující formát:

libraryName(entryPointNázev)

kde:

- **libraryName** je úplná cesta ke spravované uživatelské knihovně DLL.
- entryPointName je název třídy kvalifikované oborem názvů

Příklad: C:\MySecurityExit.dll(MySecurityExitNameSpace.MySecurityExitClassName)

Maximální délka řetězce je 128 znaků.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném režimu klienta. Podporovány jsou také pouze spravované ukončení.

XMSC_WMQ_SECURITY_EXIT_INIT

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.

Maximální délka řetězce uživatelských dat je 32 znaků.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném klientském režimu a je nastavena vlastnost “XMSC_WMQ_SECURITY_EXIT” na stránce 2064 .

XMSC_WMQ_SEND_EXIT

Datový typ:
Řetězec

Vlastnost:
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 spravované uživatelské knihovně DLL.
- entryPointName je název třídy kvalifikované oborem názvů

Například C:\MySendExit.dll(MySendExitNameSpace.MySendExitClassName)

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném režimu klienta. Podporovány jsou také pouze spravované ukončení.

XMSC_WMQ_SEND_EXIT_INIT

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.

Hodnota vlastnosti je řetězec jedné nebo více položek dat uživatele oddělených čárkami. Ve výchozím nastavení není vlastnost nastavena.

Pravidla pro uvedení uživatelských dat, která jsou předávána posloupnosti uživatelských procedur odeslání kanálu, jsou stejná jako pravidla pro uvedení uživatelských dat, která jsou předána posloupnosti uživatelských procedur příjmu kanálu. Z tohoto důvodu viz “XMSC_WMQ_RECEIVE_EXIT_INIT” na stránce 2063.

Tato vlastnost je relevantní pouze v případě, že se aplikace připojuje ke správci front ve spravovaném klientském režimu a je nastavena vlastnost “XMSC_WMQ_SEND_EXIT” na stránce 2064 .

XMSC_WMQ_SEND_CHECK_COUNT

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Počet povolených odeslaných volání mezi kontrolou asynchronních chyb vložení během jedné netransakční relace XMS.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu 0.

XMSC_WMQ_SHARE_CONV_ALLOWED

Datový typ:
System.Int32

Vlastnost:
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 se základní sdílený soket sdílel. Neoznačuje, kolik připojení sdílí jeden soket. Počet připojení sdílejících soket je určen hodnotou SHARECNV vyjednanou mezi klientem IBM MQ a serverem IBM MQ .

Aplikace může nastavit následující pojmenované konstanty pro nastavení vlastnosti:

- XMSC_WMQ_SHARE_CONV_ALLOWED_FALSE-Připojení nesdílí soket.
- XMSCS_WMQ_SHARE_CONV_ALLOWED_TRUE-Připojení sdílí soket.

Při výchozím nastavení je vlastnost nastavena na hodnotu XMSCS_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

Vlastnost:
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 adresa URL má následující formát:

```
[user[/password]@]ldap://[serveraddress][:portnum][,...]
```

Tento formát je kompatibilní s základním formátem MQJMS, ale je tento formát rozšířen.

Je platné mít prázdný `serveraddress`. V tomto případě 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 produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Ve výchozím nastavení není vlastnost nastavena.

Související pojmy
Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET
Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_CIPHER_SPEC

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název specifikace CipherSpec, která má být použita v zabezpečeném připojení ke správci front.

Specifikace šifer, které můžete použít s podporou TLS IBM MQ , jsou vypsány v následující tabulce. Požadujete-li osobní certifikát, určíte velikost klíče pro dvojici veřejný a soukromý klíč. Velikost klíče, která se použije během navázání komunikace přes zabezpečení SSL, je velikost uložená v certifikátu, pokud není určena CipherSpec, jak je uvedeno v tabulce. Při výchozím nastavení tato vlastnost není nastavena.

Název specifikace šifrování	Použitý protokol	Algoritmus přepočtu klíče	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Suite B 128 bitů	Suite 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

Název specifikace šifrování	Použitý protokol	Algoritmus přepočtu klíče	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Suite B 128 bitů	Suite B 192 bitů
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
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	Algoritmus přepočtu klíče	Šifrovací algoritmus	Šifrování bitů	FIPS[1]	Suite B 128 bitů	Suite B 192 bitů

Notes:

1. Uvádí, zda CipherSpec vyhovuje standardu FIPS (Federal Information Processing Standards) 140-2. Vysvětlení FIPS a informace o tom, jak nakonfigurovat produkt IBM MQ for FIPS 140-2 compliant, viz Federal Information Processing Standards (FIPS).
2. Tuto CipherSpec nelze použít k zabezpečení připojení ze správce front IBM MQ Explorer ke správci front, pokud nejsou použity odpovídající soubory neomezených zásad na prostředí JRE použité 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 této chybě vyhnout, buď se vyhnete použití trojitého DES (který je zamítnut), nebo povolte reset tajného klíče při použití této CipherSpec v konfiguraci FIPS 140-2.

Související pojmy

Integrita dat zpráv

Související úlohy

Zabezpečení

Určení specifikace CipherSpecs

XMSC_WMQ_SSL_CIPHER_SUITE

Datový typ:
Řetězec

Vlastnost:
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

Tato hodnota může být dodána jako alternativa k XMSCS_WMQ_SSL_CIPHER_SPEC.

Je-li zadána neprázdná hodnota pro proměnnou XMSCS_WMQ_SSL_CIPHER_SPEC, tato hodnota přepíše nastavení hodnoty XCOMS_WMQ_SSL_CIPHER_SUITE. Pokud XMSCS_WMQ_SSL_CIPHER_SPEC nemá hodnotu, použije se jako šifrovací sada, která má být poskytnuta sadě GSKit, hodnota XMSCS_WMQ_SSL_CIPHER_SUITE. V tomto případě je hodnota mapována na ekvivalentní hodnotu

CipherSpec , jak je popsáno v tématu Mapování názvůCipherSuite a CipherSpec pro připojení produktu XMS ke správci front produktu IBM MQ.

Pokud jsou parametry XMSCS_WMQ_SSL_CIPHER_SPEC a XMSCS_WMQ_SSL_CIPHER_SUITE prázdné, bude pole pChDef->SSLCipherSpec vyplněno mezerami.

Pouze pro produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Ve výchozím nastavení není vlastnost nastavena.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_CRYPTO_HW

Datový typ:
Řetězec

Vlastnost:
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 kryptografický hardware PKCS11 existuje speciální formát pro kryptografický hardware (kde DriverPath, TokenLabela TokenPassword jsou uživatelem zadané řetězce):

```
GSK_PKCS11=PKCS#11 DriverPath; PKCS#11 TokenLabel;PKCS#11 TokenPassword
```

Příkaz XMS neinterpretuje ani nemění obsah řetězce. Zkopíruje dodanou hodnotu až do limitu 256 jednobajtových znaků do struktury MQSCO.CryptoHardware .

Pouze pro produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Ve výchozím nastavení není vlastnost nastavena.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_FIPS_REQUIRED

Datový typ:
Logická hodnota

Vlastnost:
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 převádějí na dvě kanonické hodnoty pro MQSCO.FipsRequired:

Tabulka 882. Tabulka hodnot pro MQSCO.FlipsRequired

Hodnota	**Popis**	**Odpovídající hodnota MQSCO.FipsRequired**
ne	Lze použít libovolnou CipherSpec .	MQSSL_FIPS_NO (výchozí)
ano	Ve specifikaci CipherSpec pro toto připojení klienta lze použít pouze šifrovací algoritmy s certifikací FIPS.	MQSSL_FIPS_YES

XMS zkopíruje příslušnou hodnotu do MQSCO.FipsRequired před voláním MQCONNX.

Parametr MQSCO.FipsRequired je k dispozici pouze z produktu IBM WebSphere MQ 6.0. Pokud je tato vlastnost nastavena na hodnotu IBM WebSphere MQ 5.3, příkaz XMS se nepokusí vytvořit připojení ke správci front a namísto toho bude vyvolána příslušná výjimka.

Pouze pro produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_KEY_REPOSITORY

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Umístění souboru databáze klíčů, ve kterém jsou uloženy klíče a certifikáty.

XMS zkopíruje řetězec až do limitu 256 jednobajtových znaků do struktury MQSCO.KeyRepository . Produkt IBM MQ interpretuje tento řetězec jako název souboru včetně úplné cesty.

Pouze pro produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Ve výchozím nastavení není vlastnost nastavena.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_KEY_RESETCOUNT

Datový typ:
System.Int32

Vlastnost:
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 agentem MCA.

Produkt XMS zkopíruje hodnotu, kterou zadáte pro tuto vlastnost, do pole MQSCO.KeyResetCount před voláním MQCONNX.

Parametr MQSCO.KeyRestCount je k dispozici pouze z produktu IBM MQ verze 6. Je-li tato vlastnost nastavena na IBM MQ verze 5.3, produkt XMS se nepokusí navázat připojení ke správci front a místo toho vydá příslušnou výjimku.

Pouze pro produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Výchozí hodnota této vlastnosti je nula, což znamená, že tajné klíče nejsou nikdy znovu vyjednávány.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

XMSC_WMQ_SSL_PEER_NAME

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název typu peer, které se použije při připojení SSL ke správci front.

Pro tuto vlastnost není k dispozici žádný seznam kanonických hodnot. Namísto toho musíte sestavit tento řetězec podle pravidel pro SSLPEER.

Příklad názvu partnera:

```
"CN=John Smith, O=IBM ,OU=Test , C=GB"
```

XMS zkopíruje řetězec do správné jednobajtové kódové stránky a umístí správné hodnoty do 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 produkt .NET : Z produktu IBM MQ 8.0podporují spravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT) a nespravovaná připojení k produktu IBM MQ (WMQ_CM_CLIENT_UNMANAGED) obě podpory připojení TLS/SSL.

Ve výchozím nastavení není vlastnost nastavena.

Související pojmy

Podpora zabezpečení SSL a TLS pro nespravovaného klienta .NET

Podpora zabezpečení SSL a TLS pro spravovaného klienta .NET

Související odkazy

SSLPEERNAME

XMSC_WMQ_SYNCPOINT_ALL_GETS

Datový typ:
System.Boolean

Vlastnost:
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 ovládací prvek bodu synchronizace.

Platná hodnota	Význam
ano	Klient produktu XMS musí načíst všechny zprávy z front v rámci řízení synchronizačního bodu.

Výchozí hodnota je false.

XMSC_WMQ_TARGET_CLIENT

Datový typ:
System.Int32

Vlastnost:
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 zpracovat záhlaví.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WMQ_TARGET_DEST_JMS	Zprávy odeslané do místa určení obsahují záhlaví MQRFH2 . Tuto hodnotu zadejte v případě, že aplikace odesílá zprávy do jiné aplikace produktu XMS , aplikace IBM MQ classes for JMS nebo nativní aplikaci produktu IBM MQ , která je navržena tak, aby zpracovrovala záhlaví MQRFH2 .
XMSC_WMQ_TARGET_DEST_MQ	Zprávy odeslané do místa určení neobsahují záhlaví MQRFH2 . Tuto hodnotu zadejte v případě, že aplikace odesílá zprávy nativní aplikaci produktu IBM MQ , která není navržena pro zpracování záhlaví MQRFH2 .

Výchozí hodnota je XMSCS_WMQ_TARGET_DEST_JMS.

XMSC_WMQ_TEMP_Q_PREFIX

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Předpona použitá k vytvoření názvu dynamické fronty IBM MQ , která se vytvoří, když aplikace vytvoří XMS dočasnou frontu.

Pravidla pro vytvoření předpony jsou stejná jako pravidla pro vytvoření obsahu pole **DynamicQName** v deskriptoru objektu, ale poslední neprázdný znak musí být hvězdička (*). Pokud tato vlastnost není nastavena, je použita hodnota `CSQ.*` v systémech z/OS a `AMQ.*` na ostatních platformách. Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v dvoubodové doméně.

XMSC_WMQ_TEMP_TOPIC_PREFIX

Datový typ:
Řetězec

Vlastnost:
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 obsahující pouze platné znaky pro řetězec tématu IBM MQ je platnou hodnotou pro tuto vlastnost.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu "" (prázdný řetězec).

Poznámka: Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WMQ_TEMPORARY_MODEL

Datový typ:
Řetězec

Vlastnost:
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 dvoubodové doméně.

XMSC_WMQ_WILDCARD_FORMAT

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory, cíl

Tato vlastnost určuje, která verze syntaxe zástupných znaků má být použita.

Při použití publikování/odběru s IBM MQ '*' a '?' jsou považovány za zástupné znaky. Whereas '#' and '+' are treated as wildcards when using publish subscribe with IBM Integration Bus. Tato vlastnost nahrazuje vlastnost XMSCS_WMQ_BROKER_VERSION.

Platné hodnoty pro tuto vlastnost jsou:

XMSC_WMQ_WILDCARD_TOPIC_ONLY
Rozlišuje pouze zástupné znaky na úrovni tématu, tj. '#' a '+' jsou považovány za zástupné znaky. Tato hodnota je stejná jako XMSC_WMQ_BROKER_V2.

XMSC_WMQ_WILDCARD_CHAR_ONLY
Uvědomí pouze zástupné znaky znaků, např. "*" a '?' jsou považovány za zástupné znaky. Tato hodnota je stejná jako XMSC_WMQ_BROKER_V1.

Při výchozím nastavení je tato vlastnost nastavena na hodnotu XMSCS_WMQ_WILDCARD_TOPIC_ONLY.

XMSC_WPM_BUS_NAME

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory a Destination

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.

U místa určení, které je tématem, je tato vlastnost názvem sběrnice pro integraci služeb, ve které existuje přidružený prostor tématu. Tento prostor tématu je určen vlastností XMSCS_WPM_TOPIC_SPACE .

Není-li vlastnost nastavena pro cíl, předpokládá se, že fronta nebo přidružený prostor tématu existují ve sběrnici pro integraci služeb, ke které se aplikace připojuje.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_CONNECTION_PROTOCOL

Datový typ:
System.Int32

Vlastnost:
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_CPP_TCP	Připojení používá protokol TCP/IP.

XMSC_WPM_CONNECTION_PROXAS

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Nastavení blízkosti připojení pro připojení. Tato vlastnost určuje způsob zavření stroje systému zpráv, ke kterému se aplikace připojuje, musí být na 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 XMSCS_WPM_CONNECTION_PROXIMITY_BUS.

XMSC_WPM_DUR_SUB_HOME

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název použitý v identifikátoru URI:
durableSubscriptionHome

Název jádra systému zpráv, v nichž jsou spravovány všechny trvalé odběry pro připojení nebo cíl. Zprávy, které mají být doručovány na trvalé odběratele, jsou uloženy v bodu publikování se stejným strojem systému zpráv.

Před vytvořením trvalého odběratele, který připojení používá, musí být zadán domovský adresář odběru pro připojení. Jakákoli hodnota uvedená pro cíl přepíše hodnotu uvedenou pro připojení.

Ve výchozím nastavení není vlastnost nastavena.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

XMSC_WPM_HOST_NAME

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
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 má být použito pro připojení.

Poskytnutí těchto informací je nezbytné pouze v případě, že systém, na kterém je aplikace spuštěna, má dvě nebo více síťových rozhraní a vy potřebujete mít možnost určit, které rozhraní musí být pro připojení použito. Má-li systém pouze jedno síťové rozhraní, lze použít pouze toto rozhraní. Má-li systém dvě nebo více síťových rozhraní a nespecifikujete, které rozhraní musí být použito, je rozhraní vybráno náhodně.

nízká_port
Číslo lokálního portu, které má být použito pro připojení.

Je-li zadána také hodnota *high_port* , hodnota *low_port* je interpretována jako nejnižší číslo portu v rozsahu čísel portů.

vysoká_port
Nejvyšší číslo portu v rozsahu čísel portů. Jeden z portů v uvedeném rozsahu musí být použit pro připojení.

Zde je několik příkladů platných hodnot vlastnosti:

PLANET NAME
9.20.4.98
JUPER (1000)
9.20.4.98(1000,2000)
(1000)
(1000,2000)

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_ME_NAME

Datový typ:
Řetězec

Vlastnost:
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

Vlastnost:
ConnectionFactory

Úroveň spolehlivosti přechodných zpráv, které se odesílají pomocí připojení.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_AS_DESTINATION	Určeno výchozí úrovní spolehlivosti zadanou pro frontu nebo prostor tématu v rámci sběrnice pro integraci služeb.
XMSC_WPM_MAPPING_BEST_EFFORT_NON_ Trvalý	Nejlepší snaha, přechodné
XMSC_WPM_MAPPING_EXPRESS_NON_ Trvalý	Expresní přechodné
XMSC_WPM_MAPPING_RELIABLE_NON_ Trvalý	Spolehlivé přechodné
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Spolehlivá trvalá
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Zajištěné,

Výchozí hodnota je XMSCS_WPM_MAPPING_EXPRESS_NON_PERSISTENT.

XMSC_WPM_PERSISTENT_MAP

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Úroveň spolehlivosti trvalých zpráv, které se odesílají pomocí připojení.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_AS_DESTINATION	Určeno výchozí úrovní spolehlivosti zadanou pro frontu nebo prostor tématu v rámci sběrnice pro integraci služeb.
XMSC_WPM_MAPPING_BEST_EFFORT_NON_ Trvalý	Nejlepší snaha, přechodné
XMSC_WPM_MAPPING_EXPRESS_NON_ Trvalý	Expresní přechodné

Platná hodnota	Úroveň spolehlivosti
XMSC_WPM_MAPPING_RELIABLE_NON_ Trvalý	Spolehlivé přechodné
XMSC_WPM_MAPPING_RELIABLE_PERSISTENT	Spolehlivá trvalá
XMSC_WPM_MAPPING_ASSURED_PERSISTENT	Zajištěné,

Výchozí hodnota je XMSCS_WPM_MAPPING_RELIABLE_PERSISTENT.

XMSC_WPM_PORT

Datový typ:
System.Int32

Vlastnost:
Připojení

Číslo portu naslouchaného v jádru systému zpráv, ke kterému je aplikace připojena. Tato vlastnost je jen pro čtení.

XMSC_WPM_PROVIDER_ENDPOINTS

Datový typ:
Řetězec

Vlastnost:
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*

Významy součástí adresy koncového bodu jsou následující:

název_hostitele
Název hostitele nebo adresa IP systému, na kterém je umístěn server samozavedení. Není-li zadán žádný název hostitele nebo adresa IP, je standardní hodnota `localhost`.

číslo_portu
Číslo portu, na kterém server samozavedení naslouchá příchozím požadavkům. Není-li zadáno žádné číslo portu, je výchozí hodnota 7276.

název_řetězu
Název transportního řetězu samozavedení používaného serverem samozavedení. Platné hodnoty jsou:

Platná hodnota	Název transportního řetězu samozavedení
XMSC_WPM_BOOTSTRAP_HTTP	Systém zpráv BootstrapTunneled
XMSC_WPM_BOOTSTRAP_HTTPS	BootstrapTunneledSecureMessaging
XMSC_WPM_BOOTSTRAP_SSL	Systém zpráv BootstrapSecure
XMSC_WPM_BOOTSTRAP_TCP	Systém zpráv BootstrapBasic

Není-li uveden žádný název, výchozí hodnota je XMSCS_WPM_BOOTSTRAP_TCP.

Není-li zadána žádná adresa koncového bodu, bude použita výchozí hodnota localhost:7276:BootstrapBasicMessaging.

XMSC_WPM_SSL_CIPHER_SUITE

Datový typ:
Řetězec

Vlastnost:
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 883. Volby CipherSuite pro připojení ke stroji systému zpráv produktu 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 Windows . (To je diktováno sadou GSKit.)
2. TLS_RSA_WITH_3DES_EDE_CBC_SHA je zamítnutý. Nicméně lze ji přesto použít k přenosu až 32 GB dat před ukončením připojení s chybou AMQ9288. Chcete-li se této chybě vyhnout, je třeba při použití této CipherSpecbuď zabránit použití trojitého DES, nebo povolit resetování tajného klíče.

Pro tuto vlastnost není žádná výchozí hodnota. Chcete-li použít zabezpečení SSL nebo TLS, musíte zadat hodnotu této vlastnosti, jinak se 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" . Certifikát pro tento modul byl přesunut do historického stavu. Zákazníci by si měli prohlédnout IBM Crypto for C certificate a měli by si být vědomi jakýchkoli 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

Vlastnost:
ConnectionFactory

Cesta k souboru, který je souborem svazku klíčů obsahujícím veřejné nebo soukromé klíče, který má být použit v zabezpečeném připojení.

Nastavení vlastnosti souboru svazku klíčů na speciální hodnotu XMSCS_WPM_SSL_MS_CERTIFICATE_STORE určuje použití databáze klíčů produktu Microsoft Windows . Pomocí databáze klíčů produktu Microsoft Windows , která se nachází pod **Ovládacím panelem** > **Možnosti Internetu** > **Obsah** > **Certifikáty**, je třeba odebrat potřebu samostatné databáze souborů s klíči. Použití této konstanty v systému Windows x64 a jiných platformách není povoleno.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_LABEL

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Certifikát, který má být použit při ověřování na serveru. Není-li zadána žádná hodnota, bude použit výchozí certifikát.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_PW

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Heslo pro soubor svazku klíčů.

Tuto vlastnost lze použít jako alternativu k použití souboru XMSCS_WPM_SSL_KEYRING_STASH_FILE ke konfiguraci hesla pro soubor svazku klíčů.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_SSL_KEYRING_STASH_FILE

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název binárního souboru obsahujícího heslo souboru úložiště klíčů.

Tuto vlastnost lze použít jako alternativu k použití souboru XMSCS_WPM_SSL_KEYRING_PW ke konfiguraci hesla pro soubor svazku klíčů.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_TARGET_GROUP

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název cílové skupiny jader systému zpráv. Povaha cílové skupiny je určena vlastností XMSCS_WPM_TARGET_TYPE .

Tuto vlastnost nastavte v případě, že chcete omezit hledání stroje systému zpráv na podskupinu strojů systému zpráv v rámci sběrnice pro integraci služeb. Chcete-li, aby vaše aplikace byla schopna připojit se k libovolnému stroji systému zpráv v rámci sběrnice pro integraci služeb, nenastavujte tuto vlastnost.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_TARGET_VÝZNAMNOST

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Významnost cílové skupiny jader systému zpráv.

Platné hodnoty vlastnosti jsou následující:

Platná hodnota	Význam
XMSC_WPM_TARGET_SIGNIFICANCE_ Preferovaný	Je-li k dispozici stroj systému zpráv v cílové skupině, je vybrán. Jinak je vybrán stroj systému zpráv mimo cílovou skupinu za předpokladu, že se nachází ve stejné sběrnici pro integraci služeb.
XMSC_WPM_TARGET_SIGNIFICANCE_ POVINNÉ	Vybraný stroj systému zpráv musí být v cílové skupině. Není-li stroj systému zpráv v cílové skupině k dispozici, dojde k selhání procesu připojení.

Výchozí hodnota vlastnosti je XMSCS_WPM_TARGET_SIGNIFICANCE_PREFERRED.

XMSC_WPM_TARGET_TRANSPORT_CHAIN

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Název transportu příchozích požadavků, který musí aplikace používat pro připojení k jádru systému zpráv.

Hodnotou této 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	Systém zpráv InboundBasic

Výchozí hodnota vlastnosti je XMSCS_WPM_TARGET_TRANSPORT_CHAIN_BASIC.

XMSC_WPM_TARGET_TYPE

Datový typ:
System.Int32

Vlastnost:
ConnectionFactory

Typ cílové skupiny jader systému zpráv. Tato vlastnost určuje charakter cílové skupiny identifikované vlastností XMSCS_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 v rámci člena sběrnice.
XMSC_WPM_TARGET_TYPE_CUSTOM	Název cílové skupiny je název skupiny strojů systému zpráv definovaných uživatelem. Cílová skupina je všechny stroje systému zpráv, které jsou registrovány ve skupině definované uživatelem.
XMSC_WPM_TARGET_TYPE_ME	Název cílové skupiny je název stroje systému zpráv. Cílová skupina je určený stroj systému zpráv.

Ve výchozím nastavení není vlastnost nastavena.

XMSC_WPM_TEMP_Q_PREFIX

Datový typ:
Řetězec

Vlastnost:
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" následovaným předponou. Zbytek názvu se skládá ze systémem generovaných znaků.

Ve výchozím nastavení není vlastnost nastavena, což znamená, že název dočasné fronty nemá předponu.

Tato vlastnost je relevantní pouze v dvoubodové doméně.

XMSC_WPM_TEMP_TOPIC_PREFIX

Datový typ:
Řetězec

Vlastnost:
ConnectionFactory

Předpona používaná k vytvoření názvu dočasného tématu vytvořeného aplikací. Předpona může obsahovat až 12 znaků.

Název dočasného tématu začíná znaky "_T", za nimiž následuje předpona. Zbytek názvu se skládá ze systémem generovaných znaků.

Ve výchozím nastavení 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

Vlastnost:
Místo určení

Název použitý v identifikátoru URI:
topicSpace

Název prostoru témat, který obsahuje dané téma. Tato vlastnost může mít pouze místo určení, které je tématem.

Ve výchozím nastavení není vlastnost nastavena, což znamená, že se předpokládá výchozí prostor tématu.

Tato vlastnost je relevantní pouze v doméně publikování/odběru.

Odkaz na vývoj aplikací produktu Managed File Transfer

Referenční informace, které vám pomohou při vývoji aplikací pro produkt Managed File Transfer.

Příklady použití příkazu fteCreateTransfer ke spuštění programů

Příkaz **fteCreateTransfer** můžete použít k uvedení programů, které se mají spustit před nebo po přenosu.

Kromě použití produktu **fteCreateTransfer**existují i jiné způsoby vyvolání programu před přenosem nebo po něm. Další informace najdete v tématu Zadání programů pro spouštění s produktem MFT.

Všechny tyto příklady používají k určení programu následující syntaxi:

```
[type:]commandspec[,[retrycount][,[retrywait][,successrc]]]
```

Další informace o této syntaxi naleznete v tématu **fteCreateTransfer**: Spuštění nového přenosu souborů.

Spuštění spustitelného programu

Následující příklad určuje spustitelný program s názvem `mycommand` a předává dva argumenty, a a b, do programu.

```
mycommand(a,b)
```

Chcete-li spustit tento program na zdrojovém agentovi AGENT1 před spuštěním přenosu, použijte následující příkaz:

```
fteCreateTransfer -sa AGENT1 -da AGENT2 -presrc mycommand(a,b)
destinationSpecification sourceSpecification
```

Spuštění a zopakování spustitelného programu

Následující příklad určuje spustitelný program s názvem `simple`, který nepřijímá žádné argumenty. Hodnota 1 je uvedena pro `retrycount` a hodnota 5 je uvedena pro `retrywait`. Tyto hodnoty znamenají, že program bude zopakován jednou, pokud nevrátí úspěšný návratový kód po uplynutí pěti sekund. Pro `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 uvedení úspěšných návratových kódů

Následující příklad uvádí skript Ant s názvem `myscript` a předává do skriptu dvě vlastnosti. Skript se spouští pomocí příkazu **`fteAnt`** . Hodnota pro `successrc` je uvedena jako >2&<7&!5|0|14, což znamená, 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 k volání

Následující příklad uvádí skript Ant nazvaný `script2` a dva cíle, `target1` a `target2`, které se mají volat. Vlastnost `prop1` se také předává spolu s hodnotou `recmfm(F,B)`. Čárka (,) a závorky v této hodnotě jsou uvozeny pomocí znaku 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

Můžete zadat úlohu Ant jako jakákoli z následujících volání pro přenos:

- Na zdroji před zpracováním
- Na zdroji po zpracování
- předurčení
- místo určení

Když je spuštěna úloha Ant , uživatelská metadata přenosu jsou k dispozici pomocí proměnných prostředí. Přístup k těmto datům lze získat 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. Hodnota 3 je uvedena pro `retrycount`, hodnota 30 je uvedena pro `retrywait` a hodnota 0 je uvedena pro `successrc`. Tyto hodnoty znamenají, že se skript opakuje třikrát, pokud nevrátí úspěšný návratový kód 0 s čekací dobou 30 sekund mezi jednotlivými pokusy.

```
jcl:ZOSBATCH,3,30,0
```

kde ZOSBATCH je člen 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

Zadání programů pro spouštění s produktem MFT

Související odkazy

fteCreateTransfer: spuštění nového přenosu souboru

fteAnt: spuštění úloh Ant v produktu MFT

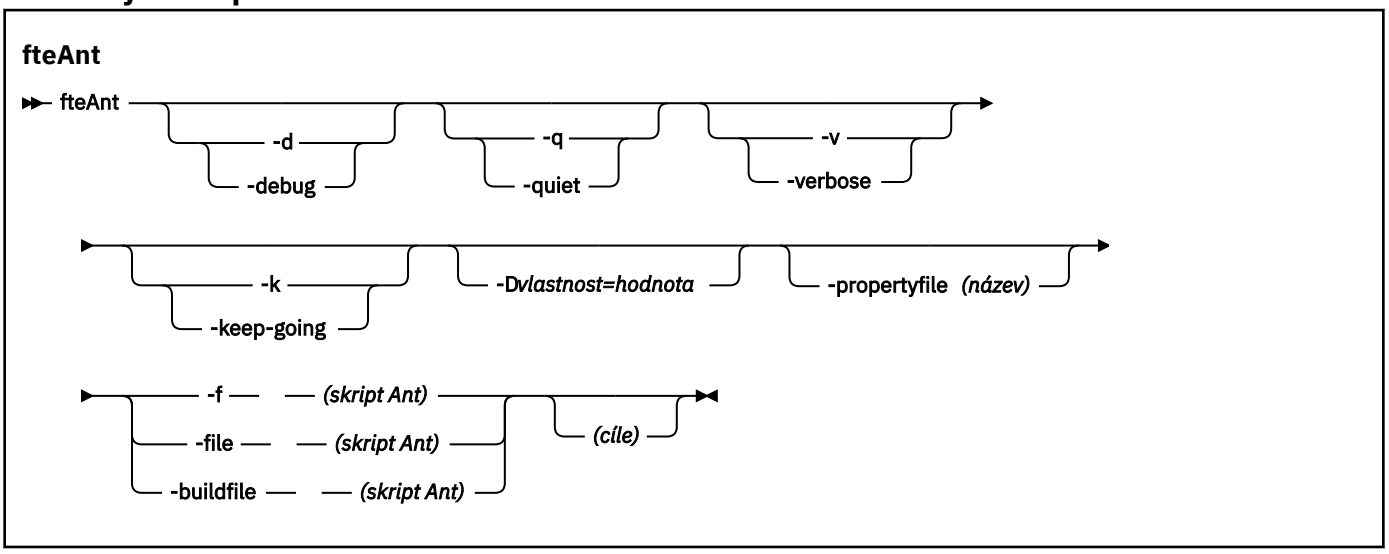
Příkaz **fteAnt** spouští skripty Ant v prostředí, které má k dispozici úlohy produktu Managed File Transfer Ant . Na rozdíl od standardního příkazu **ant** vyžaduje obslužný program **fteAnt** , abyste definovali skriptový soubor.

Úlohy MFT Ant a vnořené parametry

Produkt Managed File Transfer poskytuje počet úloh produktu Ant , které lze použít k přístupu k funkcím přenosu souborů. K dispozici je také sada vnořených parametrů; tyto parametry popisují vnořené sady prvků, které jsou společné pro několik zadaných úloh Ant.

Syntaxi příkazu **fteAnt** , parametry, příklad použití a návratové kódy jsou popsány ve zbytku tohoto tématu. Podrobnosti o úlohách Ant a vnořených parametrech, které poskytuje MFT, viz dílčí témata.

Syntaxe příkazu fteAnt



Parametry

-debug nebo -d
Volitelné. Generovat výstup ladění.

-quiet nebo -q
Volitelné. Vygenerovat minimální výstup.

-verbose nebo -v
Volitelné. Vygenerovat podrobný výstup.

-keep-going nebo -k
Volitelné. Provést všechny cíle, které nezávisí na cílech, na kterých došlo k selhání.

-D *vlastnost=hodnota*
Volitelné. Pro danou vlastnost *property*použijte hodnotu *value* . Vlastnosti nastavené s **-D** mají přednost před vlastnostmi nastavovmi v souboru vlastností.

Použijte vlastnost **com.ibm.wmqfte.propertyset** k uvedení sady voleb konfigurace, které se použijí pro úlohy Ant . Jako hodnotu této vlastnosti použijte název jiného než výchozího koordinačního správce front. Úlohy Ant pak využívají sadu voleb konfigurace, které jsou přidruženy k tomuto nevýchozímu koordinačnímu správci front. Pokud tuto vlastnost nezadáte, použije se výchozí sada voleb konfigurace, které jsou založeny na výchozím koordinačním správci front. Pokud zadáte atribut **cmdqm** pro úlohu Ant , tento atribut má přednost před sadou voleb konfigurace, které jsou určeny pro příkaz **fteAnt** . Toto chování se použije bez ohledu na to, zda používáte výchozí sadu voleb konfigurace nebo určujete sadu pomocí vlastnosti **com.ibm.wmqfte.propertyset** .

-propertyfile (*název*)
Volitelné. Načte všechny vlastnosti ze souboru s vlastnostmi produktu **-D** , které mají přednost.

-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 nezadáte hodnotu pro tento parametr, spustí se výchozí cíl skriptu.

-version
Volitelné. Zobrazí příkaz Managed File Transfer a Ant .

-? nebo -h
Volitelné. Zobrazuje syntaxi příkazu.

Příklad

V tomto příkladě se spustí cíl **copy** ve skriptu Ant `fte_script.xml` a příkaz vypíše ladící výstup na standardní výstup.

```
fteAnt -d -f fte_script.xml copy
```

Návratové kódy

0
Příkaz byl úspěšně dokončen.

1
Příkaz skončil neúspěšně.

Další návratové kódy stavu lze také zadat ze skriptů Ant, například pomocí úlohy selhání Ant .

Další informace viz Selhání .

fte: úloha awaitoutcome Ant

Čeká na dokončení operace **fte:filecopy**, **fte:filemove**nebo **fte:call** .

Atributy

id
Povinné Identifikuje přenos, od kterého se bude očekávat výsledek. Obvykle se jedná o vlastnost nastavenou atributem idProperty v rámci úloh fte: filecopy, fte: filemovenebo fte: call .

vlastnost rcproperty
Povinné Určuje vlastnost pro uložení návratového kódu úlohy **fte:awaitoutcome** .

časový limit
Volitelné. Maximální doba v sekundách, po kterou se má čekat na dokončení operace. Minimální časový limit je jedna sekunda. Nezadáte-li hodnotu časového limitu, úloha **fte:awaitoutcome** čeká na výsledek operace, která má být určena, navždy.

Příklad

V tomto příkladu je spuštěna kopie souboru a její identifikátor je uložen ve vlastnosti `copy.id` . Zatímco kopírování probíhá, jiné zpracování může probíhat. Příkaz **fte:awaitoutcome** se používá k čekání na dokončení operace kopírování. Příkaz **fte:awaitoutcome** identifikuje, která operace má čekat na použití identifikátoru uloženého ve vlastnosti `copy.id` . V produktu **fte:awaitoutcome** je uložen 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: volání úlohy Ant

Úlohu **fte:call** lze použít ke vzdálenému volání skriptů a programů.

Tato úloha vám umožňuje odeslat na agenta požadavek **fte:call** . Agent zpracuje tento požadavek spuštěním skriptu nebo programu a vrátí výsledek. Příkazy, které se mají volat, musí být přístupné agentovi. Ujistěte se, že hodnota vlastnosti commandPath v souboru `agent.properties` obsahuje umístění příkazů, které se mají volat. Všechny informace o cestě zadané vnořeným prvkem příkazu musí být relativní k umístění, které určuje vlastnost commandPath . Ve výchozím nastavení commandPath je prázdný, takže agent nemůže volat žádné příkazy. Další informace o této vlastnosti najdete v tématu commandPath MFT -vlastnost.

Další informace o souboru `agent.properties` naleznete v tématu Soubor MFT `agent.properties`.

Atributy

agent
Povinné Uvádí agenta, na který se má odeslat požadavek **fte:call** . Uveďte 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, do kterého má být odeslán požadavek. Zadejte tyto informace ve tvaru *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 naslouchá správce front
- *channel* je volitelný kanál SVRCONN, který má být použit

Vynecháte-li informace o *host*, *port*nebo *channel* pro správce front příkazů, použije se informace o připojení zadané v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- použití proměnné *host* , režim vazeb je použit
- proměnná *port* , použije se hodnota 1414.

- Proměnná *kanál* , SYSTEM.DEF.SVRCONN se používá.

Další informace viz Soubor MFT command.properties .

Atributy však nemůžete přeskočit ve středu, například `qmgrname@host@@channel`. Můžete mít, například `qmgrname@host`, nebo `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut s použitím oddělovače `@` . V závislosti na počtu nalezených tokenů je použit první token jako *qmgrname*, druhý jako *host*, třetí jako *port* a nakonec *channel*.

Další informace viz Soubor MFT command.properties.

Vlastnost **`com.ibm.wmqfte.propertySet`** můžete použít k uvedení, který soubor `command.properties` použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , je-li tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, je proveden pokus o připojení k výchozímu správci front, který je definován v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako u atributu cmdqm , tj. `qmgrname@host@port@channel`.

vlastnost idproperty
Volitelné, pokud jste neuvedli `outcome` z `defer`. Uvádí název vlastnosti, ke které se má přiřadit identifikátor přenosu. Identifikátory přenosu se generují 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 zadat, pokud jste také zadali vlastnost `outcome` produktu `ignore`. Pokud jste však také zadali vlastnost `outcome` produktu `defer`, musíte zadat volbu `idproperty` .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku **`fte:call`** . Názvy úloh můžete použít k vytvoření logických skupin přenosů. Použijte úlohu “fte: uuid Ant úloha” na stránce 2098 ke generování pseudojedinečných názvů úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti `com.ibm.wmqfte.ant.jobName` , je-li tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, nebude k požadavku produktu **`fte:call`** 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 používá ID uživatele, které se používá ke spuštění skriptu Ant.

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace **`fte:call`** před vrácením řízení skriptu Ant . Uveďte jednu z následujících možností:

čekejte
Úloha čeká na dokončení operace **`fte:call`** , než se vrátí. Je-li zadán `outcome` z `await` , je atribut `idproperty` nepovinný.

defer
Úloha se vrátí, jakmile byl odeslán požadavek **`fte:call`** a předpokládá se, že výsledek operace volání je řešen později buď pomocí úloh awaitoutcome nebo ignoreoutcome . Je-li zadán `outcome` z `defer` , je požadován atribut `idproperty` .

Ignorovat
Není-li výsledek operace **`fte:call`** důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí, jakmile je zadán požadavek **`fte:call`** , aniž by došlo k přidělení prostředků pro sledování výsledku příkazu. Je-li zadán `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut outcome , úloha se standardně použije k použití hodnoty `await`.

rcproperty
Volitelné. Určuje název vlastnosti, do které má být přiřazen výsledný kód požadavku **`fte:call`** . Výsledkový kód odráží celkový výsledek požadavku **`fte:call`** .

Tuto vlastnost nelze zadat, pokud jste také zadali vlastnost `outcome` s hodnotou `ignore` nebo `defer`. Pokud jste však určili výstup příkazu `await`, je třeba zadat hodnotu `rcproperty` .

Parametry zadané jako vnořené prvky

fte: příkaz
Uvádí příkaz, který má agent zavolat. K dané operaci**`fte:call`** lze přidružit pouze jeden prvek `fte:command` . Příkaz, který má být volán, musí být umístěn na cestě zadané vlastností commandPath v souboru `agent.properties` agenta.

fte: metadata
Můžete zadat metadata, která se mají přidružit k operaci volání. Tato metadata jsou zaznamenána ve zprávách protokolu generovaných operací volání. K danému prvku přenosu můžete přidružit pouze jeden blok metadat; tento blok však může obsahovat mnoho částí metadat.

Příklad

Tento příklad ukazuje, jak volat příkaz na serveru AGENT1 , který běží na 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 zadané 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: zrušení úlohy Ant

Ruší spravovaný přenos Managed File Transfer nebo spravované volání. Spravovaný přenos mohl být vytvořen pomocí úloh produktu **`fte:filecopy`** nebo **`fte:filemove`** . Je možné, že spravované volání bylo vytvořeno pomocí úlohy **`fte:call`** .

Atributy

agent
Povinné Uvádí agenta, na který se má odeslat požadavek **`fte:cancel`** . Hodnota je ve tvaru: *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, do kterého má být odeslán požadavek. Zadejte tyto informace ve tvaru *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 naslouchá správce front
- *channel* je volitelný kanál SVRCONN, který má být použit

Vynecháte-li informace o *host*, *port*nebo *channel* pro správce front příkazů, použije se informace o připojení zadané v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- použití proměnné *host* , režim vazeb je použit
- proměnná *port* , použije se hodnota 1414.
- Proměnná *kanál* , SYSTEM.DEF.SVRCONN se používá.

Další informace viz Soubor MFT command.properties .

Atributy však nemůžete přeskočit ve středu, například `qmgrname@host@@channel`. Můžete mít, například `qmgrname@host`, nebo `qmgrname@host@port`nebo `qmgrname@hostport@@channel`.

MFT rozdělí daný atribut s použitím oddělovače `@` . V závislosti na počtu nalezených tokenů je použit první token jako *qmgrname*, druhý jako *host*, třetí jako *port* a nakonec *channel*.

Další informace viz Soubor MFT command.properties.

Vlastnost **`com.ibm.wmqfte.propertySet`** můžete použít k uvedení, který soubor `command.properties` použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , je-li tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, je proveden pokus o připojení k výchozímu správci front, který je definován v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako u atributu cmdqm , tj. `qmgrname@host@port@channel`.

id

Povinné Uvádí identifikátor přenosu, který se má převést na zrušení. Identifikátory přenosu se generují v místě odeslání požadavku na přenos pomocí úloh fte: filecopy a fte: fileexove .

původní_uživatel

Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku **`cancel`** . Není-li atribut originuser použit, úloha standardně používá ID uživatele, které se používá ke spuštění skriptu Ant .

Příklad

Příklad odešle požadavek **`fte:cancel`** správci front příkazů `qm0`. Požadavek **`fte:cancel`** je cílený na `agent1` ve správci front `qm1` pro 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 nebyl odstraněn ze zdrojového agenta.

Atributy

cmdqm

Volitelné. Správce front příkazů, do kterého má být odeslán požadavek. Zadejte tyto informace ve tvaru *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 naslouchá správce front
- *channel* je volitelný kanál SVRCONN, který má být použit

Vynecháte-li informace o *host*, *port*nebo *channel* pro správce front příkazů, použije se informace o připojení zadané v souboru command.properties .

Upozornění: Není-li uvedena žádná hodnota pro:

- použití proměnné *host* , režim vazeb je použit
- proměnná *port* , použije se hodnota 1414.
- Proměnná *kanál* , SYSTEM.DEF.SVRCONN se používá.

Další informace viz Soubor MFT command.properties .

Atributy však nemůžete přeskočit ve středu, například qmgrname@host@@channel. Můžete mít, například qmgrname@host, nebo qmgrname@host@portnebo qmgrname@hostport@@channel.

MFT rozdělí daný atribut s použitím oddělovače @ . V závislosti na počtu nalezených tokenů je použit první token jako *qmgrname*, druhý jako *host*, třetí jako *port* a nakonec *channel*.

Další informace viz Soubor MFT command.properties.

Vlastnost **com.ibm.wmqfte.propertySet** můžete použít k uvedení, který soubor command.properties použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost com.ibm.wmqfte.ant.commandQueueManager , je-li tato vlastnost nastavena. Není-li vlastnost com.ibm.wmqfte.ant.commandQueueManager nastavena, je proveden pokus o připojení k výchozímu správci front, který je definován v souboru command.properties . Formát vlastnosti com.ibm.wmqfte.ant.commandQueueManager je stejný jako u atributu cmdqm , tj. qmgrname@host@port@channel.

dst
Povinné Uvádí cílového agenta pro operaci kopírování. Zadejte tyto informace ve tvaru: agentname@qmgrname , kde agentname je název cílového agenta a qmgrname je název správce front, ke kterému je tento agent přímo připojen.

vlastnost idproperty
Volitelné, pokud jste neuvedli outcome z defer. Uvádí název vlastnosti, ke které se má přiřadit identifikátor přenosu. Identifikátory přenosu se generují 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 zadat, pokud jste také zadali vlastnost outcome produktu ignore. Pokud jste však také zadali vlastnost outcome produktu defer, musíte zadat volbu idproperty .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku na kopírování. Názvy úloh můžete použít k vytvoření logických skupin přenosů. Použijte úlohu “fte: uuid Ant úloha” na stránce 2098 ke generování pseudojedinečných názvů úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti com.ibm.wmqfte.ant.jobName , je-li tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, nebude k požadavku na kopírování přidružen žádný název úlohy.

Původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku na kopírování. Pokud nepoužijete atribut originuser , úloha standardně používá ID uživatele, které se používá ke spuštění skriptu Ant .

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace kopírování, než vrátí řízení do skriptu Ant . Uveďte jednu z následujících možností:

čekejte
Úloha čeká na dokončení operace kopírování, než se vrátí. Je-li zadán `outcome` z `await` , je atribut `idproperty` nepovinný.

defer
Úloha se vrátí, jakmile je odeslán požadavek na kopírování a předpokládá se, že výsledek operace kopírování se bude řešit později pomocí úloh “fte: úloha awaitoutcome Ant” na stránce 2085 nebo “fte: ignoreoutcome Ant úloha” na stránce 2096 . Je-li zadán `outcome` z `defer` , je požadován atribut `idproperty` .

Ignorovat
Není-li výsledek operace kopírování důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí, jakmile je odeslán požadavek na kopírování, aniž by došlo k přidělení prostředků pro sledování výsledku přenosu. Je-li zadán `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut outcome , úloha se standardně použije k použití hodnoty `await`.

priority (priorita)
Volitelné. Uvádí prioritu, která se má přidružit k požadavku na kopírování. Obecně platí, že požadavky na přenos vyšší priority mají přednost před požadavky nižší priority. Hodnota priority musí být v rozsahu 0 až 9 (včetně). Hodnota priority 0 odpovídá nejnižší prioritě a hodnota 9 je nejvyšší prioritou. Pokud atribut `priority` nezadáte, bude výchozí hodnota přenosu nastavena na hodnotu 0.

rcproperty
Volitelné. Uvádí název vlastnosti, která má přiřadit výsledkový kód požadavku na kopírování. Výsledkový kód odráží celkový výsledek požadavku na kopírování.

Tuto vlastnost nelze zadat, pokud jste také zadali vlastnost `outcome` s hodnotou `ignore` nebo `defer`. Pokud však zadáte výstup příkazu `await`, je třeba zadat hodnotu `rcproperty` .

Časový limit transferRecovery
Volitelné. Nastavuje dobu (v sekundách), během které se zdrojový agent neustále pokouší o zotavení zastaveného přenosu souborů. Uveďte jednu z následujících možností:

-1
Agent bude pokračovat v pokusu o obnovení pozastaveného přenosu, dokud nebude přenos dokončen. Použití této volby je ekvivalent výchozího chování agenta, když není tato vlastnost nastavena.

0
Agent zastaví přenos souboru, jakmile vstoupí do zotavení.

>0
Agent se bude i nadále pokoušet o zotavení pozastaveného přenosu po dobu v sekundách, jak je nastaveno kladné celé zadané hodnoty. 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 se agent snaží o zotavení přenosu po dobu 6 hodin od okamžiku, kdy vstoupí do zotavení. Maximální hodnota pro tento atribut je 999999999.

Při zadání hodnoty časového limitu pro zotavení při přenosu je tento parametr nastaven na jednotlivé přenosové cesty. Chcete-li nastavit globální hodnotu pro všechny přenosy v síti produktu Managed File Transfer , můžete přidat vlastnost do Vlastnosti časového limitu obnovy přenosu. Další informace naleznete v tématu Volba časového limitu pro přenosy v obnově.

src
Povinné Uvádí zdrojového agenta pro operaci kopírování. Uveďte 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 zadané jako vnořené prvky

fte: filespec
Povinné Musíte uvést alespoň jednu specifikaci souboru, která identifikuje soubory, které se mají zkopírovat. Je-li to nutné, můžete uvést více než jednu specifikaci souboru. Další informace viz “fte: filespec Ant vnořený prvek” na stránce 2099.

fte: metadata
Můžete zadat metadata, která se mají přidružit k operaci kopírování. Tato metadata se přenášejí s přenosem a jsou zaznamenána ve zprávách protokolu generovaných přenosem. K danému prvku přenosu můžete přidružit pouze jeden blok metadat; tento blok však může obsahovat mnoho částí metadat. Další informace naleznete v tématu fte: metadata .

fte: presrc
Určuje vyvolání programu, které se má provést na zdrojovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:presrc` . Další informace naleznete v tématu vyvolání programu .

fte: predst
Určuje vyvolání programu, které se má provést na cílovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:predst` . Další informace naleznete v tématu vyvolání programu .

fte: postsrc
Určuje vyvolání programu, které se má provést na zdrojovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:postsrc` . Další informace naleznete v tématu vyvolání programu .

fte: postdst
Uvádí vyvolání programu, které se má provést na cílovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:postdst` . Další informace naleznete v tématu vyvolání programu .

Je-li příkaz fte:presrc, fte:predst, fte:postsrc, fte:postdst a uživatelské procedury nevrátí úspěšný stav, jsou pravidla v uvedeném pořadí následující:

1. Spusťte uživatelské procedury pro spuštění zdroje. Pokud dojde k selhání spuštění zdroje, přenos selže a nebude dále spuštěno žádné další spuštění.
2. Spusťte volání před zdrojovým kódem (je-li k dispozici). Pokud volání před zdrojovým kódem selže, přenos selže a nic dalšího se nespustí.
3. Spusťte uživatelské procedury cíle spuštění. Pokud se ukončení cíle nezdaří, přenos selže a nebude dále spuštěno žádné další spuštění.
4. Spusťte volání před místem určení (je-li k dispozici). Pokud volání před místem určení selže, přenos selže a nic dalšího se nespustí.
5. Proveďte přenosy souborů.
6. Spusťte uživatelské procedury ukončení cíle. Pro tyto výstupy není k dispozici žádný stav selhání.
7. Je-li přenos úspěšný (pokud se úspěšně přenos některých souborů úspěšně přenesl, je považován za úspěšný), spusťte volání po cíli (je-li k dispozici). Pokud se volání po cíli nezdaří, přenos selže.
8. Spusťte uživatelské procedury ukončení. Pro tyto výstupy není k dispozici žádný stav selhání.
9. Je-li přenos úspěšný, spusťte post-source volání (pokud existuje). Pokud se volání po zdrojovém kódu nezdaří, přenos selže.

Příklady

Tento příklad ukazuje základní přenos souborů mezi `agent1` a `agent2`. Příkaz pro zahájení přenosu souborů je odeslán správci front s názvem `qm0`, pomocí připojení v režimu transportu klienta. Výsledek operace přenosu souboru 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 ukazuje stejný přenos souboru, ale s přidáním metadat a programem se začne provádět po dokončení přenosu na zdrojovém agentovi.

```
<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 v zotavení

Související úlohy

Použití Apache Ant s MFT

fte: fileove Ant úloha

Úloha **fte:filemove** přesouvá soubory mezi agenty Managed File Transfer . Pokud byl soubor úspěšně přenesen ze zdrojového agenta na cílového agenta, soubor se odstraní ze zdrojového agenta.

Atributy

cmdqm

Volitelné. Správce front příkazů, do kterého má být odeslán požadavek. Zadejte tyto informace ve tvaru *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 naslouchá správce front
- *channel* je volitelný kanál SVRCONN, který má být použit

Vynecháte-li informace o *host*, *port*nebo *channel* pro správce front příkazů, použije se informace o připojení zadané v souboru command.properties .

Upozornění: Není-li uvedena žádná hodnota pro:

- použití proměnné *host* , režim vazeb je použit
- proměnná *port* , použije se hodnota 1414.
- Proměnná *kanál* , SYSTEM.DEF.SVRCONN se používá.

Další informace viz Soubor MFT command.properties .

Atributy však nemůžete přeskočit ve středu, například qmgrname@host@@channel. Můžete mít, například qmgrname@host, nebo qmgrname@host@portnebo qmgrname@hostport@@channel.

MFT rozdělí daný atribut s použitím oddělovače @ . V závislosti na počtu nalezených tokenů je použit první token jako *qmgrname*, druhý jako *host*, třetí jako *port* a nakonec *channel*.

Další informace viz Soubor MFT command.properties.

Vlastnost **com.ibm.wmqfte.propertySet** můžete použít k uvedení, který soubor `command.properties` použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , je-li tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, je proveden pokus o připojení k výchozímu správci front, který je definován v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako u atributu cmdqm , tj. `qmgrname@host@port@channel`.

dst
Povinné Uvádí cílového agenta pro operaci kopírování. Zadejte tyto informace ve tvaru: *agentname*`@`*qmgrname* , kde *agentname* je název cílového agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

vlastnost idproperty
Volitelné, pokud jste neuvedli `outcome` z `defer`. Uvádí název vlastnosti, ke které se má přiřadit identifikátor přenosu. Identifikátory přenosu se generují 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 zadat, pokud jste také zadali vlastnost `outcome` produktu `ignore`. Pokud jste však také zadali vlastnost `outcome` produktu `defer`, musíte zadat volbu `idproperty` .

JOBNAME
Volitelné. Přiřadí název úlohy k požadavku na přesun. Názvy úloh můžete použít k vytvoření logických skupin přenosů. Použijte úlohu fte: uuid ke generování pseudojedinečných názvů úloh. Pokud nepoužijete atribut jobname , úloha standardně použije hodnotu vlastnosti `com.ibm.wmqfte.ant.jobName` , je-li tato vlastnost nastavena. Pokud tuto vlastnost nenastavíte, nebude k požadavku na přesun přidružen žádný název úlohy.

Původní uživatel
Volitelné. Uvádí identifikátor původního uživatele, který se má přidružit k požadavku na přesun. Pokud nepoužijete atribut originuser , úloha standardně používá ID uživatele, které se používá ke spuštění skriptu Ant .

výsledek
Volitelné. Určuje, zda úloha čeká na dokončení operace přesunu, než vrátí řízení do skriptu Ant . Uveďte jednu z následujících možností:

čekejte
Úloha čeká na dokončení operace přesunu, než se vrátí. Je-li zadán `outcome` z `await` , je atribut `idproperty` nepovinný.

defer
Úloha se vrátí, jakmile byl odeslán požadavek na přesun a předpokládá se, že výsledek operace přesunu je řešen později buď pomocí úlohy “fte: úloha awaitoutcome Ant” na stránce 2085 , nebo “fte: ignoreoutcome Ant úloha” na stránce 2096 . Je-li zadán `outcome` z `defer` , je požadován atribut `idproperty` .

Ignorovat
Není-li výsledek operace přesunutí důležitý, můžete zadat hodnotu `ignore`. Úloha se poté vrátí, jakmile byl odeslán požadavek na přesun, aniž by došlo k alokaci žádných prostředků pro sledování výsledku přenosu. Je-li zadán `outcome` z `ignore` , nelze zadat atribut `idproperty` .

Pokud neuvedete atribut outcome , úloha se standardně použije k použití hodnoty `await`.

priority (priorita)
Volitelné. Uvádí prioritu, která se má přidružit k požadavku na přesun. Obecně platí, že požadavky na přenos vyšší priority mají přednost před požadavky nižší priority. Hodnota priority musí být v rozsahu 0 až 9 (včetně). Hodnota priority 0 odpovídá nejnižší prioritě a hodnota 9 je nejvyšší prioritou. Pokud atribut `priority` nezadáte, bude výchozí hodnota přenosu nastavena na hodnotu 0.

rcproperty
Volitelné. Uvádí název vlastnosti, která má přiřadit výsledný kód požadavku na přesun. Výsledkový kód odráží celkový výsledek požadavku na přesun.

Tuto vlastnost nelze zadat, pokud jste také zadali vlastnost `outcome` s hodnotou `ignore` nebo `defer`. Pokud jste však určili výstup příkazu `await`, je třeba zadat hodnotu `rcproperty` .

Časový limit transferRecovery
Volitelné. Nastavuje dobu (v sekundách), během které se zdrojový agent neustále pokouší o zotavení zastaveného přenosu souborů. Uveďte jednu z následujících možností:

-1
Agent bude pokračovat v pokusu o obnovení pozastaveného přenosu, dokud nebude přenos dokončen. Použití této volby je ekvivalent výchozího chování agenta, když není tato vlastnost nastavena.

0
Agent zastaví přenos souboru, jakmile vstoupí do zotavení.

>0
Agent se bude i nadále pokoušet o zotavení pozastaveného přenosu po dobu v sekundách, jak je nastaveno kladné celé zadané hodnoty. 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 se agent snaží o zotavení přenosu po dobu 6 hodin od okamžiku, kdy vstoupí do zotavení. Maximální hodnota pro tento atribut je 999999999.

Při zadání hodnoty časového limitu pro zotavení při přenosu je tento parametr nastaven na jednotlivé přenosové cesty. Chcete-li nastavit globální hodnotu pro všechny přenosy v síti produktu Managed File Transfer , můžete přidat vlastnost do Vlastnosti časového limitu obnovy přenosu. Další informace naleznete v tématu Volba časového limitu pro přenosy v obnově.

src
Povinné Uvádí zdrojového agenta pro operaci přesunutí. Zadejte tyto informace ve tvaru: *agentname*@*qmgrname* , kde *agentname* je název zdrojového agenta a *qmgrname* je název správce front, ke kterému je tento agent přímo připojen.

Parametry zadané jako vnořené prvky

fte: filespec
Povinné Musíte uvést alespoň jednu specifikaci souboru, která identifikuje soubory, které se mají přesunout. Je-li to nutné, můžete uvést více než jednu specifikaci souboru. Další informace viz “fte: filespec Ant vnořený prvek” na stránce 2099.

fte: metadata
Volitelné. Můžete zadat metadata, která se mají přidružit k operaci přesunutí souboru. Tato metadata se přenášejí s přenosem a jsou zaznamenána ve zprávách protokolu generovaných přenosem.
K danému prvku přenosu můžete přidružit pouze jeden blok metadat; tento blok však může obsahovat mnoho částí metadat. Další informace naleznete v tématu fte: metadata .

fte: presrc
Volitelné. Určuje vyvolání programu, které se má provést na zdrojovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:presrc` . Další informace naleznete v tématu vyvolání programu .

fte: predst
Volitelné. Určuje vyvolání programu, které se má provést na cílovém agentovi před spuštěním přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:predst` . Další informace naleznete v tématu vyvolání programu .

fte: postsrc
Volitelné. Určuje vyvolání programu, které se má provést na zdrojovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:postsrc` . Další informace naleznete v tématu vyvolání programu .

fte: postdst
Volitelné. Uvádí vyvolání programu, které se má provést na cílovém agentovi po dokončení přenosu. K danému přenosu můžete přidružit pouze jediný prvek `fte:postdst` . Další informace naleznete v tématu vyvolání programu .

Je-li příkaz fte:presrc, fte:predst, fte:postsrc, fte:postdst a uživatelské procedury nevrátí úspěšný stav, jsou pravidla v uvedeném pořadí následující:

1. Spusťte uživatelské procedury pro spuštění zdroje. Pokud dojde k selhání spuštění zdroje, přenos selže a nebude dále spuštěno žádné další spuštění.
2. Spusťte volání před zdrojovým kódem (je-li k dispozici). Pokud volání před zdrojovým kódem selže, přenos selže a nic dalšího se nespustí.
3. Spusťte uživatelské procedury cíle spuštění. Pokud se ukončení cíle nezdaří, přenos selže a nebude dále spuštěno žádné další spuštění.
4. Spusťte volání před místem určení (je-li k dispozici). Pokud volání před místem určení selže, přenos selže a nic dalšího se nespustí.
5. Proveďte přenosy souborů.
6. Spusťte uživatelské procedury ukončení cíle. Pro tyto výstupy není k dispozici žádný stav selhání.
7. Je-li přenos úspěšný (pokud se některý přenos úspěšně přenesl, přenos je považován za úspěšný), spusťte volání po cíli (je-li přítomno). Pokud se volání po cíli nezdaří, přenos selže.
8. Spusťte uživatelské procedury ukončení. Pro tyto výstupy není k dispozici žádný stav selhání.
9. Je-li přenos úspěšný, spusťte post-source volání (pokud existuje). Pokud se volání po zdrojovém kódu nezdaří, přenos selže.

Příklady

Tento příklad ukazuje přesun základního souboru mezi `agent1` a `agent2`. Příkaz pro zahájení přesunu souboru je odeslán správci front s názvem `qm0`, pomocí připojení v režimu transportu klienta. Výsledek operace přenosu souboru 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 v zotavení

Související úlohy

Použití Apache Ant s MFT

fte: ignoreoutcome Ant úloha

Ignoruje se výsledek příkazu **`fte:filecopy`**, **`fte:filemove`**nebo **`fte:call`** . Když zadáte úlohu **`fte:filecopy`**, **`fte:filemove`**nebo **`fte:call`** na výsledek `defer`, úloha Ant alokuje prostředky pro sledování tohoto výsledku. Pokud již nejste zainteresovaní na výsledku, můžete použít úlohu **`fte:ignoreoutcome`** k uvolnění těchto prostředků.

Atributy

id

Povinné Identifikuje výsledek, který již není předmětem zájmu. Obvykle určujete tento identifikátor pomocí vlastnosti, kterou jste nastavili pomocí atributu `idproperty` u úlohy “fte: úloha filecopy Ant” na stránce 2089, “fte: fileove Ant úloha” na stránce 2093nebo “fte: volání úlohy Ant” na stránce 2086 .

Příklad

Tento příklad ukazuje, jak můžete použít úlohu fte: ignoreoutcome k uvolnění prostředků přidělených na sledování výsledku předchozí úlohy “fte: úloha filecopy Ant” na stránce 2089 .

```
<!-- 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: příkaz ping Ant

Tato úloha produktu IBM MQ Managed File Transfer Ant testuje spojení s agentem za účelem získání odezvy, a proto určuje, zda je agent schopen zpracovávat 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 který se má odeslat požadavek **`fte:ping`** . Hodnota je ve tvaru: *`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, do kterého má být odeslán požadavek. Zadejte tyto informace ve tvaru *`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 naslouchá správce front
- *`channel`* je volitelný kanál SVRCONN, který má být použit

Vynecháte-li informace o *`host`*, *`port`*nebo *`channel`* pro správce front příkazů, použije se informace o připojení zadané v souboru `command.properties` .

Upozornění: Není-li uvedena žádná hodnota pro:

- použití proměnné *host* , režim vazeb je použit
- proměnná *port* , použije se hodnota 1414.
- Proměnná *kanál* , SYSTEM.DEF.SVRCONN se používá.

Další informace viz Soubor MFT command.properties .

Atributy však nemůžete přeskočit ve středu, například `qmgrname@host@@channel`. Můžete mít, například `qmgrname@host`, nebo `qmgrname@host@portnebo qmgrname@hostport@@channel`.

MFT rozdělí daný atribut s použitím oddělovače `@` . V závislosti na počtu nalezených tokenů je použit první token jako *qmgrname*, druhý jako *host*, třetí jako *port* a nakonec *channel*.

Další informace viz Soubor MFT command.properties.

Vlastnost **`com.ibm.wmqfte.propertySet`** můžete použít k uvedení, který soubor `command.properties` použít. Další informace viz com.ibm.wmqfte.propertySet.

Pokud nepoužijete atribut cmdqm , úloha standardně použije vlastnost `com.ibm.wmqfte.ant.commandQueueManager` , je-li tato vlastnost nastavena. Není-li vlastnost `com.ibm.wmqfte.ant.commandQueueManager` nastavena, je proveden pokus o připojení k výchozímu správci front, který je definován v souboru `command.properties` . Formát vlastnosti `com.ibm.wmqfte.ant.commandQueueManager` je stejný jako u atributu cmdqm , tj. `qmgrname@host@port@channel`.

vlastnost rcproperty
Povinné Určuje vlastnost pro uložení návratového kódu 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 mínus jedna může být také zadán tak, že příkaz čeká na odezvu agenta navždy. Pokud není uvedena žádná hodnota pro `timeout` , pak předvolba bude čekat až 5 sekund, než bude agent odpovídat.

Příklad

Tento příklad odešle požadavek **`fte:ping`** na server `agent1` hostovaný produktem `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: uuid Ant úloha

Vygeneruje pseudo-náhodný jedinečný identifikátor a přiřadí jej dané vlastnosti. Tento identifikátor můžete například použít ke generování názvů úloh pro jiné operace přenosu souborů.

Atributy

délka
Povinné Numerická délka identifikátoru UUID, který se má generovat. Tato hodnota délky nezahrnuje délku libovolné předpony zadané argumentem **`prefix`** .

vlastnost
Povinné Název vlastnosti, ke které má být přiřazen vygenerovaný klíč UUID.

Předpona
Volitelné. Předpona, která má být přidána k generovanému identifikátoru UUID. Tato předpona se nezapočítává jako část délky UUID, jak je zadáno argumentem **`length`** .

Příklad

Tento příklad definuje klíč UUID, který začíná písmeny ABC následovanými 16 pseudo-náhodných hexadecimálními znaky. Identifikátor 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: filespec Ant vnořený prvek

Argument **fte:filespec** se používá jako vnořený prvek v jiných úlohách. **fte:filespec** použijte k popisu mapování mezi jedním nebo více zdrojovými soubory, adresáři z/OS nebo datovými sadamia místem určení. Typicky se tento prvek používá při vyjádření sady souborů nebo adresářů z/OS nebo datových sad k přesunu nebo kopírování.

Vnořená podle:

- Úloha fte: filecopy
- Úloha fte: fileove

Atributy specifikace zdroje

Je třeba určit jednu z položek srcfilespec nebo srcqueue.

zdroj_souboru
Uvádí zdroj operace souboru. Hodnota tohoto atributu může zahrnovat zástupný znak.

zdroj_fronty
Uvádí, že zdrojem přenosu je fronta. Přenos přesouvá data ze zpráv uložených ve frontě uvedené tímto atributem. Tento atribut nelze zadat, je-li úloha **fte:filespec** vnořena v rámci úlohy **fte:filecopy** .

Atribut srcqueue není podporován, je-li zdrojovým agentem agent mostu protokolu.

Atributy specifikace cíle

Musíte zadat jeden z adresářů dstdir, dstds, dstfilespace, dstfile, dstqueue nebo dstpds.

dstdir
Určuje adresář jako cíl pro operaci se souborem.

z/OS **dstds**
Určuje datovou sadu, která je nastavena 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
Určuje souborový prostor jako místo určení pro operaci se souborem.

Tento atribut se použije pouze tehdy, je-li cílový agent webový agent IBM MQ 8.0 , který má přístup k souborovému 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

Určuje frontu jako cíl pro soubor pro operaci se zprávami. Do této specifikace můžete volitelně zahrnout název správce front s použitím formátu QUEUE@QUEUEMANAGER. Pokud nezadáte název správce front, použije se správce front cílového agenta, pokud jste nenastavili vlastnost výstupního agenta enableClusterQueueInputna hodnotu true. Je-li vlastnost Output enableClusterQueueInputnastavena na hodnotu true, cílový agent použije standardní procedury produktu IBM MQ k určení místa, kde je fronta umístěna. Je třeba určit platný název fronty, který existuje ve správci front.

Uvedete-li atribut dstqueue , nemůžete uvést atributy srcqueue , protože se tyto atributy navzájem vylučují.

Atribut dstqueue není podporován, je-li cílovým agentem agent mostu protokolu.

Atributy zdrojové volby

srcencoding

Volitelné. Kódování znakové sady použité souborem k přenosu.

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text.` .

Pokud neuvedete atribut srcencoding , znaková sada zdrojového systému se použije pro textové přenosy.

srceol

Volitelné. Oddělovač konce řádku použitého přenášeným souborem. Platné hodnoty jsou:

- CRLF -Použijte znak konce řádku následovaný znakem konce řádku, za nímž následuje znak konce řádku. Tato konvence je typická pro systémy Windows .
- LF -Použijte znak LF jako konec oddělovače řádků. Tato konvence je typická pro systémy UNIX .

Tento atribut můžete zadat pouze tehdy, je-li atribut conversion nastaven na hodnotu `text`. Nezadáte-li atribut srceol , budou textové přenosy automaticky určovat správnou hodnotu na základě operačního systému zdrojového agenta.

z/OS **zdrojkeeptrailingprostory**

Volitelné. Určuje, zda se mají koncové mezery uchovávat na zdrojových záznamech přečtených z datové sady s pevnou délkou v rámci přenosu textového režimu. Platné hodnoty jsou:

- `true` -jsou uchovávány koncové mezery.
- `false` -koncové mezery jsou odděleny.

Pokud nezadáte atribut srckeeptrailingspaces , je určena výchozí hodnota `false` .

Tento atribut můžete zadat pouze v případě, že zadáte také atribut srcfilespec a nastavíte atribut conversion na hodnotu `text.` .

zdrojmsgdelimbytes

Volitelné. Uvádí 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 lze zadat pouze v případě, že jste také zadali atribut srcqueue . Atribut srcmsgdelimbytes nelze zadat, pokud jste také zadali hodnotu `text` atributu conversion .

zdrojmsgdelimtext

Volitelné. Určuje posloupnost textu, která má být vložena jako oddělovač při připojování více zpráv k textovému souboru. Do oddělovače můžete zahrnout escape sekvence Java pro řetězcové literály. Například `srcmsgdelimtext="\u007d\n"`. Oddělovač textu se vloží za každou zprávu ze zdrojového agenta. Oddělovač textu je zakódován do binárního formátu za použití zdrojového kódování přenosu. Každá zpráva je čtena v binárním formátu, zakódovaný oddělovač je připojen v binárním formátu do zprávy a výsledek je přenesen v binárním formátu na cílového agenta. Pokud kódová stránka zdrojového agenta zahrnuje stavy shift-in a shift-out, agent předpokládá, že každá zpráva se

nachází ve stavu shift-out na konci zprávy. V cílovém agentovi jsou binární data převedena stejným způsobem jako přenos souboru na textový přenos. Atribut srcmsgdelimtext lze zadat pouze v případě, že jste také určili atribut srcqueue a hodnotu parametru `text` pro atribut conversion .

zdroj_zprávy_zpráv
Volitelné. Určuje pozici, do které má být vložen text nebo binární oddělovač. Platné hodnoty jsou:

- `prefix` -oddělovače jsou vloženy do cílového souboru před daty z každé zprávy.
- `postfix` -oddělovače jsou vloženy do cílového souboru za daty z každé zprávy.

Atribut srcmsgdelimposition lze zadat pouze v případě, že jste také zadali jeden z atributů srcmsgdelimbytes nebo srcmsgdelimtext .

zdrojzmsggroups
Volitelné. Určuje, že zprávy jsou seskupeny podle ID skupiny IBM MQ . První úplná skupina se zapíše do cílového souboru. Není-li tento atribut zadán, všechny zprávy ve zdrojové frontě se zapíší do cílového souboru. Atribut srcmsggroups lze zadat pouze v případě, že jste také zadali atribut srcqueue .

srcqueuetimeout
Volitelné. Určuje 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 zadán atribut srcmsggroups , bude pro úplnou skupinu, která má být zapsána do fronty.

Pokud není splněna ani jedna z těchto podmínek v době určené hodnotou parametru srcqueuetimeout, zdrojový agent zastaví čtení z fronty a dokončí přenos. Není-li atribut srcqueuetimeout zadán, zastaví zdrojový agent ihned čtení ze zdrojové fronty, je-li zdrojová fronta prázdná, nebo v případě, že je zadán atribut srcmsggroups , pokud ve frontě není žádná úplná skupina. Atribut srcqueuetimeout lze zadat pouze v případě, že jste také zadali atribut srcqueue .

Informace o nastavení hodnoty parametru *srcqueuetimeout* naleznete v tématu Pokyny pro uvedení doby čekání na přenos zpráv 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 zadat pouze v případě, že je zdrojový soubor přenosu orientovaný na záznamy, například datová sada z/OS a cílový soubor je normální, negramotově orientovaný soubor. Atribut srcrecdelimbytes nelze zadat, pokud jste také zadali hodnotu `text` atributu conversion .

srcrecdelimpos
Volitelné. Určuje umístění, do kterého je vložen binární oddělovač. Platné hodnoty jsou:

- prefix-oddělovače jsou vloženy do cílového souboru před daty z každého záznamu souboru na zdrojovém záznamu.
- postfix-oddělovače jsou vloženy do cílového souboru za daty z každého zdrojového záznamu souboru orientovaného na záznam.

Atribut srcrecdelims lze zadat pouze v případě, že jste také zadali atribut srcrecdelimbytes .

Atributy volby cíle

dstenkódování
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.` .

Není-li zadán atribut dstencoding , znaková sada cílového systému se použije pro textové přenosy.

dsteol
Volitelné. Oddělovač konce řádku, který má být použit pro přenášený soubor. Platné hodnoty jsou:

- `CRLF` -Použijte znak konce řádku následovaný znakem konce řádku, za nímž následuje znak konce řádku. Tato konvence je typická pro systémy Windows .
- `LF` -Použijte znak LF jako konec oddělovače řádků. Tato konvence je typická pro systémy UNIX .

Tento atribut můžete zadat pouze v případě, že je atribut conversion nastaven na hodnotu `text.` .

Pokud nezadáte atribut dsteol , budou textové přenosy automaticky určovat správnou hodnotu na základě operačního systému cílového agenta.

dstmsgdelimbytes
Volitelné. Uvádí hexadecimální oddělovač, který se má použít při rozdělování 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 uvedení hexadecimálního bajtu jako oddělovače je `xNN`, kde N je znak v rozsahu 0 až 9 nebo-f. Můžete uvést pořadí hexadecimálních bajtů jako oddělovač uvedením čárkami odděleného seznamu hexadecimálních bajtů, například: `x3e,x20,x20,xbf`.

Atribut dstmsgdelimbytes lze určit pouze v případě, že jste také určili atribut dstqueue a přenos je v binárním režimu. Můžete uvést pouze jednu z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstmsgdelimpattern
Volitelné. Určuje regulární výraz produktu Java , který má být použit při rozdělování 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 ve dvojitých uvozovkách, "*`regular_expression`*". Další informace viz Regulární výrazy používané MFT.

Při výchozím nastavení je délka řetězce, kterou může regulární výraz porovnat, omezena cílovým agentem na pět znaků. Toto chování můžete změnit pomocí vlastnosti agenta **`maxDelimiterMatchLength`** . Další informace najdete v tématu Rozšířené vlastnosti agenta MFT.

Atribut dstmsgdelimpattern lze určit pouze v případě, že jste také zadali atribut dstqueue a přenos je v textovém režimu. Můžete uvést pouze jednu z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstmsgdelimposition
Volitelné. Určuje umístění, ve kterém se očekává, že text nebo binární oddělovač bude uvnitř. Platné hodnoty jsou:

- `prefix` -Oddělovače se očekávají na začátku každého řádku.
- `postfix` -Oddělovače se očekávají na konci každého řádku.

Atribut dstmsgdelimposition lze zadat pouze v případě, že jste také zadali atribut dstmsgdelimpattern .

dstmsgincludedelim
Volitelné. Uvádí, zda zahrnout oddělovač, který se používá k rozdělení souboru do více zpráv ve zprávách. Je-li zadán atribut dstmsgincludeddelim , oddělovač je obsažen na konci zprávy, která obsahuje data, která předcházela oddělovači. Ve výchozím nastavení není oddělovač obsažen ve zprávách. Atribut dstmsgincludedelim můžete zadat pouze v případě, že jste také zadali jednu z atributů dstmsgdelimpattern a dstmsgdelimbytes .

dstmsgpersist
Volitelné. Určuje, zda jsou zprávy zapsané do cílové fronty trvalé. Platné hodnoty jsou:

- `true` -Zapisovat trvalé zprávy do cílové fronty. Toto je výchozí hodnota.
- `false` -Zápis netrvalých zpráv do cílové fronty.

- qdef -Hodnota perzistence je převzata z atributu DefPersistence cílové fronty.

Tento atribut můžete zadat pouze tehdy, je-li zadán také atribut dstqueue .

dstmsgprops
Volitelné. Určuje, zda má přenos první zprávy zapsané do cílové fronty sadou vlastností zprávy produktu IBM MQ . Možné hodnoty jsou:

- true -Nastavte vlastnosti zprávy u první zprávy vytvořené přenosem.
- false -nenastavujte vlastnosti zprávy na první zprávě vytvořené přenosem. Toto je výchozí hodnota.

Další informace naleznete v tématu Vlastnosti zprávy produktuMQ nastavené uživatelem MFT na zprávách zapsaných do cílových front.

Tento atribut můžete zadat pouze tehdy, je-li zadán 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 uvedena v hodnotě dstmsgsize. Formát parametru dstmsgsize je *délkajednotek*, kde *délka* je kladné celé číslo a *jednotky* jsou jednou z následujících hodnot:

- B -Bajty. Povolená minimální hodnota je dvakrát větší než maximální hodnota počtu bajtů na znak kódové stránky cílových zpráv.
- K -Kibibajty. To je ekvivalentní 1024 bajtům.
- M -Mebibajty. To je ekvivalent 1024 kibibajtů.

Je-li soubor přenášen v textovém režimu a nachází se v dvoubajtové znakové sadě nebo vícebajtové znakové sadě, rozdělí se soubor na zprávy na nejbližší hranici znaku na zadanou velikost zprávy.

Atribut dstmsgsize můžete zadat pouze v případě, že jste také zadali atribut dstqueue . Můžete uvést pouze jednu z atributů dstmsgsize, dstmsgdelimbytesa dstmsgdelimpattern .

dstunsupportedcodepage
Volitelné. Určuje akci, která má být provedena v případě, že správce cílové fronty, jak je určen atributem dstqueue , nepodporuje kódovou stránku použitou při přenosu dat souboru do fronty jako přenos textu. Platné hodnoty pro tento atribut jsou následující:

- binary -pokračujte v přenosu, ale neaplikujte konverzi kódové stránky na přenášená data. Zadání této hodnoty je ekvivalentní nastavení nenastavení atributu převodu na hodnotu text.
- fail -nepokračujte v operaci přenosu. Soubor se zaznamená jako nepřenášené. Toto nastavení je výchozí.

Atribut dstunsupportedcodepage můžete zadat pouze v případě, že jste také zadali atribut dstqueue a hodnotu text pro atribut conversion .

z/OS **dsttruncateordy**
Volitelné. Určuje, že cílové záznamy delší, než je atribut datové sady LRECL, jsou oříznuty. Je-li hodnota nastavena na true, záznamy se oseknou. Je-li nastaveno na hodnotu false, jsou záznamy zalomeny. Výchozí nastavení je false. Tento parametr je platný pouze pro přenosy v textovém režimu, kde je cílem datová sada.

Další atributy

checksum
Volitelné. Určuje algoritmus použitý pro kontrolní součet přenesených souborů.

- MD5 -použijte hašovací algoritmus MD5 .
- NONE -nepoužívat algoritmus kontrolního součtu.

Pokud nezadáte atribut checksum , použije se výchozí hodnota MD5 .

konverze
Volitelné. Určuje typ převodu, který má být použit pro soubor při jeho přenosu. Možné hodnoty jsou:

- `binary` -nelze použít žádný převod.
- `text` -použití konverze kódové stránky mezi zdrojovým a cílovým systémem. Uplatní také převod oddělovačů řádků. Atributy `srcencoding`, `dstencoding`, `srceol` a `dsteol` ovlivňují konverzi, která se použije.

Pokud neuvedete atribut conversion , zadá se předvolená hodnota `binary` .

overwrite
Volitelné. Určuje, zda může být existující cílový soubor z/OS nebo datovou sadu přepsán operací. Zadáte-li hodnotu `true`, budou přepsány všechny existující cílové soubory z/OS nebo datové sady . Uvedete-li hodnotu `false`, existence duplicitního souboru z/OS nebo datové sady v cíli má za následek selhání operace. Není-li atribut overwrite zadán, je určena výchozí hodnota `false` .

rekurze
Volitelné. Určuje, zda se přenos souboru rekurzí do podadresářů. Zadáte-li hodnotu `true`, bude přenos rekurzí do podadresářů. Zadáte-li hodnotu `false`, nebude přenos rekurzivně procházet do podadresářů. Není-li atribut recurse zadán, je zadána výchozí hodnota `false` .

Příklad

Tento příklad uvádí parametr 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: Vnořené prvky Ant metadat

Metadata se používají k přenášení dalších uživatelem definovaných informací s operací přenosu souborů.

Další informace o tom, jak produkt Managed File Transfer používá metadata, naleznete v příručce “Metadata pro uživatelské procedury produktu MFT” na stránce 2108 .

Vnořená podle:

- Úloha fte: filecopy
- Úloha fte: fileove
- Úloha fte: call

Parametry zadané jako vnořené prvky

fte: položka
Musíte zadat alespoň jednu položku uvnitř vnořeného prvku fte:metadata . Můžete určit více než jednu položku. Položky přidružují název klíče k hodnotě. Klíče musí být v bloku fte:metadata jedinečné

Atributy položky

name
Povinné Název klíče, který patří do této položky. Tento název musí být jedinečný v rámci všech parametrů **`entry`** vnořených uvnitř prvku fte: metadata .

hodnota
Povinné Hodnota, která se má přiřadit k této položce.

Příklad

Tento příklad ukazuje definici fte:metadata , která obsahuje dva záznamy.

```
<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 mohou být spuštěny 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 svého zpracování. Než budete moci spustit program, musíte se ujistit, že příkaz je v umístění uvedeném ve vlastnosti commandPath v souboru `agent.properties` agenta, který spouští příkaz.

I když má každý prvek vyvolání programu jiný název, sdílejí stejnou sadu atributů a stejnou sadu vnořených prvků. Programy lze spustit pomocí úloh Ant **`fte:filecopy`**, **`fte:filemove`**a **`fte:command`** .

Nemůžete vyvolat programy z agenta mostu Connect:Direct .

Úlohy Ant, které mohou vyvolat programy:

- Úloha fte: filecopy hnízdí parametry vyvolání programu s použitím vnořených prvků `fte:predst`, `fte:postdst`, `fte:presrca` `fte:postsrc` .
- Úloha fte: fileove hnízdí parametry vyvolání programu s použitím vnořených prvků `fte:predst`, `fte:postdst`, `fte:presrca` `fte:postsrc` .
- Úloha fte: call hnízdí parametry vyvolání programu pomocí vnořeného prvku `fte:command` .

Atributy

příkaz
: Povinné Určuje jméno programu, který má být volal. Aby agent mohl spustit příkaz, příkaz musí být v umístění uvedeném vlastností commandPath v souboru `agent.properties` agenta. Další informace viz vlastnost commandPath MFT. Všechny informace o cestě zadané v atributu `command` se považují za relativní k umístění uvedenému ve vlastnosti commandPath . Je-li `type` `executable`, je očekáván spustitelný program, jinak se očekává skript odpovídající typu volání.

retrycount
: Volitelné. Počet pokusů o opakování volání programu, pokud program nevrací návratový kód úspěchu. Program pojmenovaný atributem `command` je volán až do tohoto počtu opakování. Hodnota přiřazená k tomuto atributu musí být nezáporná. Pokud atribut `retrycount` nezadáte, bude použita výchozí hodnota nula.

retrywait
: Volitelné. Doba čekání, v sekundách, před opakováním pokusu o vyvolání programu. Pokud program uvedený atributem `command` nevrací návratový kód úspěchu a atribut `retrycount` určuje nenulová hodnota, tento parametr určuje dobu čekání mezi novými pokusy. Hodnota přiřazená k tomuto atributu musí být nezáporná. Pokud atribut `retrywait` nezadáte, bude použita výchozí hodnota nula.

úspěšný
: 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 může být složena z jednoho nebo více výrazů spojených se svislou čárovým znakem (|) pro označení logického operátoru ORnebo ampersand (&). znaků, které označují logickou spojku AND. Každý výraz může být jeden z následujících typů výrazu:

- Číslo pro označení testu rovnosti mezi návratovým kódem procesu a číslem.
- Číslo s předponou ">" označuje great-than test mezi číslem a návratovým kódem procesu.
- Číslo s předponou "<" označuje méně než test mezi číslem a návratovým kódem procesu.
- Číslo s předponou "!" písmeno označující nerovnající se testu mezi číslem a návratovým kódem procesu.

Například: >2&<7&!5|0|14 se interpretuje jako následující návratové kódy, které jsou úspěšné: 0, 3, 4, 6, 14. Všechny ostatní návratové kódy jsou interpretovány jako neúspěšné. Pokud atribut `successrc` nezadáte, bude použita výchozí hodnota nula. To znamená, že příkaz je považován za úspěšný, pokud a pouze tehdy, vrátí-li kód nula.

typ

Volitelné. Hodnota tohoto atributu určuje, jaký typ programu se volá. Uveďte jednu z následujících možností:

Spustitelné

Úloha volá spustitelný program. Může obsahovat další argumenty určené pomocí vnořeného prvku arg . Očekává se, že program bude přístupný na commandPath a tam, kde je to vhodné, prováděcí oprávnění k provedení. Skripty UNIX lze volat tak dlouho, jak 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. Množství výstupních dat je však omezeno konfigurací agenta. Výchozí hodnota je 10K bajtů dat, ale toto výchozí nastavení můžete přepsat pomocí vlastnosti agenta: maxCommandOutput.

ant_skript

Úloha spustí zadaný skript Ant pomocí příkazu **`fteAnt`** . Vlastnosti lze zadat s použitím vnořeného prvku property . Cíle Ant lze zadat s použitím vnořeného prvku target . Očekávájí se, že skript Ant je přístupný na commandPath. Výstup příkazu Ant zapsaný do stderr nebo stdout se odešle do protokolu Managed File Transfer pro volání. Množství výstupních dat je však omezeno konfigurací agenta. Výchozí hodnota je 10K bajtů dat, ale toto výchozí nastavení můžete přepsat pomocí vlastnosti agenta: maxCommandOutput.

z/OS **JCL**

The value `jcl` is supported on z/OS only and runs the specified z/OS JCL script. JCL se odešle jako úloha a vyžaduje přítomnost zakázkového listu. Je-li úloha úspěšně odeslána výstupu příkazu JCL, zapsána do protokolu Managed File Transfer , obsahuje následující text: JOB *název_úlohy*(*id_úlohy*), kde:

- *název_úlohy* je název úlohy označené zakázkový list v souboru JCL.
- *jm_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 vědět, zda úloha byla úspěšně spuštěna nebo byla úspěšně dokončena, použijte systémovou službu, jako např. SDSF. Produkt Managed File Transfer neposkytuje informace, protože pouze odesílá úlohu; systém pak určí, kdy má spustit úlohu a jak se bude prezentovat výstup úlohy. Vzhledem k tomu, že skript JCL je zadán jako dávková úloha, nedoporučuje se zadávat `jcl` pro vnořený prvek `presrc` nebo `predst` , protože víte, že úloha byla úspěšně odeslána a ne, zda byla před spuštěním přenosu úspěšně dokončena úspěšně. Neexistují žádné vnořené prvky, které jsou platné s typem `jcl`.

Následující příklad zobrazuje 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 zadané jako vnořené prvky

fte: arg
Platné pouze v případě, že hodnota atributu `type` je executable. Chcete-li zadat argumenty programu, který je volán jako součást vyvolání programu, použijte vnořené prvky `fte:arg` . Argumenty programu jsou sestaveny z hodnot zadaných prvky `fte:arg` v pořadí, ve kterém jsou rozpoznány prvky `fte:arg` . Jako vnořené prvky vyvolání programu můžete zvolit, zda mají být zadány nuly nebo více prvků `fte:arg` .

fte: vlastnost
Platí pouze tam, kde hodnota atributu `type` je antscript. Použijte atributy `name` a `value` vnořených prvků `fte:property` , které se předají do skriptu Ant ve formě názvu a hodnoty. Jako vnořené prvky vyvolání programu můžete zvolit, zda mají být zadány nuly nebo více prvků `fte:property` .

fte: target
Platí pouze tam, kde hodnota atributu `type` je antscript. Uveďte cíl ve skriptu Ant , který se má volat. Jako vnořené prvky vyvolání programu můžete zvolit, zda mají být zadány nuly nebo více prvků `fte:target` .

Atributy Arg

hodnota
Povinné Hodnota argumentu pro předání do volaného programu.

Atributy vlastností

název
Povinné Název vlastnosti, která má být předána skriptu Ant .

hodnota
Povinné Hodnota, která se má přidružit k názvu vlastnosti, která se předává skriptu Ant .

Příklady

Tento příklad ukazuje vyvolání programu `fte:postsrc` , které je určeno jako součást úlohy fte: filecopy . Programové vyvolání se používá pro program nazvaný `post.sh` a je dodáván jeden argument `/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 zadáno jako část úlohy fte: call . Vyvolání programu je pro spustitelný soubor s názvem `command.sh`, který nepředává žádné argumenty příkazového řádku. Pokud `command.sh` nevrátí návratový kód úspěchu 1, je příkaz znovu zkoušn 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 zadáno jako část úlohy fte: call . Vyvolání programu je pro cíle kopírování a komprimace ve skriptu Ant nazvaném `script.xml`, který prošel dvěma vlastnostmi.

```
<fte:call cmdqm="qm0@localhost@1414@SYSTEM.DEF.SVRCONN"
            agent="agent1@qm1"
            rcproperty="call.rc"
            originuser="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

Zadání programů pro spouštění s produktem MFT

Použití Apache Ant s MFT

Uživatelské procedury produktu MFT pro odkaz na přizpůsobení

Referenční informace, které vám pomohou nakonfigurovat uživatelské procedury pro Managed File Transfer.

Související odkazy

Zdrojový a cílový uživatelský vstup produktu MFT

Metadata pro uživatelské procedury produktu MFT

Existují tři různé typy metadat, které lze dodat uživatelským ukončovacím rutinám pro metadata prostředí 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 ukončovacím rutinám a popisuje prostředí běhové komponenty agenta, ze kterého se volá uživatelská procedura. Tato metadata jsou jen pro čtení a nelze ji aktualizovat žádnou uživatelskou procedurou.

Tabulka 884. Metadata prostředí

Klíč	**Popis**
KLÍČ_KONFIGURACE_AGENTA	Název adresáře, který obsahuje informace o konfiguraci agenta.
KLÍČ_ADRESÁŘ_AGENT_PRODUKTU	Název adresáře, ve kterém byl nainstalován kód agenta.
KLÍČ AGENT_VERSION_KEY	Číslo verze pro běhovou komponentu agenta, která 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 ukončovacím rutinám. Metadata se skládají ze systémem dodaných hodnot a hodnot dodaných uživatelem. Pokud změníte hodnoty dodané systémem, tyto změny se budou ignorovat. Počáteční uživatelem zadané hodnoty pro uživatelskou proceduru spuštění přenosu zdroje jsou založeny na hodnotách, které zadáte při definování přenosu. Zdrojový agent může měnit uživatelem zadané hodnoty jako součást zpracování uživatelské procedury spuštění

přenosu zdroje. Tato uživatelská procedura se volá před zahájením celého přenosu souboru. Tyto změny se používají při následných voláních do jiných ukončovacích rutin, které se vztahují k tomuto přenosu. Metadata přenosu se použijí na celý přenos.

Ačkoli všechny uživatelské procedury mohou číst hodnoty z metadat přenosu, mohou metadata přenosu změnit pouze spouštěcí uživatelská procedura přenosu zdroje.

Metadata přenosu nelze použít k šíření informací mezi různými přenosy souborů.

Metadata převodu dodaná systémem jsou podrobně popsána v tabulce č. 2:

Tabulka 885. Metadata přenosu

Klíč	**Popis**
CÍLOVÝ_KLÍČ AGENTA	Název agenta, který je cílem přenosu.
KLÍČ_ÚLOHY_	Název úlohy přidružený k požadavku na přenos
KLÍČ MQM_USER_KEY	Pole uživatele MQMD ze zprávy použité k odeslání požadavku na přenos
PŮVODNÍ_KLÍČ_HOSTITELE	Název hostitele zadaný jako původní název hostitele v požadavku na přenos
KLÍČ PŮVODNÍHO_UŽIVATELE	Jméno uživatele zadané jako původní ID uživatele v požadavku na přenos
KLÍČ AGENTA_ZDROJE	Název agenta, který je zdrojem přenosu
KLÍČ TRANSFER_ID_	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 do výstupní uživatelské procedury přenosu zdroje jako součást specifikace souboru. Pro zdrojový a cílový soubor existují samostatná metadata souboru.

Metadata souboru nelze použít k šíření informací mezi různými přenosy souborů.

Tabulka 886. Metadata souboru

Klíč	**Povolené hodnoty**	**Popis**
CONVERT_LINE_SEPARATORS		Hodnota klíče používaná pro textové přenosy sloužící k označení, zda jsou posloupnosti oddělovačů řádků CRLF (carriage return-line feed) nebo LF (line feed) ve zdrojových datech převedeny na pořadí oddělovače řádků v místě určení.
ODDĚLOVAT_KLÍČ		Hodnota klíče použitá k definování oddělovače pro oddělení dat záznamu při přenosu dat orientovaných na záznamy do normálních souborů. Používá se také pro přenosy zpráv do souboru a do souboru zpráv.
KLÍČ DELIMITER_POSITION_KEY	DELIMITER_POSITION_PREFIX_VALUE DELIMITER_POSITION_POSTFIX_VALUE	Použijte spolu s DELIMITER_KEY k definování pozice oddělovače; buď předpony, nebo postfix.

Tabulka 886. Metadata souboru (pokračování)

Klíč	Povolené hodnoty	Popis
DELIMITERY_TYPE_KEY	HODNOTA BINARYTYPE_BINARY_VALUE HODNOTA DELIMITER_TYPE_TEXT_VALUE DELIMITER_TYPE_SIZE_VALUE	Použijte spolu s DELIMITER_KEY k definování typu oddělovače.
KLÍČ_EXISTUJÍCÍ_CÍLE	DESTINATION_EXIST_KEY_ERROR_VALUE DESTINATION_EXIST_KEY_OVERWRITE_VALUE	Určuje chování přenosu souborů v případě, že cílový soubor existuje.
ALIAS_ALIASU_SOUBORU		Hodnota klíče použitá k definování aliasu pro přenášený soubor.
KEY_CHECKSUM_METHOD_KEY	FILE_CHECKSUM_METHOD_NOT_VALUE FILE_CHECKSUM_METHOD_MD5_VALUE	Určuje metodu kontrolního součtu, která má být použita při přenosu souboru.
FILE_CONVERSION_SOUBORU	NÁZEV_SOUBORU_CONVERSION_TEXT_TEXTOVÉHO_SOUBORU CONVERSION_BINARY_VALUE	Určuje typ převodu použitý na obsah souboru.
FILE_ENCODING_KEY		Určuje kódování použité pro textový soubor.
FILE_END_OF_LINE_KEY	FILE_END_OF_LINE_LF_VALUE FILE_END_OF_LINE_CRLF_VALUE	Určuje posloupnost znaků, která určuje konec řádku: < LF > nebo < CR> < LF>.
ALIAS_PROSTORU_SOUBORŮ		Určuje alias souboru v souborovém prostoru. **Poznámka:** Tato metadata lze použít pouze v případě, že proměnná FILE_TYPE_KEY je FILE_TYPE_FILE_SPACE_VALUE.
NÁZEV_PROSTORU_SOUBORŮ		Určuje název souborového prostoru. **Poznámka:** Tato metadata lze použít pouze v případě, že proměnná FILE_TYPE_KEY je FILE_TYPE_FILE_SPACE_VALUE.
TEXT_TYPU_SOUBORU	FILE_TYPE_FILE_TYPE_TYPE_DIRECTORY_VALUE FILE_TYPE_DATASET_VALUE FILE_TYPE_PDS_VALUE TYP_SOUBORU_SOUB_TYP_SOUBORU_SOUBOR_S_HODNOTA_SOUB_S_SOUB_SOUB	Určuje cílový soubor, frontu nebo specifikaci souborového prostoru.
KLÍČ_ID_SKUPINY		Hodnota klíče použitá pro přenosy zpráv do souboru k určení skupiny zpráv, které se mají přečíst ze zdrojové fronty. Tento atribut je platný pouze v případě, že hodnota USE_GROUPS_KEY je USE_GROUPS_TRUE_VALUE.
INCLUDE_DELIMITER_IN_MESSAGE_KEY	INCLUDE_DELIMITER_IN_MESSAGE_TRUE_VALUE INCLUDE_DELIMITER_IN_MESSAGE_FALSE_VALUE	Hodnota klíče používaná pro přenos souboru na zprávu k určení, zda zahrnout oddělovače použité k rozdělení souboru do více zpráv na konci zpráv. Tento atribut je platný pouze v případě, že hodnota DELIMITER_TYPE_KEY je DELIMITER_TYPE_BINARY_VALUE DELIMITER_TYPE_TEXT_VALUE.
VLOŽENÝ_ŘETĚZEC_ODDĚLOVAČ_NA_ZÁZNAM_ŘÁDKU		Hodnota klíče používaná pro textové přenosy ze souborů orientovaných na záznamy k určení, zda se oddělovače řádků vloží do dat za každým záznamem.

Tabulka 886. Metadata souboru (pokračování)

Klíč	Povolené hodnoty	Popis
KLÍČ KEEP_TRAILING_SPACE_KEY	KEEP_TRAILING_SPACE_TRUE_VALUE KEEP_TRAILING_SPACE_FALSE_VALUE	Hodnota klíče použitá k určení, zda jsou koncové mezery odebrány ze záznamů čtených z datových sad s formátem pevnou délkou.
NEW_RECORD_ON_LINE_SEPARATOR _KEY		Hodnota klíče používaná pro textové přenosy do souborů orientovaných na záznamy k určení, zda jsou oddělovače řádků v datech zahrnuty do dat záznamu nebo způsobí nový záznam (a nejsou zapsány).
PERSISTENT_KEY	PERSISTENT_TRUE_HODNOTA HODNOTA_TRVAČENÍ_PERZISTENCE HODNOTA_TRVAČENÍ_PAMĚTI	Hodnota klíče používaná pro přenos souboru k přenosu zpráv za účelem určení, zda jsou zprávy trvalé.
KLÁVESA_MQ_PROPS_KEY	ZA_PRAVNÍ_HODNOTA_SADY_SET_K_ZADÁNÍ HODNOTA SET_MQ_PROPS_FALSE_VALUE	Hodnota klíče používaná pro přenos souboru k přenosu zpráv za účelem určení, zda jsou vlastnosti zprávy IBM MQ nastaveny na první zprávě v souboru, a všechny zprávy zapsané do fronty, když dojde k chybě.
UNRECOGNISED_CODE_PAGE_KEY	NEROZPOZNANÁ KÓDOVÁ STRÁNKA UNRECOGNISE_CODE_PAGE_VALUE UNRECOGNISED_CODE_PAGE_BINARY_VALUE	Hodnota klíče používaná pro přenos souboru k přenosu zpráv za účelem určení, zda převod textového režimu selže nebo je proveden převod, pokud není kódová stránka dat rozpoznána cílovým správcem front.
KLÍČ USED_GROUPS_KEY	POUŽITÉ_SKUPINY_TRUE_VALUE USE_GROUPS_FALSE_VALUE	Hodnota klíče používaná pro přenosy zpráv do souboru k určení, zda má být přenesena pouze úplná skupina zpráv ze zdrojové fronty.
ČÍSLO_ČASU ČEKÁNÍ		Hodnota klíče používaná pro přenos zpráv do souboru k určení času (v sekundách), po který má zdrojový agent čekat na jeden z následujících případů: • Zpráva, která se má zobrazit ve zdrojové frontě, pokud je fronta prázdná nebo se stala prázdnou, pokud je hodnota USE_GROUPS_KEY FALSE. • Úplná skupina, která se zobrazí ve zdrojové frontě, je-li hodnota USE_GROUPS_KEY TRUE.

Názvy klíčů a názvy hodnot uvedené v tabulce 3 jsou konstanty, které jsou definovány v rozhraní FileMetaDataConstants .

Uživatelské procedury monitoru prostředků produktu MFT

Uživatelské procedury monitoru prostředků vám umožňují konfigurovat vlastní kód, který se má spustit, když je splněna podmínka spouštěče monitoru, dříve než je spuštěna přidružená úloha.

Nedoporučuje se vyvolávat nové přenosy přímo z uživatelského kódu. Za určitých okolností způsobí, že soubory budou přeneseny vícekrát, protože uživatelské procedury nejsou odolné vůči restartování agenta.

Uživatel monitoru prostředků ukončí práci s existující infrastrukturou pro uživatelské procedury. Uživatelské procedury monitoru jsou volány poté, co byl spuštěn monitor, ale předtím, než byla úloha monitoru spuštěna odpovídající úlohou. To umožňuje uživatelské proceduře upravit úlohu, která má být spuštěna, a rozhodnout, zda má úloha pokračovat či nikoli. Úlohu monitorování můžete upravit aktualizací metadat monitoru, která se poté použije pro nahrazení proměnných v dokumentu úlohy vytvořeném vytvořením původního monitoru. Alternativně může uživatelská procedura monitoru nahradit nebo aktualizovat řetězec XML definice úlohy předaný jako parametr. Uživatelská procedura monitoru může vrátit kód výsledku buď 'pokračovat', nebo 'zrušení' pro úlohu. Je-li vráceno zrušení, úloha nebude spuštěna a monitor se znovu nespustí, dokud se monitorovaný prostředek neshoduje s podmínkami spouštěče. Pokud se prostředek nezměnil, trigger se nespustí. Stejně jako u ostatních uživatelských procedur, můžete společně sledovat ukončení řetězce. Pokud jeden z uživatelských procedur vrátí kód výsledku zrušení, celkový výsledek je stornovací a úloha se nespustí.

- Mapa metadat prostředí (stejná jako jiná uživatelská procedura)
- Mapa metadat monitoru včetně neměnných metadat systému a metadat mutable user. 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 splnil podmínku spouštěče
  - LASTMODIFIEDDATE (Local)-Datum, kdy byl soubor, který splnil podmínku spouštěče, naposledy změněn. Tento datum se vyjadřuje jako místní datum časového pásma, ve kterém je agent spuštěný, a je naformátovaný na datum dle normy ISO 8601.
  - LASTMODIFIEDTIME (Lokální)-čas v lokálním formátu, že soubor, který splnil podmínku spouštěče, byl 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-datum v univerzálním formátu, že soubor, který vyhovuje podmínce 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-čas 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 reprezentující úlohu, která má být spuštěna jako výsledek spouštěče monitoru.

Uživatelské procedury monitoru vrátí následující data:

- Indikátor, který určuje, zda pokračovat v dalším postupu (pokračovat nebo zrušit)
- Řetězec, který se má vložit do zprávy protokolu s uspokojením spouštěče

V důsledku spuštění kódu uživatelské procedury monitorování mohou být aktualizovány také metadata monitoru a řetězec XML definice úlohy, které byly původně předány jako parametry.

Hodnota vlastnosti agenta monitorExitClasses (v souboru `agent.properties`) uvádí, které třídy uživatelských procedur monitoru se mají načíst, přičemž každá výstupní třída je oddělena čárkou. Příklad:

```
monitorExitClasses=testExits.TestExit1,testExits.testExit2
```

Rozhraní pro uživatelskou proceduru 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 vyhrazené pro systém IBMv 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 vzorová třída implementuje rozhraní MonitorExit . Tento příklad přidá vlastní substituční proměnnou do metadat monitoru s názvem *REDIRECTEDAGENT* , která bude naplněna hodnotou LONDON , pokud je hodina dne lichá, a hodnota PARIS po dobu i hodin. Výsledkový kód ukončení monitoru je nastaven tak, aby vždy vracel 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 využívá substituční proměnnou *REDIRECTEDAGENT* , by mohla vypadat podobně jako následující:

```
<?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 <destinationAgent> prvku `agent` nahradí buď `LONDON` nebo `PARIS`.

Musíte zadat substituční proměnnou ve třídě uživatelské procedury monitoru a XML definice úlohy velkými písmeny.

Související pojmy

“Metadata pro uživatelské procedury produktu MFT” na stránce 2108
Existují tři různé typy metadat, které lze dodat uživatelským ukončovacím rutinám pro metadata prostředí Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

“Rozhraní Java pro uživatelské procedury produktu MFT” na stránce 2118
Pomocí témat v této sekci najdete referenční informace o rozhraních produktu Java pro uživatelské výstupní rutiny.

Zdrojový a cílový uživatelský vstup produktu MFT

Související úlohy

Přizpůsobení MFT s uživatelskými procedurami

Související odkazy

“Vlastnosti agenta MFT pro uživatelské procedury” na stránce 2115
Kromě standardních vlastností v souboru `agent.properties` existuje několik rozšířených vlastností speciálně pro uživatelské procedury. Tyto vlastnosti nejsou ve výchozím nastavení zahrnuty, takže pokud chcete použít některý z nich, musíte ručně upravit soubor `agent.properties` . Pokud provedete změnu do souboru `agent.properties` během spuštění tohoto agenta, zastavte a restartujte agenta, aby se projevily změny.

Vlastnosti agenta MFT pro uživatelské procedury

Kromě standardních vlastností v souboru `agent.properties` existuje několik rozšířených vlastností speciálně pro uživatelské procedury. Tyto vlastnosti nejsou ve výchozím nastavení zahrnuty, takže pokud chcete použít některý z nich, musíte ručně upravit soubor `agent.properties` . Pokud provedete změnu

do souboru `agent.properties` během spuštění tohoto agenta, zastavte a restartujte agenta, aby se projevily změny.

Pro IBM WebSphere MQ 7.5 nebo pozdější mohou být proměnné prostředí použity ve vlastnostech produktu Managed File Transfer , které představují umístění souboru nebo adresáře. To umožňuje umístění souborů nebo adresářů používaných při spouštění částí produktu, aby se lišily v závislosti na změnách prostředí, jako např. který uživatel spouští proces. Další informace najdete v tématu 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 887. Vlastnosti agenta pro uživatelské procedury

Název vlastnosti	**Popis**
Třídy sourceTransferEndExit	Určuje seznam tříd, které implementují zdrojovou uživatelskou proceduru ukončení přenosu, oddělených čárkami.
Třídy sourceTransferStartExit	Určuje seznam tříd oddělených čárkami, které implementují uživatelskou proceduru spuštění přenosu zdroje.
Třídy destinationTransferStartExit	Určuje seznam tříd oddělených čárkami, které implementují uživatelskou proceduru spuštění cílové uživatelské procedury přenosu.
Třídy destinationTransferEndExit	Určuje seznam tříd oddělených čárkami, které implementují uživatelskou proceduru přenosu místa určení.
Cesta exitClass	Určuje seznam adresářů, které se chovají jako cesta ke třídě pro uživatelské procedury, jako cestu ke třídám určuje seznam adresářů. Před položkami v této cestě ke třídě je prohledán adresář agenta. Používáte-li tuto vlastnost v systému Windows, použijte jako oddělovač cesty znak lomítka (/), nikoli zpětné lomítko (\). Například: `exitClassPath=C:/mycomp/mqft/exits/encryptFileExit.jar; C:/mycomp/mqft/exits/fileFilter.jar.` Hodnota této vlastnosti pro produkt IBM WebSphere MQ 7.5 nebo novější může obsahovat proměnné prostředí.
exitNativeLibraryPath	Určuje seznam adresářů specifických pro jednotlivé platformy a adresářů, které se chovají jako cesta k nativní knihovně pro uživatelské procedury. Hodnota této vlastnosti pro produkt IBM WebSphere MQ 7.5 nebo novější 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

Zdroj řetězení a ukončení místa určení

Zadáte-li více uživatelských procedur, bude první ukončení v seznamu vyvoláno jako první, za nímž následuje druhá uživatelská procedura atd. Všechny změny provedené první uživatelskou procedurou budou předány jako vstup pro výstup, který je následně vyvolán, atd. Je-li například spuštěn dva spuštění přenosu zdroje, budou veškeré změny v metadatech přenosu provedené první uživatelskou procedurou vstupem do druhé uživatelské procedury. Každá uživatelská procedura vrátí svůj vlastní výsledek. Pokud všechny konce daného typu vrátí PROCED jako kód výsledku přenosu, je celkový výsledek PROCEED. Pokud jeden nebo více uživatelských procedur vrátí CANCEL_TRANSFER, celkový výsledek je CANCEL_TRANSFER. Všechny výsledné kódy a řetězce vrácené výstupem jsou výstupem v protokolu přenosu.

Je-li celkový výsledek z výstupní uživatelské procedury přenosu zdroje PROCEED, přenos bude pokračovat s použitím veškerých změn provedených východy. Je-li celkový výsledek CANCEL_TRANSFER, budou vyvolány ukončení ukončení přenosu zdroje a poté přenos bude zrušen. Stav dokončení v protokolu přenosu je "zrušeno".

Je-li celkový výsledek z ukončení cílového přenosu ukončen PROCEED, přenos pokračuje s použitím změn provedených východy. Je-li celkový výsledek CANCEL_TRANSFER, vyvolají se koncové uživatelské procedury přenosu cíle, poté se vyvolá ukončení konce přenosu zdroje. Nakonec je přenos zrušen. Stav dokončení v protokolu přenosu je "zrušeno".

Pokud zdrojový nebo cílový výstupní bod potřebuje předat informace následujícím uživatelským procedurám buď v řetězci, nebo v pořadí provedení, musí se provést aktualizací metadat přenosu. Použití metadat přenosu je specifické pro implementaci ukončení. Pokud například při výstupu dojde k vrácení výsledku vratky do CANCEL_TRANSFER a je třeba předat následující uživatelské procedury, že přenos byl zrušen, musí být proveden nastavením hodnoty metadat přenosu způsobem srozumitelným pro ostatní ukončení.

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 MFT s uživatelskými procedurami

“Metadata pro uživatelské procedury produktu MFT” na stránce 2108
Existují tři různé typy metadat, které lze dodat uživatelským ukončovacím rutinám pro metadata prostředí Managed File Transfer: prostředí, přenos a metadata souboru. Tato metadata jsou prezentována jako mapy dvojic klíč-hodnota Java .

“Rozhraní Java pro uživatelské procedury produktu MFT” na stránce 2118
Pomocí témat v této sekci najdete referenční informace o rozhraních produktu Java pro uživatelské výstupní rutiny.

Související odkazy

“Uživatelské procedury monitoru prostředků produktu MFT” na stránce 2111

Uživatelské procedury monitoru prostředků vám umožňují konfigurovat vlastní kód, který se má spustit, když je splněna podmínka spouštěče monitoru, dříve než je spuštěna přidružená úloha.

Proměnné prostředí ve vlastnostech produktu MFT
Soubor MFT `agent.properties`

Rozhraní Java pro uživatelské procedury produktu MFT

Pomocí témat v této sekci najdete referenční informace o rozhraních produktu Java pro uživatelské výstupní rutiny.

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 2146

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2146

“SourceTransferEndExit.java” na stránce 2145

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2146

“SourceTransferEndExit.java” na stránce 2145

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2146

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2146

“SourceTransferEndExit.java” na stránce 2145

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 produktu MFT” na stránce 2108

Existují tři různé typy metadat, které lze dodat uživatelským ukončovacím rutinám pro metadata prostředí 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 2146

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2143

“SourceTransferEndExit.java” na stránce 2145

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

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 2146

“DestinationTransferStartExitrozhraní .java” na stránce 2121

“DestinationTransferEndExit.java” na stránce 2120

“Rozhraní MonitorExit.java” na stránce 2139

“ProtocolBridgeCredentialExit.java” na stránce 2140

Formáty zpráv pro zprávy, které můžete vložit do fronty příkazů agenta MFT

Tato schémata XML definují formáty zpráv, které lze vložit do fronty příkazů agenta, aby bylo možné požadovat, aby agent provedl akci. Zpráva XML může být umístěna do fronty příkazů agenta pomocí příkazů příkazového řádku nebo aplikací.

- Formát zprávy požadavku na přenos souborů
- Formáty zpráv požadavku monitoru MFT
- Formát zprávy požadavku agenta PING MFT
- Formát zprávy odpovědi agenta MFT

Odkaz systému zpráv REST API

Referenční informace o produktu messaging REST API.

Další informace o použití nástroje messaging REST APInaleznete v tématu Systém zpráv pomocí produktu REST API.

REST API - prostředky

Tato kolekce témat obsahuje referenční informace o každém z prostředků produktu messaging REST API .

Další informace o použití nástroje messaging REST APInaleznete v tématu Systém zpráv pomocí produktu REST API.

/messaging/qmgr/{qmgrName}/queue/{queueName}/message

Rozhraní API REST zpráv umožňuje vkládání zpráv do fronty, V 9.2.0 nebo zpráv, které mají být procházeny nebo destruktivně získáno z fronty pomocí prostředku `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` .

POST

Pomocí metody HTTP POST s prostředkem produktu `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` lze zprávy vložit do zadané fronty v určeném správci front.

Přepne zprávu IBM MQ obsahující tělo požadavku HTTP do zadaného správce front a fronty. Správce front musí být umístěn ve stejném počítači jako server mqweb. Metoda podporuje pouze těla požadavků HTTP založená na textu. Zprávy se posílají jako formátované zprávy produktu `MQSTR` nebo `JMS TextMessage` a používají se aktuální kontext uživatele.

V 9.2.5 Rozhraní API služby REST V3 přidá schopnost uvést vlastnosti zprávy definované uživatelem a zahrnout prioritu zpráv. Hlavičky požadavků ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním API REST V3. Záhlaví požadavku ibm-mq-md-correlationId má v rozhraní REST API V3jiný formát. Hlavička může být ID specifické pro aplikaci, nebo, pokud je zakódovaný řetězec, vyžaduje předponu `ID:` . Pokud váš požadavek POST obsahuje uživatelsky definované zprávy nebo ID korelace specifické pro aplikaci, bude zpráva formátována jako `JMS TextMessage`.

Adresa 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 9.2.5 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

qmgrName

Určuje název správce front pro připojení k systému zpráv. Správce front musí být umístěn ve stejném počítači jako server mqweb.

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 kódováno jako %2F.
- Období musí být zakódováno jako %2E.
- Znak procenta musí být zakódován jako %25.

queueName

Uvádí název fronty, na kterou se má vložit zpráva.

Fronta musí být definována jako lokální, vzdálená nebo alias pro uvedeného správce front-může se také odkazovat na klastrovanou frontu.

Název fronty je citlivý na velikost písmen.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znaménko procent,%, musí být zakódováno jako %25.

Povolíte-li připojení HTTP, můžete místo HTTPS použít protokol HTTP. Další informace o povolení HTTP najdete v tématu Konfigurace portů HTTP a HTTPS.

Záhlaví požadavku

Následující záhlaví musí být odeslána s požadavkem:

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`

Poznámka: Je-li *charset* vynechána ze záhlaví `Context-Type` , předpokládá se `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 spolu s požadavkem:

Accept-Language
Toto záhlaví uvádí 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. Hlavička musí být uvedena jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů. Nepřipojujte hodnotu k hodnotě `"ID:"`, rozhraní REST API tento řetězec přidá automaticky.

Příklad:

```
ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.2.5 **ibm-mq-md-correlationId**
Toto záhlaví nastavuje ID korelace vytvořené zprávy. ID korelace může mít jednu z následujících forem:

- 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů s předponou `"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
```

Určíte-li tento tvar ID korelace, bude cíl zprávy zaměřen jako WMQ_CLIENT_JMS_COMPLEANT, a proto bude obsahovat záhlaví MQRFH2 .

ibm-mq-md-expiry

Toto záhlaví nastavuje dobu vypršení platnosti pro vytvořenou zprávu. Ukončení 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 nevypršela.

Tato hodnota je výchozí hodnotou.

Celočíselná hodnota

Milisekundy před vypršením zprávy.

Omezeno na rozsah 0-99999999900.

ibm-mq-md-persistence

Toto záhlaví nastavuje perzistenci pro vytvořenou zprávu. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

nonPersistent

Zpráva nepřežije selhání systému nebo správce front se restartuje.

Tato hodnota je výchozí hodnotou.

Trvalý

Zpráva přežije selhání systému nebo správce front se restartuje.

REST API | V3 | V 9.2.5 **ibm-mq-md-priority**

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 nastavuje cíl odpovědi pro vytvořenou zprávu. Formát záhlaví používá standardní notaci pro zadání fronty pro odpovědi a nepovinného správce front: `replyQueue[@replyQmgr]`

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

REST API | V3 | V 9.2.5 **ibm-mq-usr**

Nastavte uživatelem definované vlastnosti zprávy požadavku. Na zprávě může být nastaveno více vlastností. V jednoduchém záhlaví požadavku ibm-mq-usr můžete zadat více hodnot oddělených čárkami, 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 vlastnosti uživatele. 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` (32bitové reálné, MQFLOAT32)
- `double` (64-bitový real, MQFLOAT64)
- `string` (uzavřený řetězec)

Poznámka: Priorita zprávy pro test POST je vždy 4 pro rozhraní API služby REST V1 a rozhraní REST API V2 , standardně je nastavena na `asDestination` pro rozhraní REST API V3.

Formát těla požadavku

Tělo požadavku musí být text a použít kódování UTF-8 . Není vyžadována žádná specifická textová struktura. Vytvoří se `MQSTR` formátovaná zpráva obsahující text těla požadavku a umístí se do uvedené fronty.

REST API | V3 | V 9.2.5 Jsou-li použity uživatelem definované vlastnosti rozhraní API služby REST V3 nebo jsou použity funkce ID korelace specifické pro aplikaci, vytvoří se formátovaná zpráva JMS `TextMessage` obsahující text těla požadavku a vloží se do zadané fronty.

Další informace viz příklady.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nejsou použitelné pro messaging REST API. Další informace o zabezpečení pro REST API viz Zabezpečení konzoly IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak i administrative REST API.

Činitel zabezpečení volajícího musí mít možnost 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, která je uvedena v části *{queueName}* adresy URL prostředku, musí být oprávnění +PUT uděleno činiteli zabezpečení volajícího.
- z/OS Pro frontu, která je určena částí adresy URL prostředku *{queueName}* , musí být udělen přístup UPDATE činitele zabezpečení volajícího.

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.

Pokud používáte produkt Advanced Message Security (AMS) s produktem messaging REST API, všimněte si, že všechny zprávy jsou šifrovány pomocí kontextu mqweb serveru, nikoli kontextu uživatele, který tuto zprávu odesílá.

Stavové kódy odezvy

201
Zpráva byla úspěšně vytvořena a odeslána.

400
Byla poskytnuta neplatná data.

Například byla uvedena 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 zadáno záhlaví `ibm-mq-rest-csrf-token` . Další informace naleznete v části "Požadavky na zabezpečení" na stránce 2152.

403
Neautorizováno.

Volající je ověřován na webovém serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem, ani k podmnožině požadovaných prostředků IBM MQ , nebo se nenachází v roli `MQWebUser` . Další informace o požadovaném přístupu viz "Požadavky na zabezpečení" na stránce 2152.

404
Fronta neexistuje.

405
Fronta je blokována PUT.

415
Hlavička zprávy nebo tělo je nepodporovaným typem média.

Například záhlaví `Content-Type` je nastaveno na nepodporovaný typ média.

500
Problém serveru 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

Následující záhlaví jsou vrácena s odezvou:

Obsah-Jazyk
Určuje identifikátor jazyka pro zprávu odezvy v případě výskytu chyb nebo výjimek. Používá se v kombinaci se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakoukoli chybu nebo výjimku. Výchozí hodnota parametru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP i v případě, že neexistuje žádný obsah. Při úspěchu je hodnota nula.

Content-Type
Určuje typ těla odezvy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě výskytu 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 přiřazeno IBM MQ této zprávě. Stejně jako záhlaví požadavku `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 9.2.5 **ibm-mq-md-messageId**

Uvádí ID zprávy, které je přiřazeno 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 řetězcem `ID:`.

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

Formát těla odezvy

Pokud je zpráva úspěšně odeslána, tělo odezvy je prázdné. Dojde-li k chybě, tělo odezvy obsahuje chybovou zprávu. Další informace viz Ošetření chyb produktuREST API.

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 se přihlásí uživatele s názvem `mquser` s heslem `mquser`. V souboru cURLmůže protokol v požadavku vypadat podobně jako v následujícím příkladu produktu Windows . Token LTPA je uložen v souboru `cookiejar.txt` s použitím parametru `-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
```

After the user is logged in, the LTPA token and `ibm-mq-rest-csrf-token` HTTP header are used to authenticate further requests. Hodnota `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL odesílá zprávu do fronty `Q1` ve správci front `QM1`s 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 odesílá trvalou zprávu do fronty `Q1` ve správci front `QM1`, s vypršením platnosti 2 minuty. 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 netrvalou zprávu do fronty `Q1` ve správci front `QM1`bez 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:
  ```

```
414d5120514d414445562020202020202067d8b
f5923582e02" --data "Hello World!"
```

V 9.2.0 *GET*

Pomocí metody GET protokolu HTTP s prostředkem produktu `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` můžete procházet zprávy z asociovaného správce front a fronty.

Prochází první dostupnou zprávou z uvedeného správce front a fronty. Správce front musí být umístěn ve stejném počítači jako server mqweb. Tělo zprávy je vráceno v těle odpovědi 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 se vrátí odpovídající stavový kód pro jakékoli nevhodné zprávy. Příklad: Zpráva, která nemá formát `MQSTR` nebo `JMS TextMessage` .

V 9.2.5 Rozhraní API služby REST V3 má přidanou schopnost specifikovat uživatelem definované vlastnosti zpráv a zahrnovat prioritu zpráv se zprávami. Záhlaví odezvy ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze v rozhraní REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má v rozhraní REST API V3jiný formát. Hlavička může být ID specifické pro aplikaci nebo, pokud je zakódovaný řetězec, zachovává předponu `ID:` . Záhlaví odezvy ibm-mq-md-messageId a parametr dotazu má v rozhraní REST API V3jiný formát, zachovává si předponu `ID:` .

- “Adresa URL prostředku” na stránce 2155
- “Volitelné parametry dotazu” na stránce 2156
- “Záhlaví požadavku” na stránce 2157
- “Formát těla požadavku” na stránce 2157
- “Požadavky na zabezpečení” na stránce 2157
- “Stavové kódy odezvy” na stránce 2157
- “Záhlaví odezvy” na stránce 2158
- “Formát těla odezvy” na stránce 2160
- “Příklady” na stránce 2160

Adresa 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 9.2.5 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

qmgrName

Určuje název správce front pro připojení k systému zpráv. Správce front musí být umístěn ve stejném počítači jako server mqweb.

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 procházet zpráva.

Fronta musí být definována jako lokální nebo alias, který ukazuje na lokální frontu.

Název fronty je citlivý na velikost písmen.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znaménko procent,%, musí být zakódováno jako %25.

Povolíte-li připojení HTTP, můžete místo HTTPS použít protokol HTTP. Další informace o povolení HTTP najdete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V1 REST API V2 **correlationId=*hexValue***

Určuje, že metoda HTTP vrátí následující zprávu s příslušným ID korelace.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.2.5 **correlationId= ID:*hexValue* nebo correlationId=*specificapplication_application_value***

Určuje, že metoda HTTP vrátí následující zprávu s příslušným ID korelace.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

hodnota_specifikace_aplikace

Parametr dotazu může být zadán 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 příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.2.5 **messageId= ID:*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d414445562020202020067d8ce5923582f07
```

Záhlaví požadavku

Následující záhlaví musí být odeslána s požadavkem:

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 spolu s požadavkem:

Přijmout-Znaková sada
Toto záhlaví lze použít k určení, jaká znaková sada je přijatelná pro odezvu. Je-li toto záhlaví uvedeno, musí být nastaveno jako UTF-8.

Accept-Language
Toto záhlaví uvádí 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` nejsou použitelné pro messaging REST API. Další informace o zabezpečení pro REST API viz Zabezpečení konzoly IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak i 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.

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 poskytnuta neplatná data.

Např. byla uvedena 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 zadáno záhlaví `ibm-mq-rest-csrf-token` . Další informace naleznete v části “Požadavky na zabezpečení” na stránce 2157.

403
Neautorizováno.

Volající je ověřován na webovém serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem, ani k podmnožině požadovaných prostředků IBM MQ , nebo se nenachází v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2157.

404
Fronta neexistuje.

500
Problém serveru nebo kód chyby z IBM MQ.

501
Odpověď HTTP nebyla konstruována.

Například přijatá zpráva má chybný typ nebo má správný typ, ale tělo nebylo možné zpracovat.

502
Aktuální činitel zabezpečení nemůže obdržet zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

Následující záhlaví jsou vrácena s odezvou:

Obsah-Jazyk
Určuje identifikátor jazyka pro zprávu odezvy v případě výskytu chyb nebo výjimek. Používá se v kombinaci se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakoukoli chybu nebo výjimku. Výchozí hodnota parametru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP i v případě, že neexistuje žádný obsah. Hodnota obsahuje délku (bajty) dat zprávy.

Content-Type
Určuje typ obsahu vráceného v těle odpovědi přijaté zprávy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě výskytu chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

ibm-mq-md-correlationId
Určuje ID korelace přijaté zprávy. Záhlaví je vráceno, pokud přijatá zpráva obsahuje platné ID korelace. Je představován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-correlationId: 414d5120514d414445562020202020202067d8bf5923582e02
```

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, který představuje 24 bajtů s předponou `"ID:"`. Příklad:

```
ibm-mq-md-correlationId: ID:414d5120514d414445562020202020067d8bf5923582e02
```

- Hodnota specifická pro aplikaci. Hodnota je řetězec specifický pro aplikaci:

```
ibm-mq-md-correlationId: My-Custom-CorrelId
```

ibm-mq-md-expiry
Určuje zbývající dobu platnosti přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

bez omezení
Platnost zprávy nevypršela.

Celočíselná hodnota
Zbývající milisekundy před ukončením platnosti zprávy.

REST API V1 REST API V2 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiřazeno 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: 414d5120514d414445562020202020067d8ce5923582f07
```

REST API V3 V 9.2.5 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiřazeno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentováno jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů s předponou `"ID:"`

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d414445562020202020067d8ce5923582f07
```

ibm-mq-md-persistence
Určuje trvání přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo správce front se restartuje.

Trvalý
Zpráva přežije selhání systému nebo správce front se restartuje.

REST API V3 V 9.2.5 **ibm-mq-md-priority**
Vrátí nastavení priority zprávy. Příklad:

```
ibm-mq-md-priority: 3
```

ibm-mq-md-replyTo
Určuje odpověď na místo určení přijaté zprávy. Formát záhlaví používá standardní notaci pro frontu pro odpověď a správce front `replyQueue@replyQmgr`.

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

REST API V3 V 9.2.5 **ibm-mq-usr**
Vrací uživatelem definované vlastnosti zprávy. Na zprávě 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 vlastnosti uživatele. 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` (32bitové reálné, MQFLOAT32)
- `double` (64-bitový real, MQFLOAT64)
- `string` (uzavřený řetězec)

Formát těla odezvy

Při úspěchu obsahuje tělo odezvy tělo zprávy z přijaté zprávy. Dojde-li k chybě, tělo odezvy obsahuje chybovou zprávu zformátovanou ve formátu JSON. Obě odpovědi jsou UTF-8 kódované. Další informace viz Ošetření chyb produktuREST API.

Uvědomte si, že při přijetí zprávy jsou podporovány pouze formátované zprávy IBM MQ `MQSTR` nebo JMS `TextMessage` .

Procházení fronty, která byla označena jako metoda GET, nevrací žádný obsah.

Pokud procházená fronta obsahuje zprávy s duplicitními identifikátory zpráv, je při filtrování na identifikátoru zprávy vrácena 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 se přihlásí uživatele s názvem `mquser` s heslem `mquser`. V souboru cURLmůže protokol v požadavku vypadat podobně jako v následujícím příkladu produktu Windows . Token LTPA je uložen v souboru `cookiejar.txt` s použitím parametru `-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
```

After the user is logged in, the LTPA token and `ibm-mq-rest-csrf-token` HTTP header are used to authenticate further requests. Hodnota `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad produktu Windows cURL prochází další dostupnou zprávou z fronty Q1 ve správci front QM1s 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 prohlédne zprávu se specifickým ID korelace 000000000000000000000000000000000000000000abcdabcd, z fronty Q1 ve správci front QM1:

```
curl -k "https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/QM1/queue/Q1/message?
correlationId=000000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: text/plain"
```

- REST API V3 V 9.2.5

  Následující příklad Windows cURL je stejný jako předchozí příklad, ale používá rozhraní REST API V3. Příklad vypíše pouze zprávy s odpovídajícími ID korelace 000000000000000000000000000000000000000000abcdabcd, z fronty Q1 ve správci front QM1:

```
curl -k "https://localhost:9443/ibmmq/rest/v3/messaging/qmgr/QM1/queue/Q1/message?
correlationId=ID:000000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: text/plain"
```

DELETE

K získání zpráv z asociovaného správce front a fronty můžete použít metodu HTTP DELETE s prostředkem `/messaging/qmgr/{qmgrName}/queue/{queueName}/message` .

Destruktivně získá další dostupnou zprávu z uvedeného správce front a fronty a vrátí tělo zprávy do těla odezvy HTTP. Správce front musí být umístěn ve stejném počítači jako server mqweb. Zpráva musí mít formát `MQSTR` nebo `JMS TextMessage`a přijímá se s použitím aktuálního uživatelského kontextu.

Nekompatibilní zprávy jsou ponechány ve frontě a volajícímu se vrátí odpovídající stavový kód. Příklad: Zpráva, která nemá formát `MQSTR` nebo `JMS TextMessage` .

V 9.2.5 Rozhraní API služby REST V3 má přidanou schopnost specifikovat uživatelem definované vlastnosti zpráv a zahrnovat prioritu zpráv se zprávami. Záhlaví odezvy ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze v rozhraní REST API V3. Záhlaví požadavku ibm-mq-md-correlationId má v rozhraní REST API V3jiný formát. Hlavička může být ID specifické pro aplikaci nebo, pokud je zakódovaný řetězec, zachovává předponu `ID:` . Záhlaví odezvy ibm-mq-md-messageId a parametr dotazu má v rozhraní REST API V3jiný formát, zachovává si předponu `ID:` .

Adresa 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 9.2.5 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/message`

qmgrName
Určuje název správce front pro připojení k systému zpráv. Správce front musí být umístěn ve stejném počítači jako server mqweb.

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.

Název fronty je citlivý na velikost písmen.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znaménko procent,%, musí být zakódováno jako %25.

Povolíte-li připojení HTTP, můžete místo HTTPS použít protokol HTTP. Další informace o povolení HTTP najdete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V1 REST API V2 **correlationId=*hexValue***
Určuje, že metoda HTTP vrátí následující zprávu s příslušným ID korelace.

hexValue
Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../message?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.2.5 **correlationId= ID:*hexValue* nebo correlationId=*specificapplication_application_value***
Určuje, že metoda HTTP vrátí následující zprávu s příslušným ID korelace.

hexValue
Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

hodnota_specifikace_aplikace
Parametr dotazu může být zadán 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 příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.2.5 **messageId= ID:*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d4144455620202020202067d8ce5923582f07
```

wait=*integerValue*

Uvádí, že metoda HTTP bude čekat *integerValue* milisekund, aby se další zpráva stala dostupnou.

integerValue

Parametr dotazu musí být zadán jako celočíselná hodnota představující trvání milisekund. Maximální hodnota je 2147483647.

Příklad:

```
../message?wait=120000
```

Záhlaví požadavku

Následující záhlaví musí být odeslána s požadavkem:

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 spolu s požadavkem:

Přijmout-Znaková sada

Toto záhlaví lze použít k určení, jaká znaková sada je přijatelná pro odezvu. Je-li toto záhlaví uvedeno, musí být nastaveno jako UTF-8.

Accept-Language

Toto záhlaví uvádí 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` nejsou použitelné pro messaging REST API. Další informace o zabezpečení pro REST API viz Zabezpečení konzoly IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak i administrative REST API.

Činitel zabezpečení volajícího musí mít udělenou schopnost získat zprávy z uvedené 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.

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 poskytnuta neplatná data.

Např. byla uvedena 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 zadáno záhlaví `ibm-mq-rest-csrf-token` . Další informace naleznete v části “Požadavky na zabezpečení” na stránce 2164.

403
Neautorizováno.

Volající je ověřován na webovém serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem, ani k podmnožině požadovaných prostředků IBM MQ , nebo se nenachází v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2164.

404
Fronta neexistuje.

405
Fronta je blokována GET.

500
Problém serveru nebo kód chyby z IBM MQ.

501
Odpověď HTTP nebyla konstruována.

Například přijatá zpráva má chybný typ nebo má správný typ, ale tělo nebylo možné zpracovat.

502
Aktuální činitel zabezpečení nemůže obdržet zprávu, protože poskytovatel systému zpráv nepodporuje požadovanou funkci. Například, pokud je cesta ke třídě serveru mqweb neplatná.

503
Správce front není spuštěn.

Záhlaví odezvy

Následující záhlaví jsou vrácena s odezvou:

Obsah-Jazyk
Určuje identifikátor jazyka pro zprávu odezvy v případě výskytu chyb nebo výjimek. Používá se v kombinaci se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakoukoli chybu nebo výjimku. Výchozí hodnota parametru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP i v případě, že neexistuje žádný obsah. Hodnota obsahuje délku (bajty) dat zprávy.

Content-Type
Určuje typ obsahu vráceného v těle odpovědi přijaté zprávy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě výskytu 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 představován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

Příklad:

```
ibm-mq-md-correlationId: 414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.2.5 **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, který představuje 24 bajtů s předponou `"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-expiry
Určuje zbývající dobu platnosti přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

bez omezení
Platnost zprávy nevypršela.

Celočíselná hodnota
Zbývající milisekundy před ukončením platnosti zprávy.

REST API V1 REST API V2 **ibm-mq-md-messageId**
Uvádí ID zprávy, které je přiřazeno 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 9.2.5 **ibm-mq-md-messageId**

Uvádí ID zprávy, které je přiřazeno IBM MQ této zprávě. Stejně jako záhlaví `ibm-mq-md-correlationId` je reprezentováno jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů s předponou `"ID:"`

Příklad:

```
ibm-mq-md-messageId: ID:414d5120514d4144455620202020202067d8ce5923582f07
```

ibm-mq-md-persistence

Určuje trvání přijaté zprávy. Záhlaví může mít jednu z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo správce front se restartuje.

Trvalý
Zpráva přežije selhání systému nebo správce front se restartuje.

REST API V3 V 9.2.5 **ibm-mq-md-priority**

Vrátí nastavení priority zprávy. Příklad:

```
ibm-mq-md-priority: 3
```

ibm-mq-md-replyTo

Určuje odpověď na místo určení přijaté zprávy. Formát záhlaví používá standardní notaci pro frontu pro odpověď a správce front `replyQueue@replyQmgr`.

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

REST API V3 V 9.2.5 **ibm-mq-usr**

Vrací uživatelem definované vlastnosti zprávy. Na zprávě 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 vlastnosti uživatele. 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` (32bitové reálné, MQFLOAT32)
- `double` (64-bitový real, MQFLOAT64)
- `string` (uzavřený řetězec)

Formát těla odezvy

Při úspěchu obsahuje tělo odezvy tělo zprávy z přijaté zprávy. Dojde-li k chybě, tělo odezvy obsahuje chybovou zprávu zformátovanou ve formátu JSON. Obě odpovědi jsou UTF-8 kódované. Další informace viz Ošetření chyb produktuREST API.

Uvědomte si, že při přijetí zprávy jsou podporovány pouze formátované zprávy IBM MQ `MQSTR` a JMS `TextMessage` . Následně jsou všechny zprávy přijaty pod synchronizačním bodem a všechny neošetřené zprávy jsou ponechány ve frontě. Frontu IBM MQ lze nakonfigurovat tak, aby přesunula tyto nezpracovatelné zprávy do alternativního cíle. Další informace naleznete v tématu Zpracování nezpracovatelných zpráv v produktu IBM MQ classes for 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 se přihlásí uživatele s názvem `mquser` s heslem `mquser`. V souboru cURLmůže protokol v požadavku vypadat podobně jako v následujícím příkladu produktu Windows . Token LTPA je uložen v souboru `cookiejar.txt` s použitím parametru `-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
```

After the user is logged in, the LTPA token and `ibm-mq-rest-csrf-token` HTTP header are used to authenticate further requests. Hodnota `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`a čeká až 30 sekund na to, aby zpráva byla k dispozici. Pokud uplyne 30 sekund, aniž by byla do fronty vložena zadaná zpráva, 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"
  ```

V 9.2.0 /messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist

V 9.2.0 K získání seznamu dostupných zpráv z určené fronty v určeném správci front můžete použít metodu GET protokolu HTTP s prostředkem produktu `/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`.

V 9.2.0 *GET*

K získání seznamu dostupných zpráv z určené fronty v určeném správci front můžete použít metodu GET protokolu HTTP s prostředkem produktu `/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`.

Prochází souhrnným seznamem zpráv ze zadaného správce front a fronty. Správce front musí být umístěn ve stejném počítači jako server mqweb. Souhrnná data jsou vrácena v těle odpovědi HTTP jako pole formátu JSON. Data neobsahují informační obsah zpráv a jsou přijímána s použitím aktuálního kontextu uživatele. Z přidružené fronty nejsou odebrány žádné zprávy.

Je-li vydán požadavek na získání seznamu dostupných zpráv z fronty, která má GET blokováno, je vráceno prázdné pole JSON.

- "Adresa URL prostředku" na stránce 2168
- "Volitelné parametry dotazu" na stránce 2169
- "Záhlaví požadavku" na stránce 2170
- "Formát těla požadavku" na stránce 2170
- "Požadavky na zabezpečení" na stránce 2170
- "Stavové kódy odezvy" na stránce 2171
- "Záhlaví odezvy" na stránce 2171
- "Formát těla odezvy" na stránce 2171
- "Příklady" na stránce 2172

Adresa 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 9.2.5 `https://host:port/ibmmq/rest/v3/messaging/qmgr/{qmgrName}/queue/{queueName}/messagelist`

qmgrName

Určuje název správce front pro připojení k systému zpráv. Správce front musí být umístěn ve stejném počítači jako server mqweb.

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 jméno fronty, ze které se mají procházet zprávy.

Fronta musí být definována jako lokální nebo alias, který ukazuje na lokální frontu.

Název fronty je citlivý na velikost písmen.

Pokud název fronty obsahuje dopředné lomítko nebo znak procenta, musí být tyto znaky zakódovány pomocí adresy URL:

- Dopředné lomítko,/, musí být zakódováno jako %2F.
- Znaménko procent,%, musí být zakódováno jako %25.

Povolíte-li připojení HTTP, můžete místo HTTPS použít protokol HTTP. Další informace o povolení HTTP najdete v tématu Konfigurace portů HTTP a HTTPS.

Volitelné parametry dotazu

REST API V2 **correlationId=*hexValue***

Určuje, že metoda HTTP vrátí následující zprávu s příslušným ID korelace.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../messagelist?correlationId=414d5120514d4144455620202020202067d8bf5923582e02
```

REST API V3 V 9.2.5 **correlationId= ID:*hexValue* nebo correlationId=*specificapplication_application_value***

Určuje, že metoda HTTP vrací seznam zpráv s odpovídajícím ID korelace.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?correlationId=ID:414d5120514d4144455620202020202067d8bf5923582e02
```

hodnota_specifikace_aplikace

Parametr dotazu může být zadán 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 příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, což představuje 24 bajtů.

Příklad:

```
../message?messageId=414d5120514d4144455620202020202067d8ce5923582f07
```

REST API V3 V 9.2.5 **messageId= ID:*hexValue***

Určuje, že metoda HTTP vrátí další zprávu s příslušným ID zprávy.

hexValue

Parametr dotazu musí být zadán jako hexadecimální zakódovaný řetězec o délce 48 znaků, který představuje 24 bajtů a je uveden řetězcem `"ID:"`.

Příklad:

```
../message?messageId=ID:414d5120514d414445562020202020067d8ce5923582f07
```

limit=*integerValue*
Určuje, že tělo odezvy metody HTTP je omezeno na prvky JSON *integerValue* .

integerValue
Parametr dotazu musí být zadán 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

Následující záhlaví musí být odeslána s požadavkem:

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 spolu s požadavkem:

Přijmout-Znaková sada
Toto záhlaví lze použít k určení, jaká znaková sada je přijatelná pro odezvu. Je-li toto záhlaví uvedeno, musí být nastaveno jako UTF-8.

Accept-Language
Toto záhlaví uvádí 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` nejsou použitelné pro messaging REST API. Další informace o zabezpečení pro REST API viz Zabezpečení konzoly IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak i 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.

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 poskytnuta neplatná data.

Např. byla uvedena 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 zadáno záhlaví `ibm-mq-rest-csrf-token` . Další informace naleznete v části “Požadavky na zabezpečení” na stránce 2170.

403
Neautorizováno.

Volající je ověřován na webovém serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem, ani k podmnožině požadovaných prostředků IBM MQ , nebo se nenachází v roli `MQWebUser` . Další informace o požadovaném přístupu viz “Požadavky na zabezpečení” na stránce 2170.

404
Fronta neexistuje.

500
Problém serveru nebo kód chyby z IBM MQ.

501
Odpověď HTTP nebyla konstruována.

Například přijatá zpráva má chybný typ nebo má správný typ, ale tělo nebylo možné zpracovat.

502
Aktuální činitel zabezpečení nemůže obdržet 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 pro zprávu odezvy v případě výskytu chyb nebo výjimek. Používá se v kombinaci se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakoukoli chybu nebo výjimku. Výchozí hodnota parametru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP i v případě, že neexistuje žádný obsah. Hodnota obsahuje délku dat zprávy v bajtech.

Content-Type
Určuje typ těla odezvy. Hodnota je `application/json;charset=utf-8`.

Formát těla odezvy

Po úspěchu je tělo odezvy zakódovanou odezvou UTF-8 . Odezva obsahuje vnější objekt JSON, který obsahuje jedno 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**
Uvádí ID korelace zprávy. Hodnota je vrácena, obsahuje-li zpráva platné ID korelace.

REST API v3 V 9.2.5 **correlationId**
Uvádí ID korelace zprávy. Hodnota je vrácena, obsahuje-li zpráva platné ID korelace. ID korelace je uvozeno řetězcem "ID:", nebo se může jednat o hodnotu specifickou pro aplikaci.

REST API v1 REST API v2 **messageId**
Určuje ID zprávy, které je přiřazeno systémem IBM MQ k této zprávě. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů.

REST API v3 V 9.2.5 **messageId**
Určuje ID zprávy, které je přiřazeno systémem IBM MQ k této zprávě. Je reprezentován jako 48znakový hexadecimální kódovaný řetězec, který představuje 24 bajtů. ID zprávy je vloženo do řetězce "ID:".

formát
Určuje pole formátu MQMD. Za normálních okolností textové zprávy budou obsahovat hodnotu IBM MQ MQSTR .

Je-li zadán požadavek na získání seznamu zpráv ve frontě s vypnutým GET, vrátí se prázdné pole JSON.

Dojde-li k chybě, tělo odezvy obsahuje chybovou zprávu zformátovanou ve formátu JSON. Další informace viz Ošetření chyb produktuREST API.

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 se přihlásí uživatele s názvem `mquser` s heslem `mquser`. V souboru cURLmůže protokol v požadavku vypadat podobně jako v následujícím příkladu produktu Windows . Token LTPA je uložen v souboru `cookiejar.txt` s použitím parametru `-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
```

After the user is logged in, the LTPA token and `ibm-mq-rest-csrf-token` HTTP header are used to authenticate further requests. Hodnota `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL obsahuje seznam dalších deseti dostupných zpráv z fronty Q1 ve správci front QM1s 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"
  ```

- Následující příklad Windows cURL vypíše další dvě stovky dostupných zpráv z fronty Q1 ve správci front QM1s 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 Následující příklad Windows cURL vypíše pouze ty zprávy s příslušným ID korelace `000000000000000000000000000000000000000000abcdabcd`, z 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 9.2.5

  Následující příklad Windows cURL je stejný jako předchozí příklad, ale používá rozhraní REST API V3. Příklad vypíše pouze zprávy s odpovídajícími ID korelace 0000000000000000000000000000000000000000abcdabcd, z fronty Q1 ve správci front QM1:

```
curl -k "https://localhost:9443/ibmmq/rest/v3/messaging/qmgr/QM1/queue/Q1/messagelist?
correlationId=ID:0000000000000000000000000000000000000000abcdabcd"
-X GET -b c:\cookiejar.txt -H "ibm-mq-rest-csrf-token: token-value"
-H "Accept: application/json"
```

V 9.2.0 /messaging/qmgr/{qmgrName}/topic/{topicString}/message

Pomocí metody HTTP POST s prostředkem `/messaging/qmgr/{qmgrName}/topic/{topicString}/message` můžete publikovat zprávy do zadaného tématu v určeném správci front.

V 9.2.0 *POST*

Pomocí metody HTTP POST s prostředkem `/messaging/qmgr/{qmgrName}/topic/{topicString}/message` můžete publikovat zprávy do zadaného tématu v určeném správci front.

Publikuje textově založenou zprávu v těle požadavku HTTP do zadaného správce front a tématu. Správce front musí být na stejném počítači jako server mqweb a jsou podporovány pouze textově podporované zprávy. Zprávy jsou publikovány jako formátované zprávy MQSTR nebo `JMS TextMessage` s použitím aktuálního kontextu uživatele a mají výchozí prioritu zpráv 4 pro rozhraní REST API V2a `asDestination` pro rozhraní REST API V3.

V 9.2.5 Rozhraní API služby REST V3 přidá schopnost uvést vlastnosti zprávy definované uživatelem a zahrnout prioritu zpráv. Hlavičky požadavků ibm-mq-md-priority a ibm-mq-usr jsou k dispozici pouze s rozhraním API REST V3. Záhlaví požadavku ibm-mq-md-correlationId má v rozhraní REST API V3jiný formát. Hlavička může být ID specifické pro aplikaci, nebo, pokud je zakódovaný řetězec, vyžaduje předponu `ID:` . Pokud váš požadavek POST obsahuje uživatelsky definované zprávy nebo ID korelace specifické pro aplikaci, bude zpráva formátována jako `JMS TextMessage`.

- “Adresa URL prostředku” na stránce 2173
- “Záhlaví požadavku” na stránce 2174
- “Formát těla požadavku” na stránce 2176
- “Požadavky na zabezpečení” na stránce 2176
- “Stavové kódy odezvy” na stránce 2176
- “Záhlaví odezvy” na stránce 2177
- “Formát těla odezvy” na stránce 2177
- “Příklady” na stránce 2178

Adresa URL prostředku

`https://`*`host`*`:`*`port`*`/ibmmq/rest/v2/messaging/qmgr/`*`{qmgrName}`*`/topic/`*`{topicString}`*`/message`

V 9.2.5 `https://`*`host`*`:`*`port`*`/ibmmq/rest/v3/messaging/qmgr/`*`{qmgrName}`*`/topic/`*`{topicString}`*`/message`

qmgrName

Určuje název správce front pro připojení k systému zpráv. Správce front musí být umístěn ve stejném počítači jako server mqweb.

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 kódováno jako %2F.
- Období musí být zakódováno jako %2E.
- Znak procenta musí být zakódován jako %25.

topicString
Uvádí řetězec tématu, na kterém se má publikovat zpráva.

Řetězec tématu rozlišuje malá a velká písmena. Řetězec tématu může obsahovat více úrovní témat oddělených dopředným oddělovačem lomítka.

Obsahuje-li řetězec tématu znak procenta, tečku nebo otazník, musí být tyto znaky zakódovány pomocí adresy URL:

- Znak procenta musí být zakódován jako %25.
- Období musí být zakódováno jako %2E.
- Otazník musí být zakódován jako %3F.

Pokud řetězec tématu začíná nebo končí dopředným lomítkem, musí být zakódován pomocí %2F.

Chcete-li například publikovat řetězec tématu, postupujte takto:

- `sport/football` na správci front `MY.QMGR`, použijte následující adresu URL:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/sport/football/message
```

- `/sport/football` na správci front `MY.QMGR`, použijte následující adresu URL:

```
https://localhost:9443/ibmmq/rest/v2/messaging/qmgr/MY%2EQMGR/topic/%2Fsport/football/
message
```

Povolíte-li připojení HTTP, můžete místo HTTPS použít protokol HTTP. Další informace o povolení HTTP najdete v tématu Konfigurace portů HTTP a HTTPS.

Záhlaví požadavku

Následující záhlaví musí být odeslána s požadavkem:

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`

Poznámka: Je-li *charset* vynechána ze záhlaví `Context-Type` , předpokládá se `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 spolu s požadavkem:

Accept-Language
Toto záhlaví uvádí požadovaný jazyk pro všechny výjimky nebo chybové zprávy vrácené v těle zprávy odpovědi.

ibm-mq-md-expiry
Toto záhlaví nastavuje dobu vypršení platnosti pro vytvořenou zprávu. Doba 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 nevypršela.

Tato hodnota je výchozí hodnotou.

Celočíselná hodnota
Milisekundy před vypršením zprávy.

Omezeno na rozsah 0-99999999900.

ibm-mq-md-persistence
Toto záhlaví nastavuje perzistenci pro vytvořenou zprávu. Záhlaví musí být uvedeno jako jedna z následujících hodnot:

nonPersistent
Zpráva nepřežije selhání systému nebo správce front se restartuje.

Tato hodnota je výchozí hodnotou.

Trvalý
Zpráva přežije selhání systému nebo správce front se restartuje.

REST API | V3 | V 9.2.5 **ibm-mq-md-priority**
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 nastavuje cíl odpovědi pro vytvořenou zprávu. Formát záhlaví používá standardní notaci pro zadání fronty pro odpovědi a nepovinného správce front: `replyQueue[@replyQmgr]`

Příklad:

```
ibm-mq-md-replyTo: myReplyQueue@myReplyQMgr
```

REST API | V3 | V 9.2.5 **ibm-mq-usr**
Nastavte uživatelem definované vlastnosti zprávy požadavku. Na zprávě může být nastaveno více vlastností. V jednoduchém záhlaví požadavku ibm-mq-usr můžete zadat více hodnot oddělených čárkami, 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 vlastnosti uživatele. 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` (32bitové reálné, MQFLOAT32)
- `double` (64-bitový real, MQFLOAT64)
- `string` (uzavřený řetězec)

Formát těla požadavku

Tělo požadavku musí být text a použít kódování UTF-8 . Není vyžadována žádná specifická textová struktura. Zformátovaná zpráva `MQSTR` obsahující text požadavku se vytvoří a publikuje do zadaného tématu.

REST API V3 V 9.2.5 Jsou-li použity uživatelem definované vlastnosti rozhraní API služby REST V3 nebo jsou použity funkce ID korelace specifické pro aplikaci, vytvoří se formátovaná zpráva JMS `TextMessage` obsahující text těla požadavku a vloží se do zadané fronty.

Další informace viz příklady.

Požadavky na zabezpečení

Volající musí být ověřen na serveru mqweb. Role `MQWebAdmin` a `MQWebAdminRO` nejsou použitelné pro messaging REST API. Další informace o zabezpečení pro REST API viz Zabezpečení konzoly IBM MQ Console a REST API.

Po ověření na serveru mqweb je uživatel schopen používat jak messaging REST API , tak i administrative REST API.

Činitel zabezpečení volajícího musí mít udělena možnost publikovat zprávy na zadané téma:

- Téma zadané v části *{topicString}* adresy URL prostředku musí být povoleno `PUBLISH` .
- MQ Appliance ALW Pro téma, které je určeno částí *{topicString}* v adrese URL prostředku, musí být oprávnění +PUB uděleno bezpečnostnímu činiteli volajícího.
- z/OS Pro téma, které je určeno částí *{topicString}* v adrese URL prostředku, musí být udělen přístup `UPDATE` k činiteli zabezpečení volajícího.

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.

Pokud používáte produkt Advanced Message Security (AMS) s produktem messaging REST API, všimněte si, že všechny zprávy jsou šifrovány pomocí kontextu mqweb serveru, nikoli kontextu uživatele, který tuto zprávu odesílá.

Stavové kódy odezvy

201
Zpráva byla úspěšně vytvořena a publikována.

400
Byla poskytnuta neplatná data.

Například byla uvedena 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 zadáno záhlaví `ibm-mq-rest-csrf-token` . Další informace naleznete v části "Požadavky na zabezpečení" na stránce 2176.

403
Neautorizováno.

Volající je ověřován na webovém serveru mqweb a je přidružen k platnému činiteli. Činitel však nemá přístup ke všem, ani k podmnožině požadovaných prostředků IBM MQ , nebo se nenachází v roli `MQWebUser` . Další informace o požadovaném přístupu viz "Požadavky na zabezpečení" na stránce 2176.

404
Správce front neexistuje.

405
Téma je PUBLISI blokováno.

415
Hlavička zprávy nebo tělo je nepodporovaným typem média.

Například záhlaví `Content-Type` je nastaveno na nepodporovaný typ média.

500
Problém serveru 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

Následující záhlaví jsou vrácena s odezvou:

Obsah-Jazyk
Určuje identifikátor jazyka pro zprávu odezvy v případě výskytu chyb nebo výjimek. Používá se v kombinaci se záhlavím požadavku `Accept-Language` k označení požadovaného jazyka pro jakoukoli chybu nebo výjimku. Výchozí hodnota parametru mqweb je použita v případě, že požadovaný jazyk není podporován.

Délka obsahu
Určuje délku těla odezvy HTTP i v případě, že neexistuje žádný obsah. Při úspěchu je hodnota nula.

Content-Type
Určuje typ těla odezvy. Při úspěchu je hodnota `text/plain;charset=utf-8`. V případě výskytu chyb nebo výjimek je hodnota `application/json;charset=utf-8`.

Formát těla odezvy

Pokud byla zpráva úspěšně publikována, tělo odezvy je prázdné. Dojde-li k chybě, tělo odezvy obsahuje chybovou zprávu. Další informace viz Ošetření chyb produktuREST API.

Příklady

Následující příklad se přihlásí uživatele s názvem `mquser` s heslem `mquser`. V souboru cURLmůže protokol v požadavku vypadat podobně jako v následujícím příkladu produktu Windows . Token LTPA je uložen v souboru `cookiejar.txt` s použitím parametru `-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
```

After the user is logged in, the LTPA token and `ibm-mq-rest-csrf-token` HTTP header are used to authenticate further requests. Hodnota `ibm-mq-rest-csrf-token` `token_value` může být libovolná hodnota, včetně prázdné hodnoty.

- Následující příklad Windows cURL publikuje zprávu na řetězec 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 na řetězec tématu `myTopic/thisTopic` ve správci front `QM1`s vypršením platnosti 2 minuty. 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 Velké Británie nebo kterékoliv jiné země, kde taková opatření odporují místním zákonům: SPOLEČNOST INTERNATIONAL BUSINESS MACHINES CORPORATION TUTO PUBLIKACI POSKYTUJE TAKOVOU, "JAKÁ 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 VYPLÝVAJÍCÍCH Z OKOLNOSTÍ. 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 spolupráce softwaru, oddělení 49XA
148 00 Praha 4-Chodby

148 00 Praha 4-Chodov
U.S.A.

Poskytnutí takových informací může být podmíněno dodržením určitých podmínek a požadavků zahrnujících v některých případech uhrazení stanoveného poplatku.

IBM poskytuje licencovaný program popsaný v těchto informacích a veškeré dostupné licencované materiály na základě podmínek smlouvy IBM Customer Agreement, IBM International Program License Agreement 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 programátorských rozhraní, jsou-li poskytovány, jsou určeny k tomu, aby vám pomohly vytvořit aplikační software pro použití s tímto programem.

Tato příručka obsahuje informace o zamýšlených programovacích rozhraních, které umožňují zákazníkům 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: Nepoužívejte tyto informace o diagnostice, úpravách a ladění jako programátorské rozhraní, protože se mohou měnit.

Ochranné známky

IBM, logo IBM , ibm.comjsou ochranné známky společnosti IBM Corporation, registrované v mnoha jurisdikcích po celém světě. Aktuální seznam ochranných známek IBM je k dispozici na webu na stránce "Copyright and trademark information"www.ibm.com/legal/copytrade.shtml. Ostatní 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 obsahuje software vyvinutý v rámci projektu Eclipse Project (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: