7.5

IBM WebSphere MQ Vývoj odkazů na aplikace

Poznámka

Než začnete používat tyto informace a produkt, který podporují, přečtěte si informace, které uvádí “Poznámky” na stránce 1433.

Toto vydání se vztahuje k verzi 7, vydání 5 produktu IBM® WebSphere MQ a ke všem následujícím vydáním a modifikacím, dokud nebude v nových vydáních uvedeno jinak.

Když odešlete informace do IBM, udělíte společnosti IBM nevýlučné právo použít nebo distribuovat informace libovolným způsobem, který společnost považuje za odpovídající, bez vzniku jakýchkoliv závazků vůči vám.

Obsah

Odkaz na vývoj aplikací

Informace v této části vám pomohou při vývoji aplikací produktu IBM WebSphere MQ :

Související úlohy

Vývoj aplikací

Odkaz na aplikace MQI

Prostřednictvím odkazů uvedených v této části můžete vyvíjet aplikace MQI s následujícími odkazy:

- "Příklady kódu" na stránce 8
- "Konstanty" na stránce 50
- "Datové typy použité v rozhraní MQI" na stránce 216
- "Volání funkcí" na stránce 583
- "Atributy objektů" na stránce 753
- "Návratové kódy" na stránce 824
- "Pravidla pro ověření platnosti voleb MQI" na stránce 825
- "Kódování počítače" na stránce 848
- "Volby sestav a příznaky zpráv" na stránce 851
- "Uživatelská procedura konverze dat" na stránce 855
- "Vlastnosti zadané jako prvky MQRFH2 ." na stránce 878
- "Převod kódové stránky" na stránce 886

Související pojmy

"Uživatelské procedury, uživatelské procedury rozhraní API a odkazy na instalovatelné služby" na stránce 976

Pomocí odkazů uvedených v této části můžete rozvíjet uživatelské procedury, uživatelské procedury rozhraní API a instalovatelné aplikace služeb:

"Třídy IBM WebSphere MQ pro knihovny Java" na stránce 1377

Umístění tříd produktu IBM WebSphere MQ pro knihovny Java se liší v závislosti na platformě. Uveďte toto umístění, když spustíte aplikaci.

Související úlohy

Vývoj aplikací

Související odkazy

"Odkaz SOAP" na stránce 919

Přenos produktu WebSphere MQ pro referenční informace SOAP uspořádané abecedně.

"Referenční materiál pro most produktu IBM WebSphere MQ pro protokol HTTP" na stránce 1175

Referenční témata pro most IBM WebSphere MQ pro HTTP, která jsou uspořádána abecedně

"Třídy a rozhraní .NET produktuIBM WebSphere MQ" na stránce 1209

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

"IBM WebSphere MQ Třídy C++" na stránce 1270

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

Související informace

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

Příklady kódu

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

Příklady jazyků C

Tato kolekce témat je převážně převzata z ukázkových aplikací produktu WebSphere 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 WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 odebrán z ukázkové aplikace programu Mail Manager (program CSQ4TCD1) dodávaného s produktem WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ. Názvy a umístění ukázkových aplikací najdete v tématu Ukázkové programy (platformy kromě 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) dodávané s produktem WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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) dodávané s produktem WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 produktem WebSphere MQ pro systém 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 WebSphere MQ.

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

     /*---------------------------*/
     /*  Set up GMO structure.    */
     /*---------------------------*/
     memset(&GetMsgOpts,'\0',sizeof(GetMsgOpts));
     memcpy(GetMsgOpts.StrucId, MQGMO_STRUC_ID,
            sizeof(GetMsgOpts.StrucId);
     GetMsgOpts.Version      = MQGMO_VERSION_1;
     GetMsgOpts.WaitInterval = 1000;
     GetMsgOpts.Options      = MQGMO_SET_SIGNAL +
                               MQGMO_BROWSE_FIRST;
     q_ecb              = 0;
     GetMsgOpts.Signal1      = &q_ecb;
     /*---------------------------*/
     /*  Set up MD structure.     */
     /*---------------------------*/
```

```
 memset(&MsgDesc,'\0',sizeof(MsgDesc));
 memcpy(MsgDesc.StrucId, MQMD_STRUC_ID,
        sizeof(MsgDesc.StrucId);
 MsgDesc.Version  = MQMD_VERSION_1;
 MsgDesc.Report   = MQRO_NONE;
 memcpy(MsgDesc.MsgId,MQMI_NONE,
        sizeof(MsgDesc.MsgId));
 memcpy(MsgDesc.CorrelId,MQCI_NONE,
        sizeof(MsgDesc.CorrelId));
```

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

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

```
if (signal_sw == 1)
   {
    endloop = 0;
    do
```

```
          {
           EXEC CICS DELAY FOR HOURS(0) MINUTES(0) SECONDS(1);
           work_ecb = q_ecb & mask;
           switch (work_ecb)
               {
                case (MQEC_MSG_ARRIVED):
                    endloop = 1;
                    mqgmo_options = MQGMO_NO_WAIT;
                    MQGET(Hconn, Hobj, &MsgDesc, &GetMsgOpts,
                          BufferLength, message_buffer,
                          &DataLength, &CompCode, &Reason);
                    if (CompCode != MQCC_OK)
                        ;          /* Perform error processing. */
                    break;
                case (MQEC_WAIT_INTERVAL_EXPIRED):
                case (MQEC_WAIT_CANCELED):
                    endloop = 1;
                    break;
                default:
                    break;
               }
          } while (endloop == 0);
       }
     return;
}
```

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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 s produktem WebSphere MQ pro systémy Windows . Názvy a umístění ukázkových aplikací na jiných platformách naleznete v tématu Ukázkové programy (s výjimkou platforem 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 WebSphere 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 CSQ4BVA1) dodávané s produktem WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 převzat z ukázkové aplikace Procházet (program CSQ4BVA1) dodávané s produktem WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 převzat z ukázkové aplikace Procházet (program CSQ4BVA1) dodávané s produktem WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 převzat z ukázkové aplikace Procházet (program CSQ4BVA1) dodávané s produktem WebSphere MQ pro z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS.

Signalizace je k dispozici pouze s produktem WebSphere MQ pro systém 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 WebSphere MQ pro systém z/OS. Názvy a umístění ukázkových aplikací na jiných platformách najdete v tématu Ukázkové 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 převážně převzata z ukázkových aplikací produktu WebSphere 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 odebrán z ukázkového programu Procházet (CSQ4BAA1), který je dodáván s produktem WebSphere MQ pro systém z/OS.

```
⋮
WORKAREA DSECT
*
PARMLIST CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
COMPCODE DS    F              Completion code
```

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

```
*********************************************************
*  SECTION NAME : MAINCONN                              *
*********************************************************
*
*
MAINCONN DS    0H
         XC    HCONN,HCONN       Null connection handle
*
         CALL  MQCONN,                          X
               (MQMNAME,                        X
               HCONN,                           X
               COMPCODE,                        X
               REASON),                         X
               MF=(E,PARMLIST),VL
*
         LA    R0,MQCC_OK        Expected compcode
         C     R0,COMPCODE       As expected?
         BER   R6                Yes .. return to caller
*
         MVC   INF4_TYP,=CL10'CONNECT   '
         BAL   R7,ERRCODE             Translate error
         LA    R0,8                   Set exit code
         ST    R0,EXITCODE            to 8
         B     ENDPROG                End the program
*
```

Odpojení od správce front

Tento příklad 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 WebSphere MQ.

```
⋮
*
*        ISSUE MQI DISC REQUEST USING REENTRANT FORM
*        OF CALL MACRO
*
*        HCONN WAS SET BY A PREVIOUS MQCONN REQUEST
*        R5 = WORK REGISTER
*
```

```
DISC     DS    0H
         CALL  MQDISC,                       X
               (HCONN,                       X
               COMPCODE,                     X
               REASON),                      X
               VL,MF=(E,CALLLST)
*
         LA    R5,MQCC_OK
         C     R5,COMPCODE
         BNE   BADCALL
⋮
```

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

Vytvoření dynamické fronty

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

Tento extrakt se nevezme z ukázkových aplikací dodaných s produktem WebSphere MQ.

```
⋮
*
*    R5 = WORK REGISTER.
*
OPEN     DS   0H
*
         MVC  WOD_AREA,MQOD_AREA INITIALIZE WORKING VERSION OF
*                                  MQOD WITH DEFAULTS
         MVC  WOD_OBJECTNAME,MOD_Q    COPY IN THE MODEL Q NAME
         MVC  WOD_DYNAMICQNAME,DYN_Q  COPY IN THE DYNAMIC Q NAME
         L    R5,=AL4(MQOO_OUTPUT)    OPEN FOR OUTPUT AND
         A    R5,=AL4(MQOO_INQUIRE)   INQUIRE
         ST   R5,OPTIONS
```

```
*
* ISSUE MQI OPEN REQUEST USING REENTRANT
* FORM OF CALL MACRO
*
         CALL MQOPEN,                        X
               (HCONN,                       X
               WOD,                          X
               OPTIONS,                      X
               HOBJ,                         X
               COMPCODE,                     X
               REASON),VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK               CHECK THE COMPLETION CODE
         C   R5,COMPCODE              FROM THE REQUEST AND BRANCH
         BNE BADCALL                  TO ERROR ROUTINE IF NOT MQCC_OK
*
         MVC  TEMP_Q,WOD_OBJECTNAME  SAVE NAME OF TEMPORARY Q
*                             CREATED BY OPEN OF MODEL Q
*
⋮
BADCALL  DS   0H
```

```
⋮
*
*
*   CONSTANTS:
*
MOD_Q  DC   CL48'QUERY.REPLY.MODEL'  MODEL QUEUE NAME
DYN_Q  DC   CL48'QUERY.TEMPQ.*'      DYNAMIC QUEUE NAME
*
       CMQODA DSECT=NO,LIST=YES  CONSTANT VERSION OF MQOD
       CMQA                      MQI VALUE EQUATES
*
*   WORKING STORAGE
*
         DFHEISTG
HCONN    DS F                      CONNECTION HANDLE
OPTIONS  DS F                      OPEN OPTIONS
HOBJ     DS F                      OBJECT HANDLE
COMPCODE DS F                      MQI COMPLETION CODE
REASON   DS F                      MQI REASON CODE
TEMP_Q   DS CL(MQ_Q_NAME_LENGTH)   SAVED QNAME AFTER OPEN
*
WOD      CMQODA DSECT=NO,LIST=YES  WORKING VERSION OF MQOD
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  LIST FORM
                                                 OF CALL
*                                                 MACRO
⋮
         END
```

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 WebSphere MQ.

```
⋮
*
*    R5 = WORK REGISTER.
*
OPEN     DS   0H
*
         MVC  WOD_AREA,MQOD_AREA  INITIALIZE WORKING VERSION OF
*                              MQOD WITH DEFAULTS
         MVC  WOD_OBJECTNAME,Q_NAME    SPECIFY Q NAME TO OPEN
         LA   R5,MQOO_INPUT_EXCLUSIVE  OPEN FOR MQGET CALLS
*
         ST   R5,OPTIONS
*
* ISSUE MQI OPEN REQUEST USING REENTRANT FORM
* OF CALL MACRO
*
         CALL MQOPEN,                          X
               (HCONN,                         X
               WOD,                            X
               OPTIONS,                        X
               HOBJ,                           X
               COMPCODE,                       X
               REASON),VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK         CHECK THE COMPLETION CODE
         C   R5,COMPCODE        FROM THE REQUEST AND BRANCH
         BNE BADCALL            TO ERROR ROUTINE IF NOT MQCC_OK
*
⋮
BADCALL  DS   0H
⋮
*
*
*   CONSTANTS:
*
Q_NAME   DC   CL48'REQUEST.QUEUE'  NAME OF QUEUE TO OPEN
*
         CMQODA DSECT=NO,LIST=YES  CONSTANT VERSION OF MQOD
         CMQA                      MQI VALUE EQUATES
*
*   WORKING STORAGE
```

```
*
         DFHEISTG
HCONN    DS F           CONNECTION HANDLE
OPTIONS  DS F           OPEN OPTIONS
HOBJ     DS F           OBJECT HANDLE
COMPCODE DS F           MQI COMPLETION CODE
REASON   DS F           MQI REASON CODE
*
WOD  CMQODA DSECT=NO,LIST=YES  WORKING VERSION OF MQOD
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L  LIST FORM
                                                 OF CALL
*                                                 MACRO
.
         END
```

Zavření fronty

Tento příklad ukazuje, jak použít volání MQCLOSE k uzavření fronty.

Tento extrakt se nevezme z ukázkových aplikací dodaných s produktem WebSphere MQ.

```
.
*
* ISSUE MQI CLOSE REQUEST USING REENTRANT FROM OF
* CALL MACRO
*
*        HCONN WAS SET BY A PREVIOUS MQCONN REQUEST
*        HOBJ  WAS SET BY A PREVIOUS MQOPEN REQUEST
*        R5 = WORK REGISTER
*
CLOSE    DS    0H
         LA    R5,MQCO_NONE        NO SPECIAL CLOSE OPTIONS
         ST    R5,OPTIONS              ARE REQUIRED.
*
         CALL  MQCLOSE,                        X
               (HCONN,                         X
               HOBJ,                           X
               OPTIONS,                        X
               COMPCODE,                       X
               REASON),                        X
               VL,MF=(E,CALLLST)
*
         LA    R5,MQCC_OK
         C     R5,COMPCODE
         BNE   BADCALL
*
.
BADCALL  DS    0H
.
*                   CONSTANTS
*
         CMQA
*
*        WORKING STORAGE (REENTRANT)
*
WEG4     DSECT
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
HCONN    DS   F
HOBJ     DS   F
OPTIONS  DS   F
COMPCODE DS   F
REASON   DS   F
*
*
LEG4     EQU  *-WKEG4
         END
```

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 WebSphere MQ.

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

     LA   R5,BUFFER_LEN   RETRIEVE THE BUFFER LENGTH
     ST   R5,BUFFLEN      AND SAVE IT FOR MQM USE
*
     MVC  BUFFER,TEST_MSG    SET THE MESSAGE TO BE PUT
*
*    ISSUE MQI PUT REQUEST USING REENTRANT FORM
*    OF CALL MACRO
*
*        HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*        HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
         CALL MQPUT,                          X
              (HCONN,                         X
              HOBJ,                           X
              WMD,                            X
              WPMO,                           X
              BUFFLEN,                        X
              BUFFER,                         X
              COMPCODE,                       X
              REASON),VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK
         C   R5,COMPCODE
         BNE BADCALL
*
⋮
BADCALL DS  0H
⋮
```

```
*
*      CONSTANTS
*
   CMQMDA DSECT=NO,LIST=YES,PERSISTENCE=MQPER_PERSISTENT
   CMQPMOA DSECT=NO,LIST=YES
   CMQA
TEST_MSG DC CL80'THIS IS A TEST MESSAGE'
*
*      WORKING STORAGE DSECT
*
WORKSTG  DSECT
*
COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
```

```
*
WMD      CMQMDA DSECT=NO,LIST=NO
WPMO     CMQPMOA DSECT=NO,LIST=NO
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
⋮
         END
```

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 WebSphere MQ.

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

```
*
         MVC  WPMO_AREA,MQPMO_AREA       INITIALIZE WORKING MQPMO
*
         LA   R5,BUFFER_LEN           RETRIEVE THE BUFFER LENGTH
         ST   R5,BUFFLEN              AND SAVE IT FOR MQM USE
*
         MVC  BUFFER,TEST_MSG         SET THE MESSAGE TO BE PUT
*
* ISSUE MQI PUT REQUEST USING REENTRANT FORM OF CALL MACRO
*
*        HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*        HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
         CALL MQPUT1,                                X
                (HCONN,                              X
                 LMQOD,                              X
                 LMQMD,                              X
                 LMQPMO,                             X
                 BUFFERLENGTH,                       X
                 BUFFER,                             X
                 COMPCODE,                           X
                 REASON),VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK
         C   R5,COMPCODE
         BNE BADCALL
*
⋮
BADCALL DS  0H
⋮
*
```

```
*      CONSTANTS
*
```

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

získávání zprávy

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

Tento extrakt se nevezme z ukázkových aplikací dodaných s produktem WebSphere MQ.

```
⋮
*
*     CONNECT TO QUEUE MANAGER
*
CONN    DS  0H
⋮
*
*     OPEN A QUEUE FOR GET
*
OPEN    DS  0H
⋮
*
*     R4,R5,R6,R7 = WORK REGISTER.
*
GET  DS  0H
     LA   R4,MQMD               SET UP ADDRESSES AND
     LA   R5,MQMD_LENGTH        LENGTH FOR USE BY MVCL
     LA   R6,WMD                INSTRUCTION, AS MQMD IS
     LA   R7,WMD_LENGTH         OVER 256 BYES LONG.
     MVCL R6,R4                 INITIALIZE WORKING VERSION
*                                  OF MESSAGE DESCRIPTOR
*
     MVC  WGMO_AREA,MQGMO_AREA  INITIALIZE WORKING MQGMO
*
     LA   R5,BUFFER_LEN         RETRIEVE THE BUFFER LENGTH
     ST   R5,BUFFLEN            AND SAVE IT FOR MQM USE
*
*
* ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*        HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*        HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
         CALL  MQGET,                          X
               (HCONN,                         X
               HOBJ,                           X
               WMD,                            X
               WGMO,                           X
               BUFFLEN,                        X
```

```
               BUFFER,                          X
               DATALEN,                         X
               COMPCODE,                        X
               REASON),                         X
               VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK
         C   R5,COMPCODE
         BNE BADCALL
*
⋮
BADCALL DS  0H
⋮
```

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

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 WebSphere MQ.

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

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

```
                                        MINUTES BEFORE
                                        FAILING THE
                                        CALL
*
     LA   R5,BUFFER_LEN     RETRIEVE THE BUFFER LENGTH
     ST   R5,BUFFLEN        AND SAVE IT FOR MQM USE
*
*  ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*        HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*        HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
         CALL  MQGET,                           X
               (HCONN,                          X
               HOBJ,                            X
               WMD,                             X
               WGMO,                            X
               BUFFLEN,                         X
               BUFFER,                          X
               DATALEN,                         X
               COMPCODE,                        X
               REASON),                         X
               VL,MF=(E,CALLLST)
*
  LA  R5,MQCC_OK                 DID THE MQGET REQUEST
  C   R5,COMPCODE                  WORK OK?
  BE  GETOK                      YES, SO GO AND PROCESS.
  LA  R5,MQCC_WARNING            NO, SO CHECK FOR A WARNING.
  C   R5,COMPCODE                IS THIS A WARNING?
  BE  CHECK_W                    YES, SO CHECK THE REASON.
*
  LA  R5,MQRC_NO_MSG_AVAILABLE   IT MUST BE AN ERROR.
                                 IS IT DUE TO AN EMPTY
  C   R5,REASON                  QUEUE?
  BE  NOMSG                      YES, SO HANDLE THE ERROR
  B   BADCALL                    NO, SO GO TO ERROR ROUTINE
*
CHECK_W  DS  0H
         LA  R5,MQRC_TRUNCATED_MSG_ACCEPTED   IS THIS A
                                              TRUNCATED
         C   R5,REASON                        MESSAGE?
         BE  GETOK                  YES, SO GO AND PROCESS.
         B   BADCALL                NO, SOME OTHER WARNING
*
NOMSG    DS  0H
⋮
GETOK    DS  0H
⋮
```

```
BADCALL DS  0H
⋮
*
*      CONSTANTS
*
         CMQMDA DSECT=NO,LIST=YES
         CMQGMOA DSECT=NO,LIST=YES
         CMQA
*
TWO_MINUTES DC F'120000'       GET WAIT INTERVAL
*
*      WORKING STORAGE DSECT
```

```
*
WORKSTG  DSECT
*
COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
DATALEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD      CMQMDA DSECT=NO,LIST=NO
WGMO     CMQGMOA DSECT=NO,LIST=NO
```

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

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 WebSphere MQ.

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

```
*
     MVC  WGMO_AREA,MQGMO_AREA   INITIALIZE WORKING MQGMO
     LA   R5,MQGMO_SET_SIGNAL
     ST   R5,WGMO_OPTIONS
     MVC  WGMO_WAITINTERVAL,FIVE_MINUTES   WAIT UP TO FIVE
                                           MINUTES BEFORE
*                                          FAILING THE CALL
*
     XC   SIG_ECB,SIG_ECB   CLEAR THE ECB
     LA   R5,SIG_ECB        GET THE ADDRESS OF THE ECB
     ST   R5,WGMO_SIGNAL1   AND PUT IT IN THE WORKING
*                                 MQGMO
*

     LA   R5,BUFFER_LEN     RETRIEVE THE BUFFER LENGTH
     ST   R5,BUFFLEN        AND SAVE IT FOR MQM USE
*
*
*    ISSUE MQI GET REQUEST USING REENTRANT FORM OF CALL MACRO
*
*        HCONN WAS SET BY PREVIOUS MQCONN REQUEST
*        HOBJ WAS SET BY PREVIOUS MQOPEN REQUEST
*
         CALL  MQGET,                           X
               (HCONN,                          X
               HOBJ,                            X
               WMD,                             X
               WGMO,                            X
               BUFFLEN,                         X
               BUFFER,                          X
               DATALEN,                         X
               COMPCODE,                        X
               REASON),                         X
               VL,MF=(E,CALLLST)
*
         LA  R5,MQCC_OK        DID THE MQGET REQUEST
         C   R5,COMPCODE       WORK OK?
         BE  GETOK             YES, SO GO AND PROCESS.
         LA  R5,MQCC_WARNING   NO, SO CHECK FOR A WARNING.
         C   R5,COMPCODE       IS THIS A WARNING?
```

```
         BE  CHECK_W           YES, SO CHECK THE REASON.
         B   BADCALL           NO, SO GO TO ERROR ROUTINE
*
```

```
CHECK_W  DS  0H
         LA  R5,MQRC_SIGNAL_REQUEST_ACCEPTED
         C   R5,REASON   SIGNAL REQUEST SIGNAL SET?
         BNE BADCALL     NO, SOME ERROR OCCURRED
         B   DOWORK      YES, SO DO SOMETHING
*                          ELSE
*
CHECKSIG DS  0H
         CLC SIG_ECB+1(3),=AL3(MQEC_MSG_ARRIVED)
                                IS A MESSAGE AVAILABLE?
         BE  GET                YES, SO GO AND GET IT
*
         CLC SIG_ECB+1(3),=AL3(MQEC_WAIT_INTERVAL_EXPIRED)
                               HAVE WE WAITED LONG ENOUGH?
         BE  NOMSG             YES, SO SAY NO MSG AVAILABLE
         B   BADCALL           IF IT'S ANYTHING ELSE
*                              GO TO ERROR ROUTINE.
*
DOWORK   DS  0H
:
         TM  SIG_ECB,X'40'   HAS THE SIGNAL ECB BEEN POSTED?
         BO  CHECKSIG        YES, SO GO AND CHECK WHY
         B   DOWORK          NO, SO GO AND DO MORE WORK
*
NOMSG    DS  0H
:
GETOK    DS  0H
:
BADCALL DS  0H
:
*
*      CONSTANTS
*
           CMQMDA DSECT=NO,LIST=YES
           CMQGMOA DSECT=NO,LIST=YES
           CMQA
*
FIVE_MINUTES DC F'300000'        GET SIGNAL INTERVAL
*
*      WORKING STORAGE DSECT
*
WORKSTG  DSECT
*
COMPCODE DS F
REASON   DS F
BUFFLEN  DS F
DATALEN  DS F
OPTIONS  DS F
HCONN    DS F
HOBJ     DS F
SIG_ECB  DS F
```

```
*
BUFFER   DS CL80
BUFFER_LEN EQU *-BUFFER
*
WMD      CMQMDA DSECT=NO,LIST=NO
WGMO     CMQGMOA DSECT=NO,LIST=NO
*
CALLLST  CALL ,(0,0,0,0,0,0,0,0,0,0,0),VL,MF=L
*
:
         END
```

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 WebSphere MQ pro 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?
⋮
```

Konstanty

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

Soubory IBM WebSphere 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. Tyto soubory záhlaví jsou shrnuty v tabulce:

Tabulka 1. Soubory hlaviček C-prototypy volání, datové typy, návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**Systémy UNIX a 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. Tyto soubory jsou shrnuty v tabulce:

Tabulka 2. Soubory kopie jazyka COBOL-návratové kódy, konstanty a struktury					
Název souboru	**Popis**	**IBM i**	**Systémy UNIX**	**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
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

Tabulka 2. Soubory kopie jazyka COBOL-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**Systémy UNIX**	**Windows**	**z/OS**
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	MQCIH-záhlaví mostu CICS	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	MQIIH-záhlaví informací 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
CMQRFHx	MQRFH-Pravidla a záhlaví formátování	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**	**Systémy UNIX**	**Windows**	**z/OS**
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	MQSCO-volby konfigurace SSL		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					

Soubory začlenění PL/I

Pro programovací jazyk PL/I jsou k dispozici následující soubory 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**	**Systémy UNIX**	**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					

Soubory kopií RPG

Pro programovací jazyk RPG jsou k dispozici následující soubory COPY. 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**	**Systémy UNIX**	**Windows**	**z/OS**
Návratové kódy a konstanty					
CMQx	Definice MQI	G R			

Tabulka 4. Soubory kopie RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**Systémy UNIX**	**Windows**	**z/OS**
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	MQCIH-záhlaví mostu CICS	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			

Tabulka 4. Soubory kopie RPG-návratové kódy, konstanty a struktury (pokračování)					
Název souboru	**Popis**	**IBM i**	**Systémy UNIX**	**Windows**	**z/OS**
CMQEPHx	MQEPH-záhlaví vloženého PCF	G H			
CMQGMOx	MQGMO-Získat volby zpráv	G H R			
CMQIIHx	MQIIH-záhlaví informací 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

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 ve 32bitových verzích pouze a jsou sumarizovány v tabulce:

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

MQ_ * (Délky řetězce)

Tabulka 6. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQ_ABEND_CODE_LENGTH	4	X'00000004'
MQ_ACCOUNTING_TOKEN_LENGTH	32	X'00000020'
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'
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'

Tabulka 6. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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)	
DÉLKA_MAPOVÁNÍ_MFS_MFS_MAP	8	X'00000008'
DÉLKA_MODELU_MQ_REŽIMU	8	X'00000008'
DÉLKA_HLAVNÍHO_ZÁHLAVÍ MQ_MSG_	4000	X'00000FA0'

Tabulka 6. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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_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'
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'

Tabulka 6. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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 7. 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'

Tabulka 7. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_	64	X'00000040'
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	12	X'0000000C'
MQ_SECURITY_PROFILE_LENGTH	40	X'00000028'
MQ_SERVICE_COMPONENT_LENGTH	48	X'00000030'
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 8. Konstrukce konstant	
Název	**Struktura**
MQACH_STRUCTION_ID	"ACH¬"
POLE MQACH_STRUC_ID_ARRAY	'A','C','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 9. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQACH_VERSION_1	1	X'00000001'

Tabulka 9. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
AKTUÁLNÍ_VERZE MQACH_CURRENT_VERSION	1	X'00000001'
MQACH_LENGTH_1	(value differs by platform or version)	
AKTUÁLNÍ_DÉLKA MQACH_LENGTH	(value differs by platform or version)	

MQACT_ * (evidenční token)

Tabulka 10. 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 11. 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 12. 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 13. 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 14. 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 15. 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 16. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQIR_CONSTRUCT	"AIR¬"
POLE MQIR_STRUC_ID_ARRAY	'A','I','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 17. 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 18. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIT_VŠE	0	X'00000000'
MQAIT_CRL_LDAP	1	X'00000001'
MQACY_OCSP	2	X'00000002'

MQAS_ * (Asynchronní stavové hodnoty ve formátu příkazu)

Tabulka 19. 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 20. 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'
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'

Tabulka 20. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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)	
MQAT_USER_FIRST	65536	X'00010000'
MQAT_USER_LAST	999999999	X'3B9AC9FF'

MQAUTH_ * (Hodnoty oprávnění formátu příkazu)

Tabulka 21. 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'

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

MQAUTHOPT_ * (Volby oprávnění formátu příkazu)

Tabulka 22. 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 23. Konstrukce konstant	
Název	**Struktura**
MQAXC_STRUC_ID	"AXC¬"
MQAXC_STRUC_ID_POLE	'A','X','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 24. 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 25. Konstrukce konstant	
Název	**Struktura**
MQAXP_STRUCTION_ID	"AXP¬"
POLE MQAXP_STRUC_ID_ARRAY	'A','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 26. 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 27. 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 28. 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'
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 29. 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 30. Konstrukce konstant	
Název	**Struktura**
MQBMHO_STRUCTION_ID	"BMHO"
MQBMHO_STRUC_ID_POLE	'B','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 31. 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 32. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBMHO_NONE	0	X'00000000'
VLASTNOSTI MQBMHO_DELETE_PROPERTIES	1	X'00000001'

MQBND_ * (Výchozí vazby)

Tabulka 33. 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 34. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY OBJEKTU MQBO_STRUCT	"BO¬¬"
MQBO_STRUC_ID_POLE	'B','O','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 35. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBO_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQBO_CURRENT_VERSION	1	X'00000001'

Volby začátku

Tabulka 36. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQBO_NONE	0	X'00000000'

MQBT_ * (Typy mostů příkazového formátu)

Tabulka 37. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
FUNKCE MQBT_OTMA	1	X'00000001'

MQCA_ * (selektory znakových atributů)

Tabulka 38. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_ADMIN_TOPIC_NAME	2105	X'00000839'
MQCA_ALTERATION_DATE	2027	X'000007EB'
MQCA_ALTERATION_TIME	2028	X'000007EC'
MQCA_APPL_ID	2001	X'000007D1'
MQCA_AUTH_INFO_CONN_NAME	2053	X'00000805'
MQCA_AUTH_INFO_DESC	2046	X'000007FE'
MQCA_AUTH_INFO_NAME	2045	X'000007FD'
MQCA_AUTH_INFO_OCSP_URL	2109	X'0000083D'
MQCA_AUTO_REORG_CATALOG	2091	X'0000082B'
MQCA_AUTO_REORG_START_TIME	2090	X'0000082A'
MQCA_BACKOUT_REQ_Q_NAME	2019	X'000007E3'
MQCA_BASE_OBJECT_NAME	2002	X'000007D2'
MQCA_BASE_Q_NAME	2002	X'000007D2'
MQCA_BATCH_INTERFACE_ID	2068	X'00000814'
MQCA_CF_STRUC_DESC	2052	X'00000804'
MQCA_CF_STRUC_NAME	2039	X'000007F7'
MQCA_CHANNEL_AUTO_DEF_EXIT	2026	X'000007EA'
MQCA_CHILD	2101	X'00000835'
MQCA_CHINIT_SERVICE_PARM	2076	X'0000081C'
MQCA_CICS_FILE_NAME	2060	X'0000080C'
MQCA_CLUS_CHL_NAME	2124	X'0000084C'
MQCA_CLUSTER_DATE	2037	X'000007F5'
MQCA_CLUSTER_NAME	2029	X'000007ED'
MQCA_CLUSTER_NAMELIST	2030	X'000007EE'
MQCA_CLUSTER_Q_MGR_NAME	2031	X'000007EF'
MQCA_CLUSTER_TIME	2038	X'000007F6'
MQCA_CLUSTER_WORKLOAD_DATA	2034	X'000007F2'
MQCA_CLUSTER_WORKLOAD_EXIT	2033	X'000007F1'
MQCA_COMMAND_INPUT_Q_NAME	2003	X'000007D3'
MQCA_COMMAND_REPLY_Q_NAME	2067	X'00000813'
MQCA_CREATION_DATE	2004	X'000007D4'
MQCA_CREATION_TIME	2005	X'000007D5'
MQCA_DEAD_LETTER_Q_NAME	2006	X'000007D6'
MQCA_DEF_XMIT_Q_NAME	2025	X'000007E9'
MQCA_DNS_GROUP	2071	X'00000817'
MQCA_ENV_DATA	2007	X'000007D7'
MQCA_FIRST	2001	X'000007D1'
MQCA_IGQ_USER_ID	2041	X'000007F9'

Tabulka 38. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQCA_INITIATION_Q_NAME	2008	X'000007D8'
MQCA_LAST	4000	X'00000FA0'
MQCA_LAST_USED	2109	X'0000083D'
MQCA_LDAP_PASSWORD	2048	X'00000800'
MQCA_LDAP_USER_NAME	2047	X'000007FF'
MQCA_LU_GROUP_NAME	2072	X'00000818'
MQCA_LU_NAME	2073	X'00000819'
MQCA_LU62_ARM_SUFFIX	2074	X'0000081A'
MQCA_MODEL_DURABLE_Q	2096	X'00000830'
MQCA_MODEL_NON_DURABLE_Q	2097	X'00000831'
MQCA_MONITOR_Q_NAME	2066	X'00000812'
MQCA_NAMELIST_DESC	2009	X'000007D9'
MQCA_NAMELIST_NAME	2010	X'000007DA'
MQCA_NAMES	2020	X'000007E4'
MQCA_PARENT	2102	X'00000836'
MQCA_PASS_TICKET_APPL	2086	X'00000826'
MQCA_PROCESS_DESC	2011	X'000007DB'
MQCA_PROCESS_NAME	2012	X'000007DC'
MQCA_Q_DESC	2013	X'000007DD'
MQCA_Q_MGR_DESC	2014	X'000007DE'
MQCA_Q_MGR_IDENTIFIER	2032	X'000007F0'
MQCA_Q_MGR_NAME	2015	X'000007DF'
MQCA_Q_NAME	2016	X'000007E0'
MQCA_QSG_NAME	2040	X'000007F8'
MQCA_REMOTE_Q_MGR_NAME	2017	X'000007E1'
MQCA_REMOTE_Q_NAME	2018	X'000007E2'
MQCA_REPOSITORY_NAME	2035	X'000007F3'
MQCA_REPOSITORY_NAMELIST	2036	X'000007F4'
MQCA_RESUME_DATE	2098	X'00000832'
MQCA_RESUME_TIME	2099	X'00000833'
MQCA_SERVICE_DESC	2078	X'0000081E'
MQCA_SERVICE_NAME	2077	X'0000081D'
MQCA_SERVICE_START_ARGS	2080	X'00000820'
MQCA_SERVICE_START_COMMAND	2079	X'0000081F'
MQCA_SERVICE_STOP_ARGS	2082	X'00000822'
MQCA_SERVICE_STOP_COMMAND	2081	X'00000821'
MQCA_STDERR_DESTINATION	2084	X'00000824'
MQCA_STDOUT_DESTINATION	2083	X'00000823'
MQCA_SSL_CRL_NAMELIST	2050	X'00000802'

Tabulka 38. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCA_SSL_CRYPTO_HARDWARE	2051	X'00000803'
MQCA_SSL_KEY_LIBRARY	2069	X'00000815'
MQCA_SSL_KEY_MEMBER	2070	X'00000816'
MQCA_SSL_KEY_REPOSITORY	2049	X'00000801'
MQCA_STORAGE_CLASS	2022	X'000007E6'
MQCA_STORAGE_CLASS_DESC	2042	X'000007FA'
MQCA_SYSTEM_LOG_Q_NAME	2065	X'00000811'
MQCA_TCP_NAME	2075	X'0000081B'
MQCA_TOPIC_DESC	2093	X'0000082D'
MQCA_TOPIC_NAME	2092	X'0000082C'
MQCA_TOPIC_STRING_FILTER	2108	X'0000083C'
MQCA_TOPIC_STRING	2094	X'0000082E'
MQCA_TPIPE_NAME	2085	X'00000825'
MQCA_TRIGGER_CHANNEL_NAME	2064	X'00000810'
MQCA_TRIGGER_DATA	2023	X'000007E7'
MQCA_TRIGGER_PROGRAM_NAME	2062	X'0000080E'
MQCA_TRIGGER_TERM_ID	2063	X'0000080F'
MQCA_TRIGGER_TRANS_ID	2061	X'0000080D'
MQCA_USER_DATA	2021	X'000007E5'
MQCA_USER_LIST	4000	X'00000FA0'
MQCA_VERSION	2120	X'00000848'
MQCA_XCF_GROUP_NAME	2043	X'000007FB'
MQCA_XCF_MEMBER_NAME	2044	X'000007FC'
MQCA_XMIT_Q_NAME	2024	X'000007E8'

MQCACF_ * (Typy znakových znakových parametrů příkazu)

Tabulka 39. 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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 39. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'
MQCAF_LIST_USED	3172	X'00000C64'

MQCACH_ * (Typy parametrů kanálu znaků v příkazovém kanálu)

Tabulka 40. 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'
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'

Tabulka 40. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (informační hlavička CICS ADS-Deskriptory)

Tabulka 41. 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 42. 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 43. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCAMO_FIRST	2701	X'00000A8D'
MQCAMO_CLOSE_DATE	2701	X'00000A8D'
MQCAMO_CLOSE_TIME	2702	X'00000A8E'
MQCAMO_CONN_DATE	2703	X'00000A8F'
MQCAMO_CONN_TIME	2704	X'00000A90'
MQCAMO_DISC_DATE	2705	X'00000A91'
MQCAMO_DISC_TIME	2706	X'00000A92'
MQCAMO_END_DATE	2707	X'00000A93'
MQCAMO_END_TIME	2708	X'00000A94'
MQCAMO_OPEN_DATE	2709	X'00000A95'
MQCAMO_OPEN_TIME	2710	X'00000A96'

Tabulka 43. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 44. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQCBC_	"CBC¬"
POLE MQCBC_STRUC_ID_ARRAY	'C','B','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 45. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCBC_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQCBC_CURRENT_VERSION	1	X'00000001'

MQCBCF_ * (příznaky konstant MQCBC)

Tabulka 46. 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 47. 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 48. Konstrukce konstant	
Název	**Struktura**
MQCBD_STRUCTION_ID	"CBD¬"
POLE MQCBD_STRUC_ID_ARRAY	'C','B','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 49. 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 50. 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 51. 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 52. 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 53. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCC_OK	0	X'00000000'
VAROVÁNÍ MQCC_WARNING	1	X'00000001'

Tabulka 53. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SELHÁNÍ MQCC_FAILED	2	X'00000002'
NEZNÁMÉ MQCC_UNKNOWN	-1	X'FFFFFFFF'

MQCCSI_ * (identifikátory kódované znakové sady)

Tabulka 54. 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_ * (Volby konverzace záhlaví informací CICS)

Tabulka 55. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCKT_YES	1	X'00000001'
MQCCT_NO	0	X'00000000'

MQCD_ * (Struktura definice kanálu)

Tabulka 56. 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'
AKTUÁLNÍ_VERZE MQCD_	9	X'00000009'
MQCD_LENGTH_4	(value differs by platform or version)	
MQCD_LENGTH_5	(value differs by platform or version)	
MQCD_LENGTH_6	(value differs by platform or version)	
MQCD_LENGTH_7	(value differs by platform or version)	
MQCD_LENGTH_8	(value differs by platform or version)	

Tabulka 56. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCD_LENGTH_9	(value differs by platform or version)	
AKTUÁLNÍ_DÉLKA_MQCD_	(value differs by platform or version)	

MQCDC_ * (Převod dat kanálu)

Tabulka 57. 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 58. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCF_NONE	0	X'00000000'
MQCF_DIST_LISTS	1	X'00000001'

MQCFAC_ * (prostředek informačního záhlaví CICS)

Tabulka 59. 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 60. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFBF_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFBS_ * (struktura parametrů bajtového řetězce formátu příkazu)

Tabulka 61. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFBS_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFC_ * (Volby ovládacího prvku záhlaví příkazu format)

Tabulka 62. Hodnoty konstant

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

MQCFGR_ * (struktura parametrů skupiny příkazů)

Tabulka 63. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFGR_STRUST_LENGTH	16	X'00000010'

MQCFH_ * (struktura záhlaví příkazového formátu)

Tabulka 64. 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 65. Hodnoty konstant

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

MQCFIL_ * (Struktura konfiguračního parametru celého čísla formátu příkazu)

Tabulka 66. 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 67. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIL64_STRUC_LENGTH_FIXED	16	X'00000010'

MQCFIN_ * (struktura parametrů celého čísla příkazu)

Tabulka 68. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFIN_STRUST_LENGTH	16	X'00000010'

MQCFIN64_* (Struktura parametrů 64bitového celočíselného formátu příkazů)

Tabulka 69. 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 70. 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 71. 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 72. 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 73. 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 74. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFSF_STRUC_LENGTH_FIXED	24	X'00000018'

MQCFSL_ * (Struktura parametrů řetězce formátu příkazu)

Tabulka 75. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
PEVNÁ DÉLKA_STRUKTURA_MQCFSL_STRUCT	24	X'00000018'

MQCFST_ * (struktura parametrů řetězce formátu příkazu)

Tabulka 76. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQCFST_STRUC_LENGTH_FIXED	20	X'00000014'

MQCFSTATUS_ * (Stav prostředku CF příkazu format)

Tabulka 77. 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 78. 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'

Tabulka 78. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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 79. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCFTYPE_APPL	0	X'00000000'
MQCFTYPE_ADMIN	1	X'00000001'

MQCFUNC_ * (funkce záhlaví informací CICS)

Tabulka 80. 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'

Tabulka 80. Konstrukce konstant (pokračování)	
Název	**Struktura**
MQCFUNC_MQPUT_ARRAY	'P','U','T','¬'
MQCFUNC_MQPUT1_ARRAY	'P','U','T','1'
MQCFUNC_NON_ARRAY	'¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQCGWI_ * (informační záhlaví CICS Get Wait Interval)

Tabulka 81. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCGWI_DEFAULT	-2	X'FFFFFFFE'

MQCHAD_ * (Automatická definice kanálu)

Tabulka 82. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHAD_DISABLED	0	X'00000000'
MQCHAD_ENABLED	1	X'00000001'

MQCHIDS_ * (Nejistý formát příkazu)

Tabulka 83. 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 84. 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 85. 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'

Tabulka 85. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 86. 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 87. 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 88. 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'

Tabulka 88. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHSSTATE_IN_MQI_CALL	1700	X'000006A4'
MQCHSSTATE_COMPRESSING	1800	X'00000708'

MQCHT_ * (typy kanálů)

Tabulka 89. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHT_SENDER	1	X'00000001'
SERVER MQCHT_SERVER	2	X'00000002'
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 90. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCHTAB_Q_MGR	1	X'00000001'
MQCHTAB_CLNTCONN	2	X'00000002'

MQCI_ * (Identifikátor korelace)

Tabulka 91. 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 systému CICS)

Struktura záhlaví informací CICS

Tabulka 92. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQCIH_	"CIH¬"
POLE MQCIH_STRUC_ID_ARRAY	'C','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 93. 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'

Příznaky záhlaví informací CICS

Tabulka 94. 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 95. 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 96. 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 97. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLRT_RETAINED	1	X'00000001'

MQCLT_ * (typy odkazů záhlaví informačního kanálu CICS)

Tabulka 98. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLT_PROGRAM	1	X'00000001'
TRANSAKCE MQCLT_TRANSACTION	2	X'00000002'

MQCLWL_ * (Vytížení klastru)

Tabulka 99. 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 určuje, která přenosová fronta je standardně vybrána kanály odesílatele klastru k získání zpráv z kanálů příjemce klastru k odeslání zpráv do kanálů příjemce klastru.

Tabulka 100. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCLXQ_SCTQ	0	X'00000000'
MQCLXQ_CHANNEL	1	X'00000001'

Související odkazy

“DefClusterXmitQueueType (MQLONG)” na stránce 769
Atribut DefClusterXmitQueueType určuje, která přenosová fronta je standardně vybrána kanály odesílatele klastru k získání zpráv z kanálů příjemce klastru k odeslání zpráv do kanálů příjemce klastru.

Změnit správce front

Zjistit správce front

Dotaz na správce front (odezva)

“MQINQ-Dotaz na atributy objektu” na stránce 662
Volání MQINQ vrátí pole celých čísel a sadu znakových řetězců, které obsahují atributy objektu.

MQCMD_ * (Příkazové kódy)

Tabulka 101. 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'

Tabulka 101. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 101. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 101. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 101. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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 102. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCMDI_CMDSCOPY_ACCEPTED	1	X'00000001'

Tabulka 102. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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 103. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCMDL_LEVEL_1	100
MQCMDL_LEVEL_101	101
MQCMDL_LEVEL_110	110
MQCMDL_LEVEL_114	114
MQCMDL_LEVEL_120	120
MQCMDL_LEVEL_200	200
MQCMDL_LEVEL_201	201
MQCMDL_LEVEL_210	210
MQCMDL_LEVEL_211	211
MQCMDL_LEVEL_220	220
MQCMDL_LEVEL_221	221
MQCMDL_LEVEL_230	230
MQCMDL_LEVEL_320	320
MQCMDL_LEVEL_420	420
MQCMDL_LEVEL_500	500
MQCMDL_LEVEL_510	510
MQCMDL_LEVEL_520	520
MQCMDL_LEVEL_530	530

Tabulka 103. Konstantní názvy a hodnoty (pokračování)	
Název	**Hodnota**
MQCMDL_LEVEL_531	531
MQCMDL_LEVEL_600	600
MQCMDL_LEVEL_700	700
MQCMDL_LEVEL_701	701
MQCMDL_LEVEL_710	710
MQCMDL_LEVEL_711	711
MQCMDL_LEVEL_750	750

MQCMHO_ * (Vytvoření voleb a struktury manipulátoru zprávy)

Vytvořit strukturu voleb zpracování zpráv

Tabulka 104. Konstrukce konstant	
Název	**Struktura**
MQCMHO_STRUCTION_ID	"CMHO"
MQCMHO_STRUC_ID_POLE	'C','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 105. 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 106. 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 107. Konstrukce konstant	
Název	**Struktura**
MQCNO_STRUCTION_ID	"CNO¬"
MQCNO_STRUC_ID_POLE	'C','N','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 108. 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 109. 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_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 110. Hodnoty konstant

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

MQCODL_ * (délka výstupních dat záhlaví CICS)

Tabulka 111. Hodnoty konstant

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

MQCOMPRESS_* (Komprese kanálu)

Tabulka 112. 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 113. 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 114. 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'

Tabulka 114. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCOPY_REPORT	16	X'00000010'
MQCOPY_DEFAULT	22	X'00000016'

MQCQT_ * (Typy front klastru)

Tabulka 115. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQCQT_LOCAL_Q	1	X'00000001'
MQCQT_ALIAS_Q	2	X'00000002'
MQCQT_REMOTE_Q	3	X'00000003'
ALIAS MQCQT_Q_MGR_ALIAS	4	X'00000004'

MQCRC_ * (Návratové kódy záhlaví CICS)

Tabulka 116. 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 117. 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 systému CICS)

Tabulka 118. Konstrukce konstant	
Název	**Struktura**
MQCSC_START	"S¬¬¬"
POČÁTEČNÍ_DATA MQCSC_STARTDATA	"SD¬¬"
MQCSC_TERMINPUT	"TD¬¬"

Tabulka 118. Konstrukce konstant (pokračování)	
Název	**Struktura**
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 119. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQCSP_STRUCT	"CSP¬"
MQCSP_STRUKRE_ID_POLE	'C','S','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 120. 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 121. 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 122. 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 123. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQCT_NONE	X'00...00' (128 nulových hodnot)
MQCT_NON_ARRAY	'\0','\0',... (128 nulových hodnot)

MQCTES_ * (Koncový stav úlohy záhlaví informací CICS)

Tabulka 124. 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 125. Konstrukce konstant	
Název	**Struktura**
MQCTLO_STRUCTION_ID	"CTLO"
POLE MQCTLO_STRUC_ID_ARRAY	'C','T','L','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 126. 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 127. 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_ * (řídicí prvky bloku informací CICS-of-Work Controls)

Tabulka 128. 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 129. 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 130. 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'
AKTUÁLNÍ_VERZE MQCXP_CURRENT_VERSION	8	X'00000008'

MQDC_ * (Cílová třída)

Tabulka 131. 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 132. 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)	
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)	
MQDCC_TARGET_ENC_NORMAL	256	X'00000100'
MQDCC_TARGET_ENC_REVERSED	512	X'00000200'
MQDCC_TARGET_ENC_UNDEFINED	0	X'00000000'

Tabulka 132. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDCC_NONE	0	X'00000000'

Volby masky a faktory konverze

Tabulka 133. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MAQDCC_ZDROJOVÁ_MASKA	240	X'000000F0'
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 134. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDELO_NONE	0	X'00000000'
MQDELO_LOCAL	4	X'00000004'

MQDH_ * (struktura záhlaví distribuce)

Tabulka 135. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQDH_	"DH¬¬"
POLE MQDH_STRUC_ID_ARRAY	'D','H','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 136. 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 137. 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 138. 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 139. 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 140. 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 141. 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 142. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQDLV_AS_PARENT	0	X'00000000'
MQDLV_ALL	1	X'00000001'
MQDLV_ALL_DUR	2	X'00000002'
MQDLV_ALL_AVAIL	3	X'00000003'

MQDMHO_ * (Odstranění voleb a struktury zpracování zpráv)

Odstranit strukturu voleb obsluhy zprávy

Tabulka 143. Konstrukce konstant	
Název	**Struktura**
MQDMHO_STRUCTION_ID	"DMHO"
MQDMHO_STRUC_ID_POLE	'D','M','H','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

Volby odstranění popisovače zpráv

Tabulka 145. 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 146. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQDMPO_	"DMPO"
POLE MQDMPO_STRUC_ID_ARRAY	'D','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

Volby odstranění vlastností zprávy

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

MQDNSWLM_* (DNS WLM)

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

MQDT_ * (cílové typy)

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

MQDXP_ * (struktura výstupního parametru převodu)

Tabulka 151. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQDXP_STRUCTION_ID	"DXP¬"

Tabulka 151. Konstrukce konstant (pokračování)	
Název	**Struktura**
POLE MQDXP_STRUC_ID_ARRAY	'D','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

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

MQEC_ * (signálové hodnoty)

Tabulka 153. 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 154. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEI_UNLIMITED	-1	X'FFFFFFFF'

MQENC_ * (Kódování)

MQENC_ * (Kódování)

Tabulka 155. Hodnoty konstant podle platformy			
Název	**Platforma**	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_NATIVE	IBM i	273	X'00000111 '.
	Linux	546	X'00000222 '.
	Linux na platformě SPARC	273	X'00000111 '.
	Linux na platformě x86	546	X'00000222 '.
	Solaris na SPARC	273	X'00000111 '.
	Systémy UNIX	273	X'00000111 '.
	Windows	546	X'00000222 '.
	Micro fokus COBOL v systému Windows	17	X'00000011 '
	z/OS	785	X'00000311 '

MQENC_ * (zakódování Masky)

Tabulka 156. 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 157. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_INTEGER_UNDEFINED	0	X'00000000'
MQENC_INTEGER_NORMAL	1	X'00000001'
MQENC_INTEGER_REVERSED	2	X'00000002'

MQENC_ * (kódování pro desítková čísla komprimovaného desetinného čísla)

Tabulka 158. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQENC_DECIMAL_UNDEFINED	0	X'00000000'
MQENC_DECIMAL_NORMAL	16	X'00000010'
MQENC_DECIMAL_REVERSED	32	X'00000020'

MQENC_ * (kódování čísel s pohyblivou řádovou čárkou)

Tabulka 159. 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 160. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQEPH_	"EPH¬"
POLE MQEPH_STRUC_ID_ARRAY	'E','P','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 161. 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 162. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEPH_NONE	0	X'00000000'
VLOŽKA MQEPH_CCSID_EMBEDDED	1	X'00000001'

MQET_ * (Typy znaků změny formátu příkazu)

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

MQEVO_ * (Standardní formát událostí událostí)

Tabulka 164. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEVO_OTHER	0	X'00000000'
MQEVO_CONSOLE	1	X'00000001'
MQEVO_INIT	2	X'00000002'
MQEVO_MSG	3	X'00000003'
MQEVO_MQSET	4	X'00000004'
MQEVO_INTERNAL	5	X'00000005'

MQEVR_ * (záznam událostí ve formátu příkazu)

Tabulka 165. 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 166. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQEXPI_OFF	0	X'00000000'

MQFB_ * (hodnoty zpětné vazby)

Tabulka 167. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQFB_NONE	0	X'00000000'
MQFB_SYSTEM_FIRST	1	X'00000001'
MQFB_QUIT	256	X'00000100'
MQFB_EXPIRATION	258	X'00000102'
MQFB_COA	259	X'00000103'
MQFB_COD	260	X'00000104'
MQFB_CHANNEL_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'

Tabulka 167. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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 168. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQFC_YES	1	X'00000001'
MQFC_NO	0	X'00000000'

MQFMT_ * (Formáty)

Tabulka 169. 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¬"

Tabulka 169. Konstantní názvy a hodnoty (pokračování)	
Název	**Hodnota**
UDÁLOST MQFMT_EVENT	"MQEVENT¬"
MQFMT_IMS	"MQIMS¬¬¬"
MQFMT_IMS_VAR_STRING	"MQIMSVS¬"
ROZŠÍŘENÍ MQFMT_MD_EXTENSION	"MQHMDE¬¬"
MQFMT_PCF	"MQPCF¬¬¬"
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.

MQGA_ * (Selektory atributu skupiny)

Tabulka 170. 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 171. 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 172. 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 173. Konstrukce konstant

Název	Struktura
MQGMO_STRUCTION_ID	"GMO¬"
MQGMO_STRUC_ID_POLE	'G','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 174. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQGMO_VERSION_1	1	X'00000001'

Tabulka 174. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 175. 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'
MQGMO_PROPERTIES_IN_HANDLE	134217728	X'08000000'
MQGMO_PROPERTIES_COMPATIBILITY	268435456	X'10000000'

Tabulka 175. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQGMO_PROPERTIES_AS_Q_DEF	0	X'00000000'
MQGMO_NONE	0	X'00000000'

MQGS_ * (Stav skupiny)

Tabulka 176. 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 177. 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 178. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQHT_UNUSABLE_HBAG	-1	X'FFFFFFFF'
MQHB_NONE	-2	X'FFFFFFFE'

MQHC_ * (Manipulátory připojení)

Tabulka 179. 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 180. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MAHL_UNUSABLE_HMSG	-1	X'FFFFFFFF'
MQM_NONE	0	X'00000000'

MQHO_ * (Popisovač objektu)

Tabulka 181. Hodnoty konstant

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

MQHSTATE_ * (obslužné rutiny formátu příkazu)

Tabulka 182. Hodnoty konstant

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

MQIA_ * (Selektory celočíselných atributů)

Tabulka 183. 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_TYPE	103	X'00000067'
MQIA_ADOPTNEWMCA_INTERVAL	104	X'00000068'
MQIA_APPL_TYPE	1	X'00000001'
MQIA_ARCHIVE	60	X'0000003C'
MQIA_AUTH_INFO_TYPE	66	X'00000042'
MQIA_AUTHORITY_EVENT	47	X'0000002F'
MQIA_AUTO_REORG_INTERVAL	174	X'000000AE'
MQIA_AUTO_REORGANIZATION	173	X'000000AD'
MQIA_BACKOUT_THRESHOLD	22	X'00000016'
MQIA_BASE_TYPE	193	X'000000C1'
MQIA_BATCH_INTERFACE_AUTO	86	X'00000056'
MQIA_BRIDGE_EVENT	74	X'0000004A'
MQIA_CF_LEVEL	70	X'00000046'
MQIA_CF_RECOVER	71	X'00000047'
MQIA_CHANNEL_AUTO_DEF	55	X'00000037'
MQIA_CHANNEL_AUTO_DEF_EVENT	56	X'00000038'
MQIA_CHANNEL_EVENT	73	X'00000049'

Tabulka 183. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_CHINIT_ADAPTERS	101	X'00000065'
MQIA_CHINIT_CONTROL	119	X'00000077'
MQIA_CHINIT_DISPATCHERS	105	X'00000069'
MQIA_CHINIT_TRACE_AUTO_START	117	X'00000075'
MQIA_CHINIT_TRACE_TABLE_SIZE	118	X'00000076'
MQIA_CLUSTER_Q_TYPE	59	X'0000003B'
MQIA_CLUSTER_WORKLOAD_LENGTH	58	X'0000003A'
MQIA_CLWL_MRU_CHANNELS	97	X'00000061'
MQIA_CLWL_Q_RANK	95	X'0000005F'
MQIA_CLWL_Q_PRIORITY	96	X'00000060'
MQIA_CLWL_USEQ	98	X'00000062'
MQIA_CMD_SERVER_AUTO	87	X'00000057'
MQIA_CMD_SERVER_CONTROL	120	X'00000078'
MQIA_CMD_SERVER_CONVERT_MSG	88	X'00000058'
MQIA_CMD_SERVER_DLQ_MSG	89	X'00000059'
MQIA_CODED_CHAR_SET_ID	2	X'00000002'
MQIA_COMM_EVENT	232	X'000000E8'
MQIA_COMMAND_EVENT	99	X'00000063'
MQIA_COMMAND_LEVEL	31	X'0000001F'
MQIA_CONFIGURATION_EVENT	51	X'00000033'
MQIA_CPI_LEVEL	27	X'0000001B'
MQIA_CURRENT_Q_DEPTH	3	X'00000003'
MQIA_DEF_BIND	61	X'0000003D'
MQIA_DEF_CLUSTER_XMIT_Q_TYPE	250	X'000000FA'
MQIA_DEF_INPUT_OPEN_OPTION	4	X'00000004'
MQIA_DEF_PERSISTENCE	5	X'00000005'
MQIA_DEF_PRIORITY	6	X'00000006'
MQIA_DEF_PUT_RESPONSE_TYPE	184	X'000000B8'
MQIA_DEF_READ_AHEAD	188	X'000000BC'
MQIA_DEFINITION_TYPE	7	X'00000007'
MQIA_DIST_LISTS	34	X'00000022'
MQIA_DNS_WLM	106	X'0000006A'
MQIA_DURABLE_SUB	175	X'000000AF'
MQIA_EXPIRY_INTERVAL	39	X'00000027'
MQIA_FIRST	1	X'00000001'
MQIA_HARDEN_GET_BACKOUT	8	X'00000008'
MQIA_HIGH_Q_DEPTH	36	X'00000024'
MQIA_IGQ_PUT_AUTHORITY	65	X'00000041'
MQIA_INDEX_TYPE	57	X'00000039'

Tabulka 183. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_INHIBIT_EVENT	48	X'00000030'
MQIA_INHIBIT_GET	9	X'00000009'
MQIA_INHIBIT_PUB	181	X'000000B5'
MQIA_INHIBIT_PUT	10	X'0000000A'
MQIA_INHIBIT_SUB	182	X'000000B6'
MQIA_INTRA_GROUP_QUEUING	64	X'00000040'
MQIA_IP_ADDRESS_VERSION	93	X'0000005D'
MQIA_LAST	2000	X'000007D0'
MQIA_LAST_USED	233	X'000000E9'
MQIA_LISTENER_PORT_NUMBER	85	X'00000055'
MQIA_LISTENER_TIMER	107	X'0000006B'
MQIA_LOGGER_EVENT	94	X'0000005E'
MQIA_LU62_CHANNELS	108	X'0000006C'
MQIA_LOCAL_EVENT	49	X'00000031'
MQIA_MSG_MARK_BROWSE_INTERVAL	68	X'00000044'
MQIA_MAX_CHANNELS	109	X'0000006D'
MQIA_MAX_CLIENTS	172	X'000000AC'
MQIA_MAX_GLOBAL_LOCKS	83	X'00000053'
MQIA_MAX_HANDLES	11	X'0000000B'
MQIA_MAX_LOCAL_LOCKS	84	X'00000054'
MQIA_MAX_MSG_LENGTH	13	X'0000000D'
MQIA_MAX_OPEN_Q	80	X'00000050'
MQIA_MAX_PRIORITY	14	X'0000000E'
MQIA_MAX_PROPERTIES_LENGTH	192	X'000000C0'
MQIA_MAX_Q_DEPTH	15	X'0000000F'
MQIA_MAX_Q_TRIGGERS	90	X'0000005A'
MQIA_MAX_RECOVERY_TASKS	171	X'000000AB'
MQIA_MAX_UNCOMMITTED_MSGS	33	X'00000021'
MQIA_MCAST_BRIDGE	233	X'000000E9'
MQIA_MONITOR_INTERVAL	81	X'00000051'
MQIA_MONITORING_AUTO_CLUSSDR	124	X'0000007C'
MQIA_MONITORING_CHANNEL	122	X'0000007A'
MQIA_MONITORING_Q	123	X'0000007B'
MQIA_MSG_DELIVERY_SEQUENCE	16	X'00000010'
MQIA_MSG_DEQ_COUNT	38	X'00000026'
MQIA_MSG_ENQ_COUNT	37	X'00000025'
MQIA_NAME_COUNT	19	X'00000013'
MQIA_NAMELIST_TYPE	72	X'00000048'
MQIA_NPM_CLASS	78	X'0000004E'

Tabulka 183. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_NPM_DELIVERY	196	X'000000C4'
MQIA_OPEN_INPUT_COUNT	17	X'00000011'
MQIA_OPEN_OUTPUT_COUNT	18	X'00000012'
MQIA_OUTBOUND_PORT_MAX	140	X'0000008C'
MQIA_OUTBOUND_PORT_MIN	110	X'0000006E'
MQIA_PAGESET_ID	62	X'0000003E'
MQIA_PERFORMANCE_EVENT	53	X'00000035'
MQIA_PLATFORM	32	X'00000020'
MQIA_PM_DELIVERY	195	X'000000C3'
MQIA_PROPERTY_CONTROL	190	X'000000BE'
MQIA_PROT_POLICY_CAPABILITY	251	X'000000FB'
MQIA_PROXY_SUB	199	X'000000C7'
MQIA_PUB_COUNT	215	X'000000D7'
MQIA_PUB_SCOPE	219	X'000000DB'
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	206	X'000000CE'
MQIA_PUBSUB_MODE	187	X'000000BB'
MQIA_PUBSUB_NP_MSG	203	X'000000CB'
MQIA_PUBSUB_NP_RESP	205	X'000000CD'
MQIA_PUBSUB_SYNC_PT	207	X'000000CF'
MQIA_Q_DEPTH_HIGH_EVENT	43	X'0000002B'
MQIA_Q_DEPTH_HIGH_LIMIT	40	X'00000028'
MQIA_Q_DEPTH_LOW_EVENT	44	X'0000002C'
MQIA_Q_DEPTH_LOW_LIMIT	41	X'00000029'
MQIA_Q_DEPTH_MAX_EVENT	42	X'0000002A'
MQIA_Q_SERVICE_INTERVAL	54	X'00000036'
MQIA_Q_SERVICE_INTERVAL_EVENT	46	X'0000002E'
MQIA_Q_TYPE	20	X'00000014'
MQIA_Q_USERS	82	X'00000052'
MQIA_QMOPT_CONS_COMMS_MSGS	155	X'0000009B'
MQIA_QMOPT_CONS_CRITICAL_MSGS	154	X'0000009A'
MQIA_QMOPT_CONS_ERROR_MSGS	153	X'00000099'
MQIA_QMOPT_CONS_INFO_MSGS	151	X'00000097'
MQIA_QMOPT_CONS_REORG_MSGS	156	X'0000009C'
MQIA_QMOPT_CONS_SYSTEM_MSGS	157	X'0000009D'
MQIA_QMOPT_CONS_WARNING_MSGS	152	X'00000098'
MQIA_QMOPT_CSMT_ON_ERROR	150	X'00000096'
MQIA_QMOPT_INTERNAL_DUMP	170	X'000000AA'
MQIA_QMOPT_LOG_COMMS_MSGS	162	X'000000A2'
MQIA_QMOPT_LOG_CRITICAL_MSGS	161	X'000000A1'

Tabulka 183. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIA_QMOPT_LOG_ERROR_MSGS	160	X'000000A0'
MQIA_QMOPT_LOG_INFO_MSGS	158	X'0000009E'
MQIA_QMOPT_LOG_REORG_MSGS	163	X'000000A3'
MQIA_QMOPT_LOG_SYSTEM_MSGS	164	X'000000A4'
MQIA_QMOPT_LOG_WARNING_MSGS	159	X'0000009F'
MQIA_QMOPT_TRACE_COMMS	166	X'000000A6'
MQIA_QMOPT_TRACE_CONVERSION	168	X'000000A8'
MQIA_QMOPT_TRACE_REORG	167	X'000000A7'
MQIA_QMOPT_TRACE_MQI_CALLS	165	X'000000A5'
MQIA_QMOPT_TRACE_SYSTEM	169	X'000000A9'
MQIA_QSG_DISP	63	X'0000003F'
MQIA_READ_AHEAD	189	X'000000BD'
MQIA_RECEIVE_TIMEOUT	111	X'0000006F'
MQIA_RECEIVE_TIMEOUT_MIN	113	X'00000071'
MQIA_RECEIVE_TIMEOUT_TYPE	112	X'00000070'
MQIA_REMOTE_EVENT	50	X'00000032'
MQIA_RETENTION_INTERVAL	21	X'00000015'
MQIA_SCOPE	45	X'0000002D'
MQIA_SECURITY_CASE	141	X'0000008D'
MQIA_SERVICE_CONTROL	139	X'0000008B'
MQIA_SERVICE_TYPE	121	X'00000079'
MQIA_SHAREABILITY	23	X'00000017'
MQIA_SHARED_Q_Q_MGR_NAME	77	X'0000004D'
MQIA_SSL_EVENT	75	X'0000004B'
MQIA_SSL_FIPS_REQUIRED	92	X'0000005C'
MQIA_SSL_RESET_COUNT	76	X'0000004C'
MQIA_SSL_TASKS	69	X'00000045'
MQIA_START_STOP_EVENT	52	X'00000034'
MQIA_STATISTICS_CHANNEL	129	X'00000081'
MQIA_STATISTICS_AUTO_CLUSSDR	130	X'00000082'
MQIA_STATISTICS_INTERVAL	131	X'00000083'
MQIA_STATISTICS_MQI	127	X'0000007F'
MQIA_STATISTICS_Q	128	X'00000080'
MQIA_SUB_COUNT	204	X'000000CC'
MQIA_SUB_SCOPE	218	X'000000DA'
MQIA_SYNCPOINT	30	X'0000001E'
MQIA_TCP_CHANNELS	114	X'00000072'
MQIA_TCP_KEEP_ALIVE	115	X'00000073'
MQIA_TCP_STACK_TYPE	116	X'00000074'

Tabulka 183. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIA_TIME_SINCE_RESET	35	X'00000023'
MQIA_TOPIC_DEF_PERSISTENCE	185	X'000000B9'
MQIA_TOPIC_TYPE	208	X'000000D0'
MQIA_TRACE_ROUTE_RECORDING	137	X'00000089'
MQIA_TREE_LIFE_TIME	183	X'000000B7'
MQIA_TRIGGER_CONTROL	24	X'00000018'
MQIA_TRIGGER_DEPTH	29	X'0000001D'
MQIA_TRIGGER_INTERVAL	25	X'00000019'
MQIA_TRIGGER_MSG_PRIORITY	26	X'0000001A'
MQIA_TRIGGER_TYPE	28	X'0000001C'
MQIA_TRIGGER_RESTART	91	X'0000005B'
MQIA_USAGE	12	X'0000000C'
MQIA_USE_DEAD_LETTER_Q	234	X'000000EA'
MQIA_USER_LIST	2000	X'000007D0'
MQIA_WILDCARD_OPERATION	216	X'000000D8'
MQIA_XR_CAPABILITY	243	X'000000F3'

MQIACF_ * (typy parametrů s celočíselnými parametry)

Tabulka 184. 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'

Tabulka 184. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 184. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
MQIAKF_LAST_USED	1351	X'00000547'

MQIACH_ * (Formáty celočíselných kanálů formátu příkazu)

Tabulka 185. 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'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACH_INDOUBT_STATUS	1528	X'000005F8'
MQIACH_LAST_SEQ_NUMBER	1529	X'000005F9'
MQIACH_LAST_SEQUENCE_NUMBER	1529	X'000005F9'
MQIACH_CURRENT_MSGS	1531	X'000005FB'
MQIACH_CURRENT_SEQ_NUMBER	1532	X'000005FC'
MQIACH_CURRENT_SEQUENCE_NUMBER	1532	X'000005FC'
MQIACH_SSL_RETURN_CODE	1533	X'000005FD'
MQIACH_MSGS	1534	X'000005FE'
MQIACH_BYTES_SENT	1535	X'000005FF'
MQIACH_BYTES_RCVD	1536	X'00000600'
MQIACH_BYTES_RECEIVED	1536	X'00000600'
MQIACH_BATCHES	1537	X'00000601'
MQIACH_BUFFERS_SENT	1538	X'00000602'
MQIACH_BUFFERS_RCVD	1539	X'00000603'
MQIACH_BUFFERS_RECEIVED	1539	X'00000603'
MQIACH_LONG_RETRIES_LEFT	1540	X'00000604'
MQIACH_SHORT_RETRIES_LEFT	1541	X'00000605'
MQIACH_MCA_STATUS	1542	X'00000606'
MQIACH_STOP_REQUESTED	1543	X'00000607'
MQIACH_MR_COUNT	1544	X'00000608'
MQIACH_MR_INTERVAL	1545	X'00000609'
MQIACH_NPM_SPEED	1562	X'0000061A'
MQIACH_HB_INTERVAL	1563	X'0000061B'
MQIACH_BATCH_INTERVAL	1564	X'0000061C'
MQIACH_NETWORK_PRIORITY	1565	X'0000061D'
MQIACH_KEEP_ALIVE_INTERVAL	1566	X'0000061E'
MQIACH_BATCH_HB	1567	X'0000061F'
MQIACH_SSL_CLIENT_AUTH	1568	X'00000620'
MQIACH_ALLOC_RETRY	1570	X'00000622'
MQIACH_ALLOC_FAST_TIMER	1571	X'00000623'
MQIACH_ALLOC_SLOW_TIMER	1572	X'00000624'
MQIACH_DISC_RETRY	1573	X'00000625'
MQIACH_PORT_NUMBER	1574	X'00000626'
MQIACH_HDR_COMPRESSION	1575	X'00000627'
MQIACH_MSG_COMPRESSION	1576	X'00000628'
MQIACH_CLWL_CHANNEL_RANK	1577	X'00000629'
MQIACH_CLWL_CHANNEL_PRIORITY	1578	X'0000062A'
MQIACH_CLWL_CHANNEL_WEIGHT	1579	X'0000062B'
MQIACH_CHANNEL_DISP	1580	X'0000062C'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACH_INBOUND_DISP	1581	X'0000062D'
MQIACH_CHANNEL_TYPES	1582	X'0000062E'
MQIACH_ADAPS_STARTED	1583	X'0000062F'
MQIACH_ADAPS_MAX	1584	X'00000630'
MQIACH_DISPS_STARTED	1585	X'00000631'
MQIACH_DISPS_MAX	1586	X'00000632'
MQIACH_SSLTASKS_STARTED	1587	X'00000633'
MQIACH_SSLTASKS_MAX	1588	X'00000634'
MQIACH_CURRENT_CHL	1589	X'00000635'
MQIACH_CURRENT_CHL_MAX	1590	X'00000636'
MQIACH_CURRENT_CHL_TCP	1591	X'00000637'
MQIACH_CURRENT_CHL_LU62	1592	X'00000638'
MQIACH_ACTIVE_CHL	1593	X'00000639'
MQIACH_ACTIVE_CHL_MAX	1594	X'0000063A'
MQIACH_ACTIVE_CHL_PAUSED	1595	X'0000063B'
MQIACH_ACTIVE_CHL_STARTED	1596	X'0000063C'
MQIACH_ACTIVE_CHL_STOPPED	1597	X'0000063D'
MQIACH_ACTIVE_CHL_RETRY	1598	X'0000063E'
MQIACH_LISTENER_STATUS	1599	X'0000063F'
MQIACH_SHARED_CHL_RESTART	1600	X'00000640'
MQIACH_LISTENER_CONTROL	1601	X'00000641'
MQIACH_BACKLOG	1602	X'00000642'
MQIACH_XMITQ_TIME_INDICATOR	1604	X'00000644'
MQIACH_NETWORK_TIME_INDICATOR	1605	X'00000645'
MQIACH_EXIT_TIME_INDICATOR	1606	X'00000646'
MQIACH_BATCH_SIZE_INDICATOR	1607	X'00000647'
MQIACH_XMITQ_MSGS_AVAILABLE	1608	X'00000648'
MQIACH_CHANNEL_SUBSTATE	1609	X'00000649'
MQIACH_SSL_KEY_RESETS	1610	X'0000064A'
MQIACH_COMPRESSION_RATE	1611	X'0000064B'
MQIACH_COMPRESSION_TIME	1612	X'0000064C'
MQIACH_MAX_XMIT_SIZE	1613	X'0000064D'
MQIACH_DEF_CHANNEL_DISP	1614	X'0000064E'
MQIACH_SHARING_CONVERSATIONS	1615	X'0000064F'
MQIACH_MAX_SHARING_CONVS	1616	X'00000650'
MQIACH_CURRENT_SHARING_CONVS	1617	X'00000651'
MQIACH_MAX_INSTANCES	1618	X'00000652'
MQIACH_MAX_INSTS_PER_CLIENT	1619	X'00000653'
MQIACH_CLIENT_CHANNEL_WEIGHT	1620	X'00000654'

Tabulka 185. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
MQIACH_CONNECTION_AFFINITY	1621	X'00000655'
MQIACH_RESET_REQUESTED	1623	X'00000657'
MQIACH_BATCH_DATA_LIMIT	1624	X'00000658'
MQIACH_MSG_HISTORY	1625	X'00000659'
MQIACH_MULTICAST_PROPERTIES	1626	X'0000065A'
MQIACH_NEW_SUBSCRIBER_HISTORY	1627	X'0000065B'
MQIACH_MC_HB_INTERVAL	1628	X'0000065C'
MQIACH_USE_CLIENT_ID	1629	X'0000065D'
MQIACH_MQTT_KEEP_ALIVE	1630	X'0000065E'
MQIACH_IN_DOUBT_IN	1631	X'0000065F'
MQIACH_IN_DOUBT_OUT	1632	X'00000660'
MQIACH_MSGS_SENT<	1633	X'00000661'
MQIACH_MSGS_RECEIVED	1634	X'00000662'
MQIACH_MSGS_RCVD	1634	X'00000662'
MQIACH_PENDING_OUT	1635	X'00000663'
MQIACH_AVAILABLE_CIPHERSPECS	1636	X'00000664'
MQIACH_MATCH	1637	X'00000665'
MQIACH_USER_SOURCE	1638	X'00000666'
MQIACH_WARNING	1639	X'00000667'
MQIACH_DEF_RECONNECT	1640	X'00000668'
MQIACH_CHANNEL_SUMMARY_ATTRS	1642	X'0000066A'
MQIACH_LAST_USED	1642	X'0000066A'

MQIAMO_ * (Typy parametrů parametru monitorování celého formátu příkazu)

Tabulka 186. 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'
MQIAMO_CONNS_MAX	713	X'000002C9'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
MQIAMO_MSGS_EXPIRED	758	X'000002F6'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
MQIAMO_MCAST_BATCH_TIME	802	X'00000322'

Tabulka 186. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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 187. 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 188. 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_ * (Ověřovatel záhlaví informačního systémuIMS)

Tabulka 189. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQIAUT_NONE	"¬¬¬¬¬¬¬¬"
MQIAUT_NON_ARRAY	'¬','¬','¬','¬','¬','¬','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQIAV_ * (hodnoty celočíselných atributů)

Tabulka 190. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIAV_NOT_APPLICABLE	-1	X'FFFFFFFF'
MQIAV_UNDEFINED	-2	X'FFFFFFFE'

MQICM_ * (režimy vázaného zpracování záhlaví informačního systémuIMS)

Tabulka 191. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQICM_COMMIT_THEN_SEND	'0'
MQICM_SEND_THEN_COMMIT	'1'

MQIDO_ * (Nejisté volby příkazového formátu)

Tabulka 192. 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 193. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQIEP_	"IEP¬"
MQIEP_STRUC_ID_POLE	'I','E','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 194. 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 195. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIGQ_DISABLED	0	X'00000000'
MQIGQ_ENABLED	1	X'00000001'

MQIGQPA_ * (Oprávnění k zařazení do fronty v rámci skupiny)

Tabulka 196. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIGQPA_DEFAULT	1	X'00000001'

Tabulka 196. 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 obsahuIMS)

Struktura záhlaví informací systému IMS

Tabulka 197. Konstrukce konstant	
Název	**Struktura**
MQIIH_STRUCTION_ID	"IIH¬"
MQIIH_STRUC_ID_POLE	'I','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 198. 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 IMS

Tabulka 199. 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 200. Konstrukce konstant	
Název	**Struktura**
MQIMPO_STRUCT	"IMPO"
MQIMPO_STRUC_ID_ARRAY	'I','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 201. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIMPO_VERSION_1	1	X'00000001'

Tabulka 201. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VERZE AKTUÁLNÍ_VERZE MQIMPO_CURRENT_VERSION	1	X'00000001'

Zjistit volby vlastností zprávy

Tabulka 202. 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 203. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQINBD_Q_MGR	0	X'00000000'
SKUPINA MQINBD_GROUP	3	X'00000003'

MQIND_ * (Speciální hodnoty indexu)

Tabulka 204. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIND_NONE	-1	X'FFFFFFFF'
MQINDAL_VŠE	-2	X'FFFFFFFE'

MQIPADDR_ * (Verze IP adres)

Tabulka 205. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQIPADDR_IPV4	0	X'00000000'
MQIPADDR_IPV6	1	X'00000001'

MQISS_ * (Rozsahy zabezpečení záhlaví informačního záhlaví systémuIMS)

Tabulka 206. Konstantní názvy a hodnoty	
Název	**Hodnota**
KONTROLA MQISS_CHECK	'C'
MQISS_FULL	'F'

MQIT_ * (Typy indexů)

Tabulka 207. 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 208. 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_ * (identifikátor instance transakce záhlaví informačního záhlaví systémuIMS)

Tabulka 209. 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 210. 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 211. Hodnoty konstant

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

MQMASTER_ * (hlavní administrace)

Tabulka 212. 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 213. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCAS_STOPPED	0	X'00000000'
MQMCAS_RUNNING	3	X'00000003'

MQMCAT_ * (typy MCA)

Tabulka 214. 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 215. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMCD_FOLDER_VERSION	1	X'00000001'

Názvy značek voleb značek publikování/odběru

Tabulka 216. 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 217. 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 217. 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 218. 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 219. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQM_STRUCT	"MD¬¬"
POLE MQMD_STRUC_ID_ARRAY	'M','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 220. 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 221. Konstrukce konstant	
Název	**Struktura**
MQM_STRUCTION_ID	"MDE¬"
MQM_STRUC_STRUC_ID_POLE	'M','D','E','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 222. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDE_VERSION_2	2	X'00000002'

Tabulka 222. 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 223. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQMDEF_NONE	0	X'00000000'

MQMDS_ * (Posloupnost doručení zprávy)

Tabulka 224. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PRIORITA MQMS_PRIORITY	0	X'00000000'
MQSD_FIFO	1	X'00000001'

MQMF_ * (Příznaky zprávy)

Tabulka 225. 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 226. Konstrukce konstant	
Název	**Struktura**
MQMHBO_STRUCTION_ID	"MHBO"
MQMHBO_STRUC_ID_ARRAY	'M','H','B','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 227. 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 228. 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 229. 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 230. Hodnoty konstant

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

MQMO_ * (Volby shody)

Tabulka 231. 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 232. 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 233. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQMON_NOT_AVAILABLE	-1	X'FFFFFFFF'
MQMON_NONE	-1	X'FFFFFFFF'

Tabulka 233. 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 234. 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 235. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQMTOK_NONE	X'00...00' (16 nul)
MQMTOKEN_NE_ARRAY	'\0','\0',... (16 nul)

MQNC_ * (Počet názvů)

Tabulka 236. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POČET NÁZVŮ MQNC_MAX_NAMELIST_NAME_COUNT	256	X'00000100'

MQNPM_ * (přechodná třída zpráv)

Tabulka 237. 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 238. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQNPMS_NORMAL	1	X'00000001'
MQNPMS_FAST	2	X'00000002'

MQNT_ * (Typy seznamu názvů)

Tabulka 239. 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 240. 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 241. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
NEJPRVE MQOA_	1	X'00000001'
MQOA_LAST	9000	X'00002328'

MQOD_ * (Struktura deskriptoru objektu)

Tabulka 242. Konstrukce konstant

Název	Struktura
MQOD_STRUCTION_ID	"OD¬¬"
MQOD_STRUC_ID_POLE	'O','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 243. 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'

Tabulka 243. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VERZE AKTUÁLNÍ_VERZE	4	X'00000004'
AKTUÁLNÍ_DÉLKA_AKTUÁLNÍ_HODNOTY	(value differs by platform or version)	

MQOII_ * (Identifikátor instance objektu)

Tabulka 244. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQOII_NONE	X'00...00' (24 nul)
MQOII_NON_ARRAY	'\0','\0',... (24 nul)

MQOL_ * (Původní délka)

Tabulka 245. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOL_UNDEFINED	-1	X'FFFFFFFF'

MQOM_ * (Zastaralé Db2 Volby zpráv ve skupině zjišťování)

Tabulka 246. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOM_NO	0	X'00000000'
MQOM_ANO	1	X'00000001'

MQOO_ * (Otevřít volby)

Tabulka 247. 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'
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'

Tabulka 247. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 248. 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 249. 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 250. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQOP_REGISTER	256	X'00000100'
MQOP_DEREGISTRACI	512	X'00000200'

Operační kódy pro MQCTL a MQCB

Tabulka 251. 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 252. 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 253. 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 254. 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'
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 255. 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'

MQPA_ * (Oprávnění k vložení)

Tabulka 256. 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 257. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY OBJEKTU MQPD_BEAN	"PD¬¬"
POLE MQPD_STRUC_ID_ARRAY	'P','D','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 258. 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 259. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NONE	0	X'00000000'

Volby podpory vlastností

Tabulka 260. 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 261. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPD_NO_CONTEXT	0	X'00000000'
KONTEXT MQPD_USER_CONTEXT	1	X'00000001'

MQPER_ * (hodnoty perzistence)

Tabulka 262. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPER_PERSISTENCE_AS_PARENT	-1	X'FFFFFFFF'
MQPER_NOT_PERSISTENT	0	X'00000000'
MQPER_PERSISTENT	1	X'00000001'
MQPER_PERSISTENCE_AS_Q_DEF	2	X'00000002'
MQPER_PERSISTENCE_AS_TOPIC_DEF	2	X'00000002'

MQPL_ * (platformy)

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_NSS	13	X'0000000D'
MQPL_OPEN_TP1	15	X'0000000F'
MQPL_VM	18	X'00000012'
MQPL_TPF	23	X'00000017'
MQPL_VSE	27	X'0000001B'
MQPL_NATIVNÍ	1	X'00000001'

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

Vložit strukturu voleb zprávy

Tabulka 263. Konstrukce konstant	
Název	**Struktura**
MQPMO_STRUC_ID	"PMO¬"
MQPMO_STRUC_ID_POLE	'P','M','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 264. 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'
AKTUÁLNÍ_DÉLKA MQPMO_LENGTH	(value differs by platform or version)	(value differs by platform or version)

Volby vložení zprávy

Tabulka 265. 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'

Tabulka 265. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 266. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMO_POPB_VOLBA_VOLBY	2097152	X'00200000'

MQPMRF_ * (Vložení polí záznamu zprávy)

Tabulka 267. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPMRF_ID_ZPRÁVY	1	X'00000001'
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 268. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPO_YES	1	X'00000001'
MQPO_NO	0	X'00000000'

MQPRI_* (priorita)

Tabulka 269. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPRI_PRIORITY_AS_Q_DEF	-1	X'FFFFFFFF'

Tabulka 269. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 270. 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 271. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPROP_UNRESTRICTED_LENGTH	-1	X'FFFFFFFF'

MQPRT_ * (Put Response Values)

Tabulka 272. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPRT_RESPONSE_AS_PARENT	0	X'00000000'
MQPRT_SYNC_RESPONSE	1	X'00000001'
ODEZVA MQPRT_ASYNC_RESPONSE	2	X'00000002'

MQPS_ * (Publikování/odběr)

Stav publikování/odběru ve formátu příkazu

Tabulka 273. 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

Tabulka 274. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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

Tabulka 275. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
PŘÍKAZ MQPS_COMMAND_B	"bMQPSCommandb"	
MQPS_COMP_CODE_B	"bMQPSCompCodeb"	
MQPS_CORRELA_ID_B	"bMQPSCorrelIdb"	
MQPS_DELETE_OPTIONS_B	"bMQPSDelOptsb"	
MQPS_ERROR_ID_B	"bMQPSErrorIdb"	
MQPS_ERROR_POS_B	"bMQPSErrorPosb"	
MQPS_INTEGER_DATA_B	"bMQPSIntDatab"	
MQPS_PARAMETER_ID_B	"bMQPSParmIdb"	
MQPS_PUBLICATION_OPTIONS_B	"bMQPSPubOptsb"	
MQPS_PUBLISH_TIMESTAMP_B	"bMQPSPubTimeb"	
MQPS_Q_MGR_NAME_B	"bMQPSQMgrNameb"	

Tabulka 275. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPS_Q_NÁZEVB	"bMQPSQNameb"	
MQPS_REASON_B	"bMQPSReasonb"	
MQPS_REASON_TEXT_B	"bMQPSReasonTextb"	
VOLBY REGISTRACE MQPS_REGISTRATIONS_B	"bMQPSRegOptsb"	
MQPS_SEQUENCE_NUMBER_B	"bMQPSSeqNumb"	
MQPS_STREAM_NAME_B	"bMQPSStreamNameb"	
MQPS_STRING_DATA_B	"bMQPSStringDatab"	
MQPS_SUBSCRIPTION_IDENTITY_B	"bMQPSSubIdentityb"	
MQPS_SUBSCRIPTION_NAME_B	"bMQPSSubNameb"	
MQPS_SUBSCRIPTION_USER_DATA_B	"bMQPSSubUserDatab"	
MQPS_TOPIC_B	"bMQPSTopicb"	
MQPS_USER_ID_B	"bMQPSUserIdb"	

Hodnoty značek příkazů publikování/odběru jako řetězce

Tabulka 276. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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

Tabulka 277. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPS_DELETE_PUBLICATION_B	"bDeletePubb"	
MQPS_DEREGISTER_PUBLISHER_B	"bDeregPubb"	
MQPS_DEREGISTER_SUBSCRIBER_B	"bDeregSubb"	
MQPS_PUBLISH_B	"bPublishb"	
MQPS_REGISTER_PUBLISHER_B	"bRegPubb"	
MQPS_REGISTER_SUBSCRIBER_B	"bRegSubb"	
MQPS_REQUEST_UPDATE_B	"bReqUpdateb"	

Hodnoty značek voleb publikování/odběru jako řetězce

Tabulka 278. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NÁZEV_ADR_MQP	"AddName"	

Tabulka 278. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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

Tabulka 279. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NÁZEV_SADY_MQ_ADR_B	"bAddNameb"	
MQPS_ANONYMOUS_B	"bAnonb"	
MQPS_CORRELAC_ID_AS_IDENTITY_B	"bCorrelAsIdb"	
MQPS_DEREGISTER_ALL_B	"bDeregAllb"	
MQPS_DIRECT_REQUESTS_B	"bDirectReqb"	
MQPS_DUPLICATES_OK_B	"bDupsOKb"	
MQPS_FULL_RESPONSE_B	"bFullRespb"	
MQPS_INCLUDE_STREAM_NAME_B	"bInclStreamNameb"	
MQPS_INFORCE_IF_RETAINED_B	"bInformIfRetb"	

Tabulka 279. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPS_IS_RETAINED_PUBLICATION_B	"bIsRetainedPubb"	
MQPS_JOIN_EXCLUSIVE_B	"bJoinExclb"	
SDÍLENÝ_SDÍLENÝ_NÁZEV_SDÍLENÉ_FRONTY	"bJoinSharedb"	
POUZE MQPS_LEAV_ONLY_B	"bLeaveOnlyb"	
MQPS_LOCAL_B	"bLocalb"	
MQPS_LOCKED_B	"bLockedb"	
MQPS_NEW_PUBLICATIONS_ONLY_B	"bNewPubsOnlyb"	
MQPS_NO_ALTERATION_B	"bNoAlterb"	
MQPS_NO_REGISTRATION_B	"bNoRegb"	
MQPS_NON_PERSISTENT_B	"bNonPersb"	
MQPS_NON_B	"bNoneb"	
POUZE KEM_SUBSCRIBERS_ONLY_B	"bOtherSubsOnlyb"	
MQPS_PERSISTENT_B	"bPersb"	
MQPS_PERSISTENT_AS_PUBLISH_B	"bPersAsPubb"	
MQPS_PERSISTENT_AS_Q_B	"bPersAsQueueb"	
MQPS_PUBLISH_ON_REQUEST_ONLY_B	"bPubOnReqOnlyb"	
MQPS_RETAIN_PUBLICTION_B	"bRetainPubb"	
MQPS_VARIABLE_USER_ID_B	"bVariableUserIdb"	

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

Tabulka 280. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSC_FOLDER_VERSION	1	X'00000001'

MQPSC_ * (Názvy značek publikování/odběru značek)

Tabulka 281. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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"	

Tabulka 281. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSC_SUBSCRIPTION_IDENTITY	"SubIdentity"	
MQPSC_SUBSCRIPTION_USER_DATA	"SubUserData"	
MQPSC_CORREL_ID	"CorrelId"	

MQPSC_ * (Názvy značek publikování/odběru značek XML)

Tabulka 282. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (hodnoty značky voleb publikování/odběru jako řetězce)

Tabulka 283. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSC_DELETE_PUBLICATION	"DeletePub"	
MQPSC_DEREGISTER_SUBSCRIBER	"DeregSub"	
PUBLIKOVÁNÍ MQPSC_PUBLISH	"Publish"	
MQPSC_REGISTER_SUBSCRIBER	"RegSub"	
MQPSC_REQUEST_UPDATE	"ReqUpdate"	

MQPSC_ * (hodnoty značky voleb publikování/odběru jako řetězce)

Tabulka 284. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 285. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSCR_FOLDER_VERSION	1	X'00000001'

Názvy značek voleb značek publikování/odběru

Tabulka 286. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
DOKONČENÍ MQPSCR_COMPLETION	"Completion"	
MQPSCR_RESPONSE	"Response"	
MQPSCR_REASON	"Reason"	

Názvy značek XML značek voleb pro publikování/odběr

Tabulka 287. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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

Tabulka 288. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQPSCR_OK	"ok"	
VAROVÁNÍ MQPSCR_WARNING	"warning"	
CHYBA MQPSCR_ERROR	"error"	

MQPSM_ * (režim/odběr-režim)

Tabulka 289. 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 290. 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 291. 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 292. 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 293. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQPXP_STRUCT	"PXP¬"
POLE MQPXP_STRUC_ID_ARRAY	'P','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 294. 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 295. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_GET_INHIBED	1	X'00000001'
MQQA_GET_ALLOWED	0	X'00000000'

Blokovat hodnoty Put

Tabulka 296. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_PUT_BLOKOVÁNO	1	X'00000001'
MQQA_PUT_ALLOWED	0	X'00000000'

Sdílitelnost fronty

Tabulka 297. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQA_SHAREABLE	1	X'00000001'
MQQA_NOT_SHAREABLE	0	X'00000000'

Upevňování zadního zatížení

Tabulka 298. 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 299. 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 300. 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 301. 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 302. 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 303. 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 304. 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 305. 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 306. Hodnoty konstant

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

MQQSGD_ * (Dispozice skupiny sdílení front)

Tabulka 307. 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 příkazů QSG ve formátu příkazu)

Tabulka 308. 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 309. 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 310. 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 311. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQSOT_ALL	1	X'00000001'
MQQSOT_INPUT	2	X'00000002'

Tabulka 311. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝSTUP MQQSOT_OUTPUT	3	X'00000003'

MQQSUM_ * (Zprávy ve stavu fronty nepotvrzené zprávy)

Tabulka 312. 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 313. 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 314. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQQ_VŠE	1001	X'000003E9'

MQRC_ * (Kód příčiny)

Tabulka 315. 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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 315. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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_ * (Kód příčiny záhlaví příkazového řádku)

Tabulka 316. 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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'
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'

Tabulka 316. Hodnoty konstant (pokračování)

Název	Desetinná hodnota	Hexadecimální hodnota
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'

MQRCN_ * (Konstanty opětovného připojení klienta)

Tabulka 317. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRCN_NO	0	X'00000000'
MQRCN_YES	1	X'00000001'
MQRCN_Q_MGR	2	X'00000002'

Tabulka 317. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRCN_DISABLED	3	X'00000003'

MQRCVTIME_ * (Typy časových limitů příjmu)

Tabulka 318. 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 319. 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 320. 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 321. 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'
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'

Tabulka 321. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 322. Konstrukce konstant	
Název	**Struktura**
MQRFH_STRUCT	"RFH¬"
MQRFH_STRUC_ID_POLE	'R','F','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 323. 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 324. 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 325. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRFH2_NAME_VALUE_VERSION	1	X'00000001'

MQRFH2_* (Názvy značek publikování a odběru značek)

Tabulka 326. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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)

Tabulka 327. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 328. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRL_UNDEFINED	-1	X'FFFFFFFF'

MQRMH_ * (Struktura záhlaví referenční zprávy)

Tabulka 329. Konstrukce konstant	
Název	**Struktura**
MQRMH_STRUCTION_ID	"RMH¬"
MQRMH_STRUC_ID_ARRAY	'R','M','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 330. 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 331. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRMHF_LAST	1	X'00000001'
MQRMHF_NOT_LAST	0	X'00000000'

MQRO_ * (Volby sestav)

Tabulka 332. 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 333. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQRO_REJECT_UNSUP_MASK	270270464	X'101C0000'
MQRO_ACCEPT_UNSUP_MASK	-270532353	X'EFE000FF'
MQRO_ACCEPT_UNSUP_IF_XMIT_MASK	261888	X'0003FF00'

MQROUTE_ * (trasová-trasa)

Maximální počet aktivit trasování cesty (MQIACF_MAX_ACTIVITIES)

Tabulka 334. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQROUTE_UNLIMITED_ACTIVITIES	0	X'00000000'

Podrobnosti trasy trasování (MQIACF_ROUTE_DETAIL)

Tabulka 335. 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 336. 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 337. 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 338. 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 339. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRP_ANO	1	X'00000001'
MQRP_NO	0	X'00000000'

MQRQ_ * (Kvalifikátory důvodu formátu příkazu)

Tabulka 340. 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'

Tabulka 340. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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'

MQRT_ * (Typy obnovení příkazového formátu)

Tabulka 341. 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 342. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQRU_PUBLISH_ON_REQUEST	1	X'00000001'
MQRU_PUBLISH_ALL	2	X'00000002'

MQSCA_ * (ověření klienta připojeného přes SSL)

Tabulka 343. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
POŽADOVÁNO MQSCA_REQUIRED	0	X'00000000'
MQSCA_OPTIONAL	1	X'00000001'

MQSCO_ * (volby konfigurace SSL)

Struktura voleb konfigurace SSL

Tabulka 344. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQSCO_	"SCO¬"
POLE MQSCO_STRUC_ID_ARRAY	'S','C','O','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 345. 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 SSL

Tabulka 346. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
VÝCHOZÍ HODNOTA MQSCO_RESET_COUNT_DEFAULT	0	X'00000000'

Rozsah definice definice fronty příkazů

Tabulka 347. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCOM_Q_MGR	1	X'00000001'
BUŇKA MQSCO_CELL	2	X'00000002'

MQSCOPE_ * (obor publikování)

Tabulka 348. 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 349. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSCYC_UPPER	0	X'00000000'
MQSCYC_SMÍŠENÝ	1	X'00000001'

MQSD_ * (struktura deskriptoru objektu)

Tabulka 350. 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 351. 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 352. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSECITEM_VŠE	0	X'00000000'
MQSECITEM_MQADMIN	1	X'00000001'
TABULKA MQSECITEM_MQNLIST	2	X'00000002'
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'

MQSECSW_ * (Přepínače zabezpečení a přepínače příkazů ve formátu příkazu)

Přepínače zabezpečení ve formátu příkazu

Tabulka 353. 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'

Tabulka 353. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 354. 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 355. 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 356. Konstantní názvy a hodnoty	
Název	**Hodnota**
MQSEG_BLOKOVÁNO	'¬'
MQSEG_ALLOWED	'A'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQSEL_ * (speciální hodnoty selektoru)

Tabulka 357. 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 358. 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 359. 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 360. 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 361. Konstrukce konstant

Název	Struktura
MQSMPO_STRUCTION_ID	"SMPO"
POLE MQSMPO_STRUC_ID_ARRAY	'S','M','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 362. 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 363. 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'

Tabulka 363. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSMPO_NONE	0	X'00000000'

MQSO_ * (Volby odběru)

Tabulka 364. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSO_NONE	0	X'00000000'
MQSO_NON_DURABLE	0	X'00000000'
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 365. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSP_AVAILABLE	1	X'00000001'
MQSP_NOT_AVAILABLE	0	X'00000000'

MQSQQM_ * (Název správce front sdílené fronty)

Tabulka 366. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSQQM_USE	0	X'00000000'
MQSQQM_IGNORE	1	X'00000001'

MQSR_ * (Akce)

Tabulka 367. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSR_ACTION_PUBLICATION	1	X'00000001'

MQSRO_ * (Struktura voleb požadavku na odběr)

Tabulka 368. 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 369. 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 370. 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 SSL FIPS)

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

MQSTAT_ * (volby statistiky)

CHYBA MQSTAT_TYPE_ASYNC_ERROR	0	X'00000000'
PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION	0	X'00000000'
CHYBA PŘI CHYBĚ MQSTAT_TYPE_RECONNECTION_ERROR	0	X'00000000'

MQSTS_ * (Struktura struktury tvorby sestav stavu)

Tabulka 372. 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 373. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSTS_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQSTS_CURRENT_VERSION	1	X'00000001'

MQSUB_ * (trvalé odběry)

Trvalé odběry

Tabulka 374. 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'

Trvalé odběry

Tabulka 375. 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 376. 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 377. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSUS_YES	1	X'00000001'
MQSUS_NO	0	X'00000000'

MQSVC_ * (Služba)

Typy služeb

Tabulka 378. 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 379. 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 380. 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 381. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYNCPOINT_YES	0	X'00000000'
MQSYNCPOINT_IFPER	1	X'00000001'

MQSYSP_ * (Systémové hodnoty parametrů systému)

Tabulka 382. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQSYSP_NO	0	X'00000000'
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'

Tabulka 382. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 383. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_BLOCK	1	X'00000001'
MQTA_PASSTHRU	2	X'00000002'

Povolené odběry

Tabulka 384. 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 385. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTA_PROXY_SUB_FORCE	1	X'00000001'
MQTA_PROXY_SUB_FIRSTUSE	2	X'00000002'

Publikace povoleny

Tabulka 386. 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 387. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTC_OFF	0	X'00000000'
MQTC_ON	1	X'00000001'

MQTCPKEEP_ * (TCP Keepalive)

Tabulka 388. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTCPKEEP_NO	0	X'00000000'
MQTCPKEEP_YES	1	X'00000001'

MQTCPSTACK_ * (Typy zásobníků TCP)

Tabulka 389. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTCPSTACK_SINGLE	0	X'00000000'
MQTCPSTACK_MULTIPLE	1	X'00000001'

MQTIME_ * (jednotky času formátu příkazu)

Tabulka 390. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTIME_JEDNOTKA_MIN	0	X'00000000'
MQTIME_JEDNOTKY_SEKUNDY	1	X'00000001'

MQTM_ * (Struktura zprávy spouštěče)

Tabulka 391. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQTM_STRUCT	"TM¬¬"
POLE MQTM_STRUC_ID_ARRAY	'T','M','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 392. 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 393. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQTC_STRUCT	"TMC¬"
POLE MQTC_STRUC_ID_ARRAY	'T','M','C','¬'
MQTMC_VERSION_1	"¬¬¬1"

Tabulka 393. Konstrukce konstant (pokračování)	
Název	**Struktura**
MQTMC_VERSION_2	"¬¬¬2"
AKTUÁLNÍ_VERZE MQTMC_VERSION	"¬¬¬2"
MQTMC_VERSION_1_ARRAY	'¬','¬','¬','1'
MQTMC_VERSION_2_ARRAY	'¬','¬','¬','2'
POLE MQTMC_CURRENT_VERSION_VERSION_ARRAY	'¬','¬','¬','2'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

MQTOPT_ * (typ tématu)

Tabulka 394. 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 395. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTRAXSTR_NO	0	X'00000000'
MQTRAXSTR_YES	1	X'00000001'

MQTSCOPE_ * (Obor odběru)

Tabulka 396. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTSCOPY_QMGR	1	X'00000001'
MQTSCOPY_ALL	2	X'00000002'

MQTT_ * (Typy spouštěčů)

Tabulka 397. 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 398. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQTYPE_AS_SET	0	X'00000000'
MQTYPE_NULL	2	X'00000002'

Tabulka 398. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 399. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
NEJPRVE MQUA_	65536	X'00010000'
MQUA_LAST	999999999	X'3B9AC9FF'

MQUIDSUPPR_ * (Podpora ID uživatele formátu příkazu)

Tabulka 400. 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 401. 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 402. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQUOW_NONE	0	X'00000000'
MQUOWST_ACTIVE	1	X'00000001'
MQUOW_PREPART	2	X'00000002'
MQUOW_UNRESOLVED	3	X'00000003'

MQUOWT_ * (Formát příkazů pracovní jednotky)

Tabulka 403. 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 404. 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 405. 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_EXPAND_USER	1	X'00000001'
MQUSAGE_EXPAND_SYSTEM	2	X'00000002'
MQUSAGE_EXPAND_NONE	3	X'00000003'

Hodnoty použití datové sady formátu příkazu

Tabulka 406. 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 407. Hodnoty konstant

Název	Desetinná hodnota	Hexadecimální hodnota
MQVL_NULL_ UKONČENO	-1	X'FFFFFFFF'
MQVL_EMPTY_STRING	0	X'00000000'

MQVU_ * (variabilní ID uživatele)

Tabulka 408. 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 409. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQWDR_	"WDR¬"
POLE MQWDR_STRUC_ID_ARRAY	'W','D','R','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 410. 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 411. 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 412. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQWIH_	"WIH¬"
MQWIH_STRUC_ID_POLE	'W','I','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 413. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWIH_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQWIH_CURRENT_VERSION	1	X'00000001'

Tabulka 413. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWIH_LENGTH_1	120	X'00000078'
AKTUÁLNÍ_DÉLKA MQWIH_CURRENT_LENGTH	120	X'00000078'

Příznaky záhlaví informací o pracovní zátěži

Tabulka 414. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQWIH_NONE	0	X'00000000'

MQWQR_ * (Struktura záznamu výstupní fronty pracovní zátěže klastru)

Tabulka 415. 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 416. 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 417. 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 418. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQWXP_STRUCTION_ID	"WXP¬"

Tabulka 418. Konstrukce konstant (pokračování)	
Název	**Struktura**
POLE MQWXP_STRUC_ID_ARRAY	'W','X','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 419. 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 420. 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

MQXACT_ * (Typy volajících rozhraní API)

Tabulka 421. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXACT_EXTERNAL	1	X'00000001'
MQXACT_INTERNAL	2	X'00000002'

MQXC_ * (Výstupní příkazy)

Tabulka 422. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXC_MQOPEN	1	X'00000001'
MQXC_MQCLOSE	2	X'00000002'
MQXC_MQGET	3	X'00000003'
MQXC_MQPUT	4	X'00000004'
MQXC_MQPUT1	5	X'00000005'
MQXC_MQINQ	6	X'00000006'
MQXC_MQSET	8	X'00000008'
MQXC_MQBACK	9	X'00000009'
MQXC_MQCMIT	10	X'0000000A'

MQXCC_ * (Odezvy ukončení)

Tabulka 423. 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 424. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXDR_OK	0	X'00000000'
MQXDR_CONVERSION_FAILED	1	X'00000001'

MQXE_ * (Prostředí)

Tabulka 425. 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 426. Konstrukce konstant	
Název	**Struktura**
MQXE_STRUCTION_ID	"XEPO"
MQXEEPO_STRUC_ID_POLE	'X','E','P','O'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 427. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXEPO_VERSION_1	1	X'00000001'
MQXEEPO_AKTUÁLNÍ_VERZE	1	X'00000001'

Volby ukončení

Tabulka 428. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXEEPO_NONE	0	X'00000000'

MQXF_ * (identifikátory funkce rozhraní API)

Tabulka 429. 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'
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'

Tabulka 429. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXF_AXUNREG	35	X'00000023'

MQXP_ * (struktura parametru ukončení přeletu rozhraní API)

Tabulka 430. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQXP_STRUCTION_ID	"XP¬¬"
POLE MQXP_STRUC_ID_ARRAY	'X','P','¬','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 431. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXP_VERSION_1	1	X'00000001'

MQXPDA_ * (oblast určení problému)

Tabulka 432. 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 433. 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 434. Konstrukce konstant	
Název	**Struktura**
ID STRUKTURY MQXQ_STRUCTION_ID	"XQH¬"
MQXQHL_STRUC_ID_POLE	'X','Q','H','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 435. 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 436. 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 437. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXR2_PUT_WITH_DEF_ACTION	0	X'00000000'
MQXR2_PUT_WITH_DEF_USERID	1	X'00000001'
MQXR2_PUT_WITH_MSG_USERID	2	X'00000002'
MQXR2_USE_AGENT_BUFFER	0	X'00000000'
MQXR2_USE_EXIT_BUFFER	4	X'00000004'
MQXR2_DEFAULT_CONTINUATION	0	X'00000000'
MQXR2_CONTINUE_CHAIN	8	X'00000008'
MQXR2_SUPPRESS_CHAIN	16	X'00000010'

Tabulka 437. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXR2_STATIC_CACHE	0	X'00000000'
MQXR2_DYNAMIC_CACHE	32	X'00000020'

MQXT_ * (Identifikátory konce)

Tabulka 438. 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 439. 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 440. 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 441. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQXWD_VERSION_1	1	X'00000001'

MQZAC_ * (Struktura kontextu aplikace)

Tabulka 442. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZAC_STRUCT	"ZAC¬"
MQZAC_STRUC_ID_ARRAY	'Z','A','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 443. 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 444. Konstrukce konstant	
Název	**Struktura**
ID_KONSTRUKCE_MQZAD_OBJEKTU	"ZAD¬"
MQZAD_STRUC_ID_POLE	'Z','A','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 445. 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 446. 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 447. 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'
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'

Tabulka 447. Hodnoty konstant (pokračování)		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
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 448. 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 449. 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 450. 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 451. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZED_STRUCT	"ZED¬"
POLE MQZED_STRUC_ID_ARRAY	'Z','E','D','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 452. 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 453. Konstrukce konstant	
Název	**Struktura**
ID_STRUKTURY MQZFP_STRUCT	"ZFP¬"
POLE MQZFP_STRUC_ID_	'Z','F','P','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 454. 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 455. Konstrukce konstant	
Název	**Struktura**
MQZIC_STRUCTION_ID	"ZIC¬"
MQZIC_STRUC_ID_POLE	'Z','I','C','¬'

Poznámka: Symbol ¬ představuje jeden prázdný znak.

Tabulka 456. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZIC_VERSION_1	1	X'00000001'
AKTUÁLNÍ_VERZE MQZIC_AKTUÁLNÍ_VERZE	1	X'00000001'

MQZID_ * (ID funkcí pro služby)

ID funkcí společná pro všechny služby

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

ID funkcí pro službu Autorita

Tabulka 458. 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 459. 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 460. 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 461. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZIO_PRIMARY	0	X'00000000'
MQZIO_SECONDARY	1	X'00000001'

MQZNS_ * (Verze rozhraní služby názvů)

Tabulka 462. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZNS_VERSION_1	1	X'00000001'

MQZSE_ * (Spuštění instalovatelných služeb-Indikátor výčtu)

Tabulka 463. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
SPUŠTĚNÍ MQZSE_START	1	X'00000001'
MQZ_CONTINUE	0	X'00000000'

MQZSL_ * (indikátor selektoru instalovatelných služeb)

Tabulka 464. 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 465. Hodnoty konstant		
Název	**Desetinná hodnota**	**Hexadecimální hodnota**
MQZTO_PRIMÁRNÍ	0	X'00000000'
MQZ_SEKUNDÁRNÍ	1	X'00000001'

MQZUS_ * (Verze rozhraní služby ID uživatele)

Tabulka 466. 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. Popisy, polí a deklarace jazyka pro příslušné jazyky s každým datovým typem.

Představení datových typů používaných v rozhraní MQI

Tento oddíl představuje datové typy použité v rozhraní MQI a poskytuje vám rady při jejich používání v podporovaných programovacích jazycích.

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:

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.
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. Při odesílání dat MQBYTE mezi správci front, kteří používají různé znakové sady nebo kódování, se data MQBYTE *nejsou* v žádném případě převádějí. 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.

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 uvedeny v části “Konstanty” na stránce 50
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-Získat-volby zprávy” na stránce 337 .

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 50.

Název elementárního datového typu	Datový typ	Popis
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).
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 jazyku 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).

Název elementárního datového typu	Datový typ	Popis
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.
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.

Název elementárního datového typu	Datový typ	Popis
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.
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.

Název elementárního datového typu	Datový typ	Popis
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 WebSphere MQ . Jedná se o stejný identifikátor, který je použit v trasování MQ a výpisů 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; jedná se o 16bajtovou hranici na systému IBM i a 8bajtovou hranici na jiných platformách. Některé programovací jazyky podporují typované ukazatele; rozhraní MQI je také používá v několika případech (například PMQCHAR a PMQLONG v programovacím jazyku C).
MQTID	Identifikátor podprocesu	Identifikátor podprocesu produktu WebSphere MQ . Jedná se o stejný identifikátor, který je použit v trasování MQ a výpisů 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.

Název elementárního datového typu	Datový typ	Popis
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
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

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

Název elementárního datového typu	Datový typ	Popis
PMQGMO	Ukazatel	Ukazatel na datovou strukturu typu MQGMO
PMQHCONFIG	Ukazatel	Ukazatel na data typu MQHCONFIG
PMQHCONN	Ukazatel	Ukazatel na data typu MQHCONN
PMQHMSG	Ukazatel	Ukazatel na data typu MQHMSG
PMQHOBJ	Ukazatel	Ukazatel na data typu MQHOBJ
PMQIIH.	Ukazatel	Ukazatel na datovou strukturu typu MQIIH
PMQIMPO.	Ukazatel	Ukazatel na datovou strukturu typu MQIMPO
PMQINT8	Ukazatel	Ukazatel na data typu MQINT8
PMQINT16	Ukazatel	Ukazatel na data typu MQINT16
PMQINT32	Ukazatel	Ukazatel na data typu MQINT32
PMQINT64	Ukazatel	Ukazatel na data typu MQINT64
PMQLONG	Ukazatel	Ukazatel na data typu MQLONG
PMQMD	Ukazatel	Ukazatel na strukturu typu MQMD
PMQMDE	Ukazatel	Ukazatel na datovou strukturu typu MQMDE
PMQMD1	Ukazatel	Ukazatel na datovou strukturu typu MQMD1
PMQMD2	Ukazatel	Ukazatel na datovou strukturu typu MQMD2
PMQMHBO	Ukazatel	Ukazatel na datovou strukturu typu MQMHBO
PMQOD	Ukazatel	Ukazatel na datovou strukturu typu MQOD
PMQOR	Ukazatel	Ukazatel na datovou strukturu typu MQOR
PMQPD	Ukazatel	Ukazatel na datovou strukturu typu MQPD
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

Název elementárního datového typu	Datový typ	Popis
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
PMQXQH	Ukazatel	Ukazatel na datovou strukturu typu MQXQH

Deklarace C

Datový typ	Zastupování
MQBOOL	`typedef MQLONG MQBOOL;`
MQBYTE	`typedef unsigned char MQBYTE;`

Datový typ	Zastupování
MQBYTE8	`typedef MQBYTE MQBYTE8[8];`
MQBYTE16	`typedef MQBYTE MQBYTE16[16];`
MQBYTE24	`typedef MQBYTE MQBYTE24[24];`
MQBYTE32	`typedef MQBYTE MQBYTE32[32];`
MQBYTE40	`typedef MQBYTE MQBYTE40[40];`
MQCHAR	`typedef char MQCHAR;`
MQCHAR4	`typedef MQCHAR MQCHAR4[4];`
MQCHAR8	`typedef MQCHAR MQCHAR8[8];`
MQCHAR12	`typedef MQCHAR MQCHAR12[12];`
MQCHAR20	`typedef MQCHAR MQCHAR20[20];`
MQCHAR28	`typedef MQCHAR MQCHAR28[28];`
MQCHAR32	`typedef MQCHAR MQCHAR32[32];`
MQCHAR48	`typedef MQCHAR MQCHAR48[48];`
MQCHAR64	`typedef MQCHAR MQCHAR64[64];`
MQCHAR128	`typedef MQCHAR MQCHAR128[128];`
MQCHAR256	`typedef MQCHAR MQCHAR256[256];`
MQFLOAT32	`typedef float MQFLOAT32;`
MQFLOAT64	`typedef double MQFLOAT64;`
KONFIGURACE MQHCONFIG	`typedef void MQPOINTER MQHCONFIG;`
MQHCONN	`typedef MQLONG MQHCONN;`
MQOBJ	`typedef MQLONG MQHOBJ;`

Datový typ	Zastupování
MQINT8	`typedef signed char MQINT8;`
MQINT16	`typedef short MQINT16;`

MQINT64	Na 64bitových systémech UNIX :
	`typedef long;`
	Na 32bitovém systému AIX, Solaris a HP-UX:
	`typedef int64_t;`
	V systémech IBM i, Linuxa z/OS:
	`typedef long long;`
	V systému Windows:
	`typedef _int64;`

MQLONG	V systému IBM i:
	`typedef long MQLONG;`
	jiné platformy:
	`if defined(MQ_64_BIT)`
	`  typedef int MQLONG;`
	`else`
	`  typedef long MQLONG;`

MQPID	`typedef MQLONG MQPID;`
MQPTR	`typedef void MQPOINTER MQPTR;`
MQTID	`typedef MQLONG MQTID;`
MQUINT8	`typedef unsigned char MQUINT8;`
MQUINT16	`typedef unsigned short MQUINT16;`

MQUINT64	Na 64bitových systémech UNIX :
	`typedef unsigned long;`
	Na 32bitovém systému AIX, Solaris a HP-UX:
	`typedef uint64_t;`
	V systémech IBM i, Linuxa z/OS:
	`typedef unsigned long long;`
	V systému Windows:
	`typedef unsigned _int64;`

Datový typ	Zastupování

MQULONG	V systému IBM i:
	`typedef unsigned long MQULONG;`
	jiné platformy:
	`if defined(MQ_64_BIT)`
	`    typedef unsigned int MQULONG;`
	`else`
	`    typedef unsigned long MQULONG;`

OBJEKT PMQBO	`typedef MQBO MQPOINTER PMQBO;`
PMQBOOL	`typedef MQBOOL MQPOINTER PMQBOOL;`
PMQBYTE	`typedef MQBYTE MQPOINTER PMQBYTE;`
PMQBYTE8	`typedef MQBYTE8[8] MQPOINTER PMQBYTE8[8];`
PMQBYTE16	`typedef MQBYTE16[16] MQPOINTER PMQBYTE16[16];`
PMQBYTE24	`typedef MQBYTE24[24] MQPOINTER PMQBYTE24[24];`
PMQBYTE32	`typedef MQBYTE32[32] MQPOINTER PMQBYTE32[32];`
PMQBYTE40	`typedef MQBYTE40[40] MQPOINTER PMQBYTE40[40];`
PMQBYTE128	`typedef MQBYTE128[128] MQPOINTER PMQBYTE128[128];`
PMQCHAR	`typedef MQCHAR MQPOINTER PMQCHAR;`
PMQCHAR4	`typedef MQCHAR4[4] MQPOINTER PMQCHAR4[4];`
PMQCHAR8	`typedef MQCHAR8[8] MQPOINTER PMQCHAR8[8];`
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];`

Datový typ	Zastupování
PMQCHAR128	`typedef MQCHAR128[128] MQPOINTER PMQCHAR128[128];`
PMQCHAR256	`typedef MQCHAR256[256] MQPOINTER PMQCHAR256[256];`
PMQCHAR264	`typedef MQCHAR264[264] MQPOINTER PMQCHAR264[264];`
PMQCIH	`typedef MQCIH MQPOINTER PMQCIH;`
PMQCNO	`typedef MQCNO MQPOINTER PMQCNO;`
PMQDLH	`typedef MQDLH MQPOINTER PMQDLH;`
PMQFUNC	`typedef void MQPOINTER PMQFUNC;`
PMQFLOAT32	`typedef MQFLOAT32 MQPOINTER PMQFLOAT32;`
PMQFLOAT64	`typedef MQFLOAT64 MQPOINTER PMQFLOAT64;`
PMQGMO	`typedef MQGMO MQPOINTER PMQGMO;`
PMQHCONFIG	`typedef MQHCONFIG MQPOINTER PMQHCONFIG;`
PMQHCONN	`typedef MQHCONN MQPOINTER PMQHCONN;`
PMQHOBJ	`typedef MQHOBJ MQPOINTER PMQHOBJ;`
PMQIIH.	`typedef MQIIH MQPOINTER PMQIIH;`
PMQINT8	`typedef MQINT8 MQPOINTER PMQINT8;`
PMQINT16	`typedef MQINT16 MQPOINTER PMQINT16;`
PMQLONG	`typedef MQLONG MQPOINTER PMQLONG;`
PMQMD	`typedef MQMD MQPOINTER PMQMD;`
PMQMD1	`typedef MQMD1[1] MQPOINTER PMQMD1[1];`
PMQMDE	`typedef MQMDE MQPOINTER PMQMDE;`
PMQOD	`typedef MQOD MQPOINTER PMQOD;`

Datový typ	Zastupování
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;`
PMQULELONG.	`typedef MQULONG MQPOINTER PMQULONG;`
PMQVOID	`typedef void MQPOINTER PMQVOID;`
PMQWIH	`typedef MQWIH MQPOINTER PMQWIH;`
PMQXQH	`typedef MQXQH MQPOINTER PMQXQH;`
PPMQBO	`typedef PMQBO MQPOINTER PPMQBO;`
PPMQBYTE	`typedef PMQBYTE MQPOINTER PPMQBYTE;`
PPMQCHAR	`typedef PMQCHAR MQPOINTER PPMQCHAR;`
PPMQCNO	`typedef PMQCNO MQPOINTER PPMQCNO;`
PPMQGMO	`typedef PMQGMO MQPOINTER PPMQGMO;`
PPMQHCONN	`typedef PMQHCONN MQPOINTER PPMQHCONN;`
PPMQHOBJ	`typedef PMQHOBJ MQPOINTER PPMQHOBJ;`
PPMQLONG	`typedef PMQLONG MQPOINTER PPMQLONG;`

Datový typ	Zastupování
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 najdete v tématu “Datové typy” na stránce 241 .

Deklarace COBOL

Datový typ	Zastupování
MQBOOL	`PIC S9(9) BINARY`
MQBYTE	`PIC X`
MQBYTE8	`PIC X(8)`
MQBYTE16	`PIC X(16)`
MQBYTE24	`PIC X(24)`
MQBYTE32	`PIC X(32)`
MQBYTE40	`PIC X(40)`
MQCHAR	`PIC X`
MQCHAR4	`PIC X(4)`
MQCHAR8	`PIC X(8)`
MQCHAR12	`PIC X(12)`
MQCHAR20	`PIC X(20)`
MQCHAR28	`PIC X(28)`

Datový typ	Zastupování
MQCHAR32	`PIC X(32)`
MQCHAR48	`PIC X(48)`
MQCHAR64	`PIC X(64)`
MQCHAR128	`PIC X(128)`
MQCHAR256	`PIC X(256)`
MQFLOAT32	`USAGE COMP-1`
MQFLOAT64	`USAGE COMP-2`
MQHCONN	`PIC S9(9) BINARY`
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.

Datový typ	Zastupování
MQBOOL	`fixed bin(31)`

Datový typ	Zastupování
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)`
MQCHAR128	`char(128)`
MQCHAR256	`char(256)`
MQFLOAT32	`binary float(21) ieee`
MQFLOAT64	`binary float(52) ieee`
MQHCONN	`fixed bin(31)`
MQOBJ	`fixed bin(31)`

Datový typ	Zastupování
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
Produkt System/390 assembler je podporován pouze v systému z/OS .

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`

Datový typ	Zastupování
MQCHAR12	`DS CL12`
MQCHAR20	`DS CL20`
MQCHAR28	`DS CL28`
MQCHAR32	`DS CL32`
MQCHAR48	`DS CL48`
MQCHAR64	`DS CL64`
MQCHAR128	`DS CL128`
MQCHAR256	`DS CL256`
MQFLOAT32	`DS EB`
MQFLOAT64	`DS DB`
MQHCONN	`DS F`
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-úvod

Tento oddíl uvádí datové typy struktury použité v rozhraní MQI. Samotné datové typy struktury jsou popsány v následujících sekcích.

Souhrn

V následujících tabulkách jsou shrnuty datové typy struktury použité v rozhraní MQI.

Tabulka 467. Datové typy struktury použité při voláních MQI (nebo výstupních funkcích):

Struktura	**Popis**	**Volání kde se používá**
MQACHCITY	Záhlaví řetězce uživatelských procedur rozhraní API	
MQAIR	Záznam ověřovacích informací	MQCONNX
MQAXC	Kontext ukončení rozhraní API	
MQAXON	Výstupní parametr rozhraní API	
MQBMHO	Volby zpracování vyrovnávací paměti pro zprávy	MQBUFMH5
Objekt MQBO	Volby začátku	MQBEGIN
Funkce MQCBD	Deskriptor zpětného volání	MQCB
OBJEKT MQCBO	Volby vytvoření balíku	Balík mqCreate
MQCHARV	Řetězec proměnné délky	MQINQMP
MQCNO	Volby připojení	MQCONNX
MQCP	Parametry zabezpečení	MQCONNX
MQCTLO.	Volby zpětného dotazu	MQCTL
MQDMPO	Volby odstranění vlastností zprávy	MQDLTMP
MQGMO	Volby získání zprávy	MQGET
Objekt MQIMPO	Zjistit volby vlastností zprávy	MQINQMP
MQMD	deskriptor zprávy	MQBUFMH, MQMHBUF, MQCB, MQGET, MQPUT, MQPUT1
MQMHBO	Volby zpracování zpráv do vyrovnávací paměti	MQMHBUF
MQOD	deskriptor objektu	MQOPEN, MQPUT1
Objekt MQOR	Záznam objektu	MQOPEN, MQPUT1
MQPD	Deskriptor vlastnosti	MQSETMP
MQPMO	Volby vložení zprávy	MQPUT, MQPUT1
MQPMR	záznam vložení zprávy	MQPUT, MQPUT1
Objekt MQRR	Záznam odpovědi	MQOPEN, MQPUT, MQPUT1
Aplikace MQSCO	Volby konfigurace SSL	MQCONNX
MQSD.	Deskriptor odběru	MQSUB
MQSMPO	Nastavit volbu vlastnosti zprávy	MQSETMP

Tabulka 467. Datové typy struktury použité při voláních MQI (nebo výstupních funkcích): (pokračování)		
Struktura	**Popis**	**Volání kde se používá**
MQSRO	Volby požadavku na odběr	MQSUBRQ.
MQSTS	Struktura vykazování stavu	Příkaz MQSTAT

Tabulka 468. Datové typy struktury použité v datech zprávy:	
Struktura	**Popis**
MQCIH.	Záhlaví informací CICS
MQCFH	Záhlaví PCF
MQEPR	Vložené záhlaví PCF
Objekt MQDH	Záhlaví distribuce
MQDLH	Záhlaví nedoručených zpráv (nedoručené zprávy)
MQIIH.	Záhlaví informací IMS
MQMDE	Rozšíření deskriptoru zpráv
MQRFH.	Pravidla a formátovací záhlaví
MQRFH2	Pravidla a formátování záhlaví 2
MQRMH	Záhlaví referenční zprávy
MQTM, MQ	zpráva spouštěče
MQTMC2	Zpráva spouštěče (znakový formát 2)
MQWIHKM	Záhlaví informací o práci
MQXQH	Hlavička přenosové fronty

Poznámka: Struktura MQDXP (parametr uživatelské procedury konverze dat) je popsána v části "Uživatelská procedura konverze dat" na stránce 855spolu s přidruženými voláními pro převod dat.

Pravidla pro datové typy struktury

Programovací jazyky se liší ve své úrovni podpory struktur a jsou přijata určitá pravidla a konvence pro konzistentně mapování struktur MQI v jednotlivých programovacích jazycích:

1. Struktury musí být zarovnány podle jejich přirozených hranic.
   - Většina struktur MQI vyžaduje zarovnání 4 bajtů.
   - V systému IBM istruktury obsahující ukazatele vyžadují 16bajtové zarovnání; to jsou: 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 zarovnány na 4bajtové hranice.
   - Pole s datovými typy, která se rovná MQPTR, musí být zarovnána s 16bajtovými hrami na IBM ia 4bajtovými hranicemi v jiných prostředích.
   - Ostatní pole jsou zarovnána na 1bajtové hranice.
3. Délka struktury musí být násobkem jeho hranice zarovnání.
   - Většina struktur MQI má délky, které jsou násobky 4 bajtů.
   - Na IBM istruktur obsahující ukazatele mají délky, které jsou násobky 16 bajtů.

4. Je-li to nutné, musí být přidány vyplňující bajty nebo pole, aby bylo zaručeno dodržení výše uvedených pravidel.

Konvence použité v popisech

Popis jednotlivých datových typů struktury zahrnuje:

- Přehled účelu a použití struktury
- Popisy polí ve struktuře, ve formě, která je nezávislá na programovacím jazyce
- Příklady toho, jak je struktura deklarována v každém z podporovaných programovacích jazyků

Popis jednotlivých datových typů struktury obsahuje následující sekce:

Název struktury
Název struktury, za nímž následuje souhrn polí ve struktuře.

Přehled
Stručný popis účelu a použití struktury.

Pole
Popisy polí. Pro každé pole je za názvem pole následován jeho elementární datový typ v závorkách (). V textu jsou názvy polí zobrazeny pomocí kurzívy typu; například *Version*.

K dispozici je také popis účelu pole spolu se seznamem hodnot, které může pole provést. Názvy konstant se zobrazují velkými písmeny; například MQGMO_STRUC_ID. Soubor konstant se stejnou předponou je zobrazen pomocí znaku *, například: MQIA_ *.

V popisech polí se používají následující termíny:

Vstup
Zadejte informace do pole, když vytvoříte hovor.

výstup
Správce front vrátí informace v poli po dokončení nebo selhání volání.

Vstup a výstup
Když zavoláte, dodáte informace do pole a správce front změní informace, když se volání dokončí nebo selže.

Počáteční hodnoty
Tabulka zobrazující počáteční hodnoty pro každé pole v souborech definic dat dodaných s rozhraním MQI.

Deklarace C
Typické prohlášení o struktuře v C.

Deklarace COBOL
Typické deklarace struktury v jazyce COBOL.

Deklarace PL/I
Typické prohlášení o struktuře v PL/I.

Deklarace asembleru System/390
Typické deklarace struktury v jazyku System/390 assembler.

Deklarace jazyka Visual Basic
Typické prohlášení o konstrukci ve Visual Basicu.

Programování v C

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

Soubory záhlaví

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

Tyto soubory záhlaví jsou shrnuty v části Tabulka 469 na stránce 241.

Tabulka 469. Soubory záhlaví C	
Soubor	**Obsah**
CMQC	Prototypy funkcí, datové typy a pojmenované konstanty pro hlavní rozhraní MQI
CMQXC	Prototypy funkce, datové typy a pojmenované konstanty pro ukončení konverze dat
CMQEC	Prototypy funkcí, datové typy a pojmenované konstanty pro hlavní strukturu rozhraní MQI, uživatelské procedury pro převod dat a vstupní body rozhraní (CMQEC obsahuje CMQXC a CMQC.)

Chcete-li zlepšit přenositelnost aplikací, uveďte název souboru záhlaví malými písmeny na direktivě preprocesoru `#include` :

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

Funkce
Nemusíte uvádět všechny parametry, které jsou předávány pomocí adresy pokaždé, když vyvoláváte funkci.

- Předat parametry, které jsou *pouze vstup* a typu MQHCONN, MQBOBJ nebo MQLONG podle hodnoty.
- Předání všech ostatních parametrů podle adresy.

Není-li požadován konkrétní parametr, použijte jako parametr pro vyvolání funkce ukazatel null jako místo adresy dat parametru. Parametry, pro které je to možné, jsou identifikovány v popisech volání.

Jako hodnotu funkce se nevrací žádný parametr; v terminologii C to znamená, že všechny funkce vrací `void`.

Atributy funkce jsou definovány proměnnou makra MQENTRY; hodnota této proměnné makra závisí na daném prostředí.

Parametry s nedefinovaným datovým typem

Parametr *`Buffer`* na funkcích MQGET, MQPUT a MQPUT1 má nedefinovaný datový typ. Tento parametr se používá k odesílání a přijímání 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. Deklarujte skutečný parametr funkce jako ukazatel na neobsazený a uveďte adresu libovolného druhu dat jako parametru ve vyvolání funkce.

Datové typy

Definujte všechny datové typy pomocí příkazu jazyka C `typedef` . Pro každý datový typ také definujte odpovídající datový typ ukazatele. Název datového typu ukazatele je název elementárních nebo strukturních datových typů s předponou písmenem P, která označuje ukazatel. Definujte atributy ukazatele pomocí proměnné makra MQPOPOINTER; hodnota této proměnné makra závisí na daném prostředí. Následující příklad ukazuje, 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, porovnejte nebo nastavíte 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 tyto příkazy nefungují správně pro data deklarovaná s datovými typy MQBYTEn.

Manipulace se znakovými řetězci
Když správce front vrátí data znaků do aplikace, správce front vždy vycpá znaková data mezerami do definované délky pole; správce front *nevrací* řetězce ukončené hodnotou null.

Proto při kopírování, porovnání nebo zřetězení takových řetězců použijte řetězcové funkce `strncpy`, `strncmp`nebo `strncat`.

Nepoužívejte funkce řetězce, které vyžadují, aby byl řetězec ukončený znakem 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 poskytnutí počátečních hodnot pro struktury MQ při deklarování 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 vyskytují v MQMD), rozhraní MQI definuje určité platné hodnoty. Pro každou z platných hodnot jsou k dispozici *dvě* proměnné makra:

- Jedna makroproměnná definuje hodnotu jako řetězec s délkou, kromě implikovaných shodných s hodnotou null, přesně definované délky pole. Například pro pole *Format* v produktu MQMD je poskytnuta následující proměnná makra (¬ představuje prázdný znak):

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

  Použijte tento tvar s funkcemi `memcpy` a `memcmp` .
- Další proměnná makra definuje hodnotu jako pole znaků; název této proměnné makra je název řetězce tvořená řetězcem s příponou _ARRAY. Příklad:

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

  Tento formulář použijte k inicializaci pole, když deklarujete instanci struktury s hodnotami odlišnými od proměnné poskytnuté proměnnou makra MQMD_DEFAULT. (To není vždy nezbytné; v některých prostředích můžete použít řetězcovou formu hodnoty v obou situacích. Můžete však použít formulář pole pro deklarace, protože to je nezbytné z důvodu kompatibility s programovacím jazykem C + +.)

Počáteční hodnoty pro dynamické struktury

Když se požaduje proměnný počet instancí struktury, jsou instance obvykle vytvářeny v hlavní paměti získávané dynamicky pomocí funkcí `calloc` nebo `malloc` . Chcete-li inicializovat pole v těchto strukturách, zvažte následující postup:

1. Deklarujte instanci struktury pomocí příslušné makro proměnné MQxxx_DEFAULT k inicializaci struktury. Tato instance se stane modelem pro další instance:

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

   Klíčová slova `static` nebo `auto` mohou být kódována podle deklarace, aby byla instance modelu vytvořena statická nebo dynamická doba životnosti, jak je požadováno.
2. Použijte funkce `calloc` nebo `malloc` k získání paměti pro dynamickou instanci struktury:

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

3. Použijte funkci `memcpy` ke zkopírování instance modelu do dynamické instance:

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

Použít z C++

V případě programovacího jazyka C + + obsahují hlavičkové soubory následující další příkazy, které jsou obsaženy pouze v případě, že používáte kompilátor jazyka C + +:

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

/* rest of header file */

#ifdef __cplusplus
  }
#endif
```

Notační konvence
Tyto informace ukazují, jak vyvolat funkce a deklarovat parametry.

V některých případech jsou parametry pole s velikostí, která není pevná. Pro tyto hodnoty je malá písmena n použita ke znázornění číselné konstanty. Když kódíte deklaraci pro tento parametr, nahraďte hodnotu n numerickou hodnotou, která je povinná.

Programování COBOL

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

soubory COPY
K dispozici jsou různé soubory COPY, které vám pomohou při psaní aplikačních programů COBOL, které používají rozhraní MQI. Jsou zde dva soubory obsahující pojmenované konstanty a dva soubory pro každou ze struktur.

Každá struktura je poskytována ve dvou podobách: formulář s počátečními hodnotami a formulář bez:

- Použijte struktury s počátečními hodnotami ve WORKING-STORAGE SECTION programu COBOL; jsou obsaženy v souborech COPY s příponami s příponou V (pro hodnoty).
- Použijte struktury bez počátečních hodnot v `LINKAGE SECTION` programu COBOL; jsou obsaženy v souborech COPY s příponami s příponou L (pro Linkage).

Soubory COPY jsou shrnuty v části Tabulka 470 na stránce 244. Ne všechny uvedené soubory jsou dostupné ve všech prostředích.

Tabulka 470. Soubory COBOL COPY		
Soubor (s počátečními hodnotami)	**Soubor (bez počátečních hodnot)**	**Obsah**
CMQIRV	CMQIRL	Záznam ověřovacích informací
CMQBOV	CMQBOL	Začátek struktury voleb
CMQCIHV	CMQCIHL	Struktura záhlaví informací CICS
CMQCNOV	CMQCNOL	Struktura voleb připojení
CMQDHV	CMQDHL	Struktura záhlaví distribuce
CMQDLHV	CMQDLHL	Struktura záhlaví nedoručených zpráv
CMQDXPV	CMQDXPL	Struktura výstupního parametru konverze dat
CMQGMOV	CMQGMOL	Získat strukturu voleb zprávy
CMQIIHV	CMQIIHL	Struktura informačního záhlaví IMS
CMQMDV	CMQMDL	Struktura deskriptoru zpráv
CMQMDEV	CMQMDEL.	Struktura rozšíření deskriptoru zpráv
CMQMD1V	CMQMD1L	Struktura deskriptoru zpráv verze 1
CMQODOV	CMQODUL.	Struktura deskriptoru objektu
CMQORV	CMQORL	Struktura záznamu objektu
CMQPMOV	CMQPMOL	Vložit strukturu voleb zprávy
CMQRFHV.	CMQRFHL	Pravidla a formátování struktury záhlaví
CMQRFH2V	CMQRFH2L	Pravidla a formátování struktury záhlaví verze 2
CMQRMHV	CMQRMHL	Struktura záhlaví referenční zprávy
CMQRRV	CMQRRL	Struktura záznamu odezvy
CMQSCOROVER	CMQSCOL	Volby konfigurace SSL
CMQTMV.	CMQTML	Struktura zprávy spouštěče
CMQTMCV	CMQTMCL	Struktura zprávy spouštěče (znakový formát)
CMQTMC2V	CMQTMC2L	Struktura zprávy spouštěče (znakový formát) verze 2
CMQWIHV	CMQWIHL	Struktura záhlaví pracovních informací
CMQXQHV	CMQXQHLL	Struktura záhlaví přenosové fronty
CMQV	-	Pojmenované konstanty pro hlavní rozhraní MQI
CMQXV	-	Pojmenované konstanty pro ukončení konverze dat
CMQMD2V	CMQMD2L	Struktura deskriptoru zpráv verze 2

Struktury

V souboru COPY začíná každá deklarace struktury s položkou level-10 ; to vám umožňuje deklarovat několik instancí struktury zakódováním deklarace level-01 a poté použitím příkazu COPY ke kopírování ve zbytku deklarace struktury. Chcete-li se odkázat na odpovídající instanci, použijte klíčové slovo IN :

```
* Declare two instances of MQMD
 01  MY-MQMD.
```

```
     COPY CMQMDV.
 01  MY-OTHER-MQMD.
     COPY CMQMDV.
*
* Set MSGTYPE field in MY-OTHER-MQMD
     MOVE MQMT-REQUEST TO MQMD-MSGTYPE IN MY-OTHER-MQMD.
```

Srovnejte struktury podle příslušných hranic. Použijete-li příkaz COPY k zahrnutí struktury za položkou, která není položkou level-01 , ujistěte se, že struktura začíná na příslušném offsetu od začátku položky level-01 . Většina struktur MQI vyžaduje čtyřbajtové zarovnání; výjimky z nich jsou MQCNO, MQOD a MQPMO, které vyžadují 16bajtové zarovnání v systému IBM i.

V tomto oddílu jsou názvy polí ve strukturách zobrazeny bez předpony. V COBOLu, názvy polí mají předponu s názvem struktury, za nímž následuje pomlčka. Avšak pokud název struktury končí numerickou číslicí, což označuje, že struktura je druhou nebo novější verzí původní struktury, numerická číslice se vynechá z předpony. Názvy polí v jazyce COBOL jsou zobrazeny velkými písmeny (je-li to nutné, lze použít malá nebo velká písmena.) Například pole *MsgType* popsané v “MQMD-deskriptor zprávy” na stránce 383 se stane MQMD-MSGTYPE v COBOLu.

Struktury V-suffix jsou deklarovány s počátečními hodnotami pro všechna pole; musíte nastavit pouze ta pole, ve kterých chcete, aby hodnota, která se liší od dodané počáteční hodnoty.

Ukazatele
Některé struktury musí adresovat volitelná data, která mohou být nesousedící se strukturou, jako např. záznamy MQOR a MQRR, které jsou adresovány strukturou MQOD.

Chcete-li adresovat tato volitelná data, struktury obsahují pole, která jsou deklarována s datovým typem ukazatele. Avšak COBOL nepodporuje datový typ ukazatele ve všech prostředích. Z tohoto důvodu mohou být volitelná data adresována také pomocí polí, která obsahují posunutí dat od začátku struktury.

Chcete-li portportovat aplikaci mezi prostředími, zjistěte, zda je datový typ ukazatele dostupný ve všech zamýšlených prostředích. Pokud tomu tak není, musí aplikace adresovat volitelná data pomocí polí offsetu místo polí ukazatele.

V těchto prostředích, kde ukazatele nejsou podporovány, deklarujte pole ukazatele jako bajtové řetězce odpovídající délky, přičemž počáteční hodnota je celobajtová bajtová řetězec. Neměňte tuto počáteční hodnotu, pokud používáte pole offsetu.

Pojmenované konstanty

V tomto oddílu jsou zobrazeny názvy konstant, které obsahují znak podtržítka (_) jako součást názvu. V COBOLu, použijte znak pomlčky (-) místo podtržítka.

Konstanty, které mají znakové řetězce, používají jednoduchou uvozovku jako oddělovač řetězců (').
V některých prostředích může být nutné zadat vhodnou volbu kompilátoru, aby kompilátor přijímal jednoduché uvozovky jako oddělovač řetězců v místě dvojitých uvozovek.

Pojmenované konstanty jsou deklarovány v souborech COPY jako položky level-10 . Chcete-li použít konstanty, deklarujte explicitně položku level-01 a pak použijte příkaz COPY ke kopírování v deklaracích konstant:

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

Předchozí metoda způsobí, že se konstanty zabírají v programu i v případě, že na ně nejsou odkazy. Pokud zahrnete konstanty do mnoha samostatných programů v rámci stejné jednotky spuštění, existuje více kopií konstant, které zbytečně zabírají hlavní paměť. Tento efekt se vyvarujte použitím jedné z následujících technik:

- Přidejte klauzuli GLOBAL do deklarace level-01 :

```
* Declare a global structure to hold the constants
```

```
01  MY-MQ-CONSTANTS GLOBAL.
    COPY CMQV.
```

Tato příčina alokuje paměť pouze pro jednu sadu konstant v rámci jednotky běhu. Tyto konstanty však mohou být odkazovány jakýmkoliv programem v rámci jednotky spuštění, a to nejen programem, který obsahuje deklaraci level-01 .

Poznámka: Klauzule GLOBAL není podporována ve všech prostředích.

- Manuálně zkopírujte do každého programu pouze ty konstanty, na které tento program odkazuje. Nepoužívejte příkaz COPY ke zkopírování všech konstant do programu.

Notační konvence

Poslední uvedená témata v této sekci ukazují, jak vyvolat volání a deklarovat parametry. V některých případech jsou parametry tabulky nebo znakové řetězce, jejichž velikost není pevná. Pro tyto hodnoty je malá písmena n použita ke znázornění číselné konstanty. Když kódíte deklaraci pro tento parametr, nahraďte hodnotu n numerickou hodnotou, která je povinná.

Programování assembleru System/390

Tento oddíl obsahuje informace, které vám pomohou při použití rozhraní MQI z programovacího jazyka System/390 Assembler.

Makra

K dispozici jsou různá makra, která vám pomohou s napsáním aplikačních programů v asembleru, které používají rozhraní MQI.

Pro pojmenované konstanty existují dvě makra a jedno makro pro každou ze struktur. Tyto soubory jsou shrnuty v Tabulka 471 na stránce 246.

Tabulka 471. Makra programu pro

Soubor	**Obsah**
CMQA	Pojmenované konstanty (equates) pro hlavní rozhraní MQI
CMQCIHA	Struktura záhlaví informací CICS
CMQCNOA	Struktura voleb připojení
CMQDLHA	Struktura záhlaví nedoručených zpráv
CMQDXPA	Struktura výstupního parametru konverze dat
CMQGMOA	Získat strukturu voleb zprávy
CMQIIHA	Struktura informačního záhlaví IMS
CMQMDA	Struktura deskriptoru zpráv
CMQMDEA	Struktura rozšíření deskriptoru zpráv
CMQODA	Struktura deskriptoru objektu
CMQPMOA	Vložit strukturu voleb zprávy
CMQRFHA	Pravidla a formátování struktury záhlaví
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

Tabulka 471. Makra programu pro (pokračování)	
Soubor	**Obsah**
CMQWIHA	Struktura záhlaví pracovních informací
CMQXA	Pojmenované konstanty pro ukončení konverze dat
CMQXPA	Struktura výstupního parametru rozhraní API překračování
CMQXQHA	Struktura záhlaví přenosové fronty

Struktury
Struktury jsou generovány makry, které mají různé parametry k řízení akce makra. Tyto parametry jsou popsány v následujících sekcích.

Čas od času se zavádí nové verze struktur produktu MQ . Další pole v nové verzi mohou způsobit, že struktura, která byla dříve menší než 256 bajtů, by měla být větší než 256 bajtů. Z tohoto důvodu jsou pokyny pro asembler, které jsou určeny ke kopírování struktury MQ , nebo k nastavení struktury MQ na hodnoty null, aby správně pracovaly se strukturami, které mohou být větší než 256 bajtů. Případně můžete použít makro DCLVER nebo makro CMQVERA s parametrem VERSION k deklaraci specifické verze struktury.

Zadání názvu struktury
Chcete-li deklarovat více než jednu instanci struktury, bude makro pojmenovává název každého pole ve struktuře s řetězcem specifikovatelným uživatelem a podtržítkem.

Použitý řetězec je jmenovka zadaná při vyvolání makra. Není-li zadán žádný popisek, použije se k sestavení 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, které jsou řízeny parametrem DSECT :

DCET=ANO
Assembler DSECT instruction is used to start a new data section; the definition definition immediately follows the DSECT statement. Návěští pro vyvolání makra se použije jako název sekce dat; není-li zadán žádný popisek, použije se název struktury.

DSECT=NE
Pokyny pro asembler DC se používají k definování struktury na aktuální pozici v rutině. Pole jsou inicializovaná s hodnotami, které lze zadat zakódováním příslušných parametrů ve vyvolání makra. Pole, pro které nejsou zadány žádné hodnoty při vyvolání makra, jsou inicializovány výchozími hodnotami.

Uvedená hodnota musí být velká. Není-li parametr DSECT zadán, předpokládá se hodnota 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 pole *Version* a neřídí verzi struktury, která byla ve skutečnosti deklarována. Chcete-li určit verzi struktury deklarované struktury, použijte parametr DCLVER :

DCLVER=AKTUÁLNÍ
Deklarovaná verze je aktuální (nejnovější) verze.

DCLVER=URČENÝ
Deklarovaná verze je verzí určená parametrem VERSION . Pokud vynecháte parametr VERSION , výchozí je verze 1.

Pokud zadáte parametr VERSION , hodnota musí být samostatnou definující číselnou konstantu nebo pojmenovanou konstantu pro požadovanou verzi (například MQCNO_VERSION_3). Pokud zadáte nějakou jinou hodnotu, bude struktura deklarována tak, jako by byla zadána hodnota DCLVER=CURRENT , i když se hodnota parametru VERSION vyřeší na platnou hodnotu.

Uvedená hodnota musí být velká. Vynecháte-li parametr DCLVER , použije se hodnota převzata z globální proměnné makra MQDCLVER . Tuto proměnnou lze nastavit pomocí makra CMQVERA.

Deklarování jedné struktury vložené do jiné

Chcete-li deklarovat jednu strukturu jako komponentu jiné struktury, použijte parametr NESTED :

NEST=ANO
Deklarace struktury je vnořena do jiné.

NESTED=NE
Deklarace struktury není vnořena do jiné.

Uvedená hodnota musí být velká. Pokud vynecháte parametr NESTED , předpokládá se parametr NESTED=NO .

Určení počátečních hodnot pro pole
Určete hodnotu, která má být použita k inicializaci pole ve struktuře kódováním názvu tohoto pole (bez předpony) jako parametru v rámci vyvolání makra spolu s požadovanou hodnotou.

Chcete-li například deklarovat strukturu deskriptoru zpráv s polem *MsgType* inicializovaným s parametrem MQMT_REQUEST, a pole *ReplyToQ* inicializováno s řetězcem "MY_REPLY_TO_QUEUE", použijte následující:

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

Uvedete-li pojmenovanou konstantu (equate) jako hodnotu při vyvolání makra, použijte makro CMQA, abyste definovali pojmenovanou konstantu. Neuzavírejte hodnoty znakových řetězců do jednoduchých uvozovek.

Řízení výpisu

Řídit vzhled deklarace struktury v seznamu asembleru pomocí parametru LIST :

LIST=ANO
Deklarace struktury se zobrazí ve výpisu assembleru.

SEZNAM=NE
Deklarace struktury se neobjevuje ve výpisu assembleru.

Uvedená hodnota musí být velká. Pokud vynecháte parametr LIST , předpokládá se parametr LIST=NO .

Makro CMQVERA

Toto makro vám umožňuje nastavit výchozí hodnotu, která má být použita pro parametr DCLVER v makrech struktury. Hodnota uvedená CMQVERA je použita makrem struktury pouze, pokud vynecháte parametr DCLVER z vyvolání struktury makra. Výchozí hodnota je nastavena kódováním makra CMQVERA s parametrem DCLVER :

DCLVER=AKTUÁLNÍ
Výchozí verze je nastavena na aktuální (nejnovější) verzi.

DCLVER=URČENÝ
Výchozí verze je nastavena na verzi zadanou parametrem VERSION .

Musíte uvést parametr DCLVER a hodnota musí být velká písmena. Hodnota nastavená CMQVERA zůstává výchozí hodnotou až do dalšího vyvolání CMQVERA nebo na konci sestavy. Vynecháte-li CMQVERA, je výchozí hodnota DCLVER=CURRENT.

Notační konvence

Pozdější sekce ukazují, jak vyvolat volání a deklarovat parametry. V některých případech jsou parametry pole nebo znakové řetězce s velikostí, která není pevná, malá písmena n se používají ke znázornění číselné konstanty. Když kódíte deklaraci pro tento parametr, nahraďte hodnotu n numerickou hodnotou, která je povinná.

MQAIR-záznam ověřovacích informací

Struktura MQAIR představuje záznam ověřovacích informací.

Následující tabulka shrnuje pole ve struktuře.

Tabulka 472. Pole v aplikaci MQAIR		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Verze	Číslo verze struktury	verze
AuthInfoType	Typ ověřovacích informací	AuthInfoType
AuthInfoConnName	Název připojení k serveru LDAP CRL	AuthInfoConnName
LDAPUserNamePtr	Adresa jména uživatele LDAP	LDAPUserNamePtr
Posunutí LDAPUserName	Ofset jména uživatele LDAP od začátku MQSCO	PosunutíLDAPUserName
Délka LDAPUserName	Délka jména uživatele LDAP	LDAPUserNameDélka
LDAPPassword	Heslo pro přístup k serveru LDAP	LDAPPassword
Poznámka: Zbývající pole se budou ignorovat, pokud je *Verze* menší než MQAIR_VERSION_2.		
OCSPResponderURL	Adresa URL, na které lze kontaktovat odpovídací modul OCSP	OCSPResponderURL

Přehled pro MQAIR

Struktura MQAIR umožňuje aplikaci spuštěnou jako klient produktu WebSphere MQ MQI k určení informací o ověřovateli, který má být použit pro připojení klienta. Struktura je vstupním parametrem volání MQCONNX.

Dostupnost: AIX, HP-UX, Solaris, Linux and Okna clients.

Znaková sada a kódování: Data v aplikaci MQAIR musí být ve znakové sadě a kódování lokálního správce front; tyto údaje jsou dány atributem správce front **CodedCharSetId** a MQENC_NATIVE.

Pole pro MQAIR

Struktura MQAIR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

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.

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.

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.

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 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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQAIR

Tabulka 473. Počáteční hodnoty polí v aplikaci MQAIR

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQIR_CONSTRUCT	'AIR¬'
Verze	MQAIR_VERSION_1	1
AuthInfoType	MQAIT_CRL_LDAP	1
AuthInfoConnName	Není	Nulový řetězec nebo prázdné znaky
LDAPUserNamePtr	Není	Nulový ukazatel nebo bajty null
Posunutí LDAPUserName	Není	0
Délka LDAPUserName	Není	0
LDAPPassword	Není	Nulový řetězec nebo prázdné znaky
OCSPResponderURL	Není	Nulový řetězec nebo prázdné znaky

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQAIR_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

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

Deklarace C

```
typedef struct tagMQAIR MQAIR;
struct tagMQAIR {
  MQCHAR4    StrucId;             /* Structure identifier */
  MQLONG     Version;             /* Structure version number */
  MQLONG     AuthInfoType;        /* Type of authentication
                                     information */
  MQCHAR264  AuthInfoConnName;    /* Connection name of CRL LDAP
                                     server */
  PMQCHAR    LDAPUserNamePtr;     /* Address of LDAP user name */
  MQLONG     LDAPUserNameOffset;  /* Offset of LDAP user name from start
                                     of MQAIR structure */
  MQLONG     LDAPUserNameLength;  /* Length of LDAP user name */
  MQCHAR32   LDAPPassword;        /* Password to access LDAP server */
  MQCHAR256  OCSPResponderURL;    /* URL of OCSP responder */

};
```

Deklarace COBOL

```
**   MQAIR structure
  10 MQAIR.
**    Structure identifier
```

```
   15 MQAIR-STRUCID            PIC X(4).
**    Structure version number
   15 MQAIR-VERSION            PIC S9(9) BINARY.
**    Type of authentication information
   15 MQAIR-AUTHINFOTYPE       PIC S9(9) BINARY.
**    Connection name of CRL LDAP server
   15 MQAIR-AUTHINFOCONNNAME   PIC X(264).
**    Address of LDAP user name
   15 MQAIR-LDAPUSERNAMEPTR    POINTER.
**    Offset of LDAP user name from start of MQAIR structure
   15 MQAIR-LDAPUSERNAMEOFFSET PIC S9(9) BINARY.
**    Length of LDAP user name
   15 MQAIR-LDAPUSERNAMELENGTH PIC S9(9) BINARY.
**    Password to access LDAP server
   15 MQAIR-LDAPPASSWORD       PIC X(32).
**    URL of OCSP responder
   15 MQAIR-OCSPRESPONDERURL   PIC X(256).
```

Deklarace jazyka Visual Basic

```
Type MQAIR
  StrucId            As String*4   'Structure identifier'
  Version            As Long       'Structure version number'
  AuthInfoType       As Long       'Type of authentication information'
  AuthInfoConnName   As String*264 'Connection name of CRL LDAP server'
  LDAPUserNamePtr    As MQPTR      'Address of LDAP user name'
  LDAPUserNameOffset As Long       'Offset of LDAP user name from start'
                                   'of MQAIR structure'
  LDAPUserNameLength As Long       'Length of LDAP user name'
  LDAPPassword       As String*32  'Password to access LDAP server'
End Type
```

MQBMHO-Vyrovnávací paměť pro volby obsluhy zprávy

Následující tabulka shrnuje pole ve struktuře. MQBMHO struktury-vyrovnávací paměť pro volby zpracování zpráv

Tabulka 474. Pole v MQBMHO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby řízení akce MQBMHO	Volby

Přehled pro MQBMHO

Dostupnost: Vše. Struktura obslužného programu vyrovnávací paměti pro zpracování zpráv-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 (MQENC_NATIVE).

Pole pro MQBMHO

Struktura voleb popisovače zpráv pro zpracování zpráv-pole

Struktura MQBMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQBMHO
Struktura obsluhy vyrovnávací paměti pro zpracování zpráv-počáteční hodnoty

Tabulka 475. Počáteční hodnoty polí v MQBMHO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQBMHO_STRUCTION_ID	'BMHO'
Version	MQBMHO_VERSION_1	1
Options	MQBMHO_NONE	0

Notes:

1. V programovacím jazyce C-proměnná makraHodnota MQBMHO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQBMHO MyBMHO = {MQBMHO_DEFAULT};
```

Deklarace C
Struktura obsluhy vyrovnávací paměti pro strukturu zprávy-deklarace jazyka C

```
typedef struct tagMQBMHO MQBMHO;
struct tagMQBMHO {
  MQCHAR4  StrucId;        /* Structure identifier */
```

```
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQBUFMH */
};
```

Deklarace COBOL
Struktura popisovače zpráv do vyrovnávací paměti-deklarace jazyka COBOL

```
**   MQBMHO structure
  10 MQBMHO.
**    Structure identifier
   15 MQBMHO-STRUCID               PIC X(4).
**    Structure version number
   15 MQBMHO-VERSION               PIC S9(9) BINARY.
**    Options that control the action of MQBUFMH
   15 MQBMHO-OPTIONS               PIC S9(9) BINARY.
```

Deklarace PL/I
Struktura obsluhy vyrovnávací paměti pro strukturu zpráv-deklarace jazyka PL/I

```
Dcl
 1 MQBMHO based,
   3 StrucId        char(4),       /* Structure identifier */
   3 Version        fixed bin(31), /* Structure version number */
   3 Options        fixed bin(31), /* Options that control the action
                                      of MQBUFMH */
```

Deklarace High Level Assembler
Struktura manipulátoru vyrovnávací paměti pro zprávy-deklarace jazyka assembler

```
MQBMHO                      DSECT
MQBMHO_STRUCID              DS   CL4  Structure identifier
MQBMHO_VERSION              DS   F    Structure version number
MQBMHO_OPTIONS              DS   F    Options that control the
*                                     action of MQBUFMH
MQBMHO_LENGTH               EQU  *-MQBMHO
MQBMHO_AREA                 DS   CL(MQBMHO_LENGTH)
```

MQBO-Začátek voleb

Následující tabulka shrnuje pole ve struktuře.

Tabulka 476. Pole v MQBO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby, které řídí akci MQBEGIN	Volby

Přehled pro MQBO

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows; není k dispozici pro klienty WebSphere MQ MQI.

Účel: Struktura MQBO umožňuje aplikaci určit volby související s vytvořením jednotky práce. Struktura je vstupním/výstupním parametrem pro volání MQBEGIN.

Znaková sada a kódování: Data ve struktuře MQBO musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front uvedeného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro objekt MQBO

Struktura MQBO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Volby (MQLONG)
Toto pole je vždy vstupním polem. Jeho počáteční hodnota je MQBO_NONE.

Hodnota musí být:

MQBO_NONE
Nejsou uvedeny žádné volby.

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.

Počáteční hodnoty a deklarace jazyka pro MQBO

Tabulka 477. Počáteční hodnoty polí v MQBO pro MQBO

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY OBJEKTU MQBO_STRUCT	'BO¬¬'
Version	MQBO_VERSION_1	1
Options	MQBO_NONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraObjekt MQBO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQBO MyBO = {MQBO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQBO MQBO;
struct tagMQBO {
  MQCHAR4  StrucId;  /* Structure identifier */
  MQLONG   Version;  /* Structure version number */
  MQLONG   Options;  /* Options that control the action of MQBEGIN */
};
```

Deklarace COBOL

```
**   MQBO structure
  10 MQBO.
**    Structure identifier
   15 MQBO-STRUCID PIC X(4).
**    Structure version number
   15 MQBO-VERSION PIC S9(9) BINARY.
**    Options that control the action of MQBEGIN
   15 MQBO-OPTIONS PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQBO based,
   3 StrucId char(4),       /* Structure identifier */
   3 Version fixed bin(31), /* Structure version number */
   3 Options fixed bin(31); /* Options that control the action of
                               MQBEGIN */
```

Deklarace jazyka Visual Basic

```
Type MQBO
  StrucId As String*4 'Structure identifier'
  Version As Long     'Structure version number'
  Options As Long     'Options that control the action of MQBEGIN'
End Type
```

MQCBC-Kontext zpětného volání

Následující tabulka shrnuje pole ve struktuře. Struktura popisující rutinu zpětného volání.

Tabulka 478. Pole v MQCBC		
Pole	**Popis**	**Téma**
StrucID	Identifikátor struktury	StrucID
Version	Číslo verze struktury	verze
CallType	Proč byla volána funkce	CallType
Hobj	Popisovač objektu	HOBJ
CallbackArea	Pole pro funkci zpětného volání k použití	CallbackArea
ConnectionArea	Pole pro funkci zpětného volání k použití	ConnectionArea
CompCode	Kód dokončení	CompCode
Reason	Kód příčiny	Příčina
State	Údaj o stavu aktuálního spotřebitele	Stav
DataLength	Délka zprávy	DataLength
BufferLength	Délka vyrovnávací paměti zpráv v bajtech	BufferLength
Flags	Obecné příznaky	Příznaky
Poznámka: Pokud je verze nižší než MQCBC_VERSION_2 , je zbývající pole ignorováno.		
ReconnectDelay	Počet milisekund před pokusem o opětovné připojení	ReconnectDelay

Přehled pro MQCBC

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSa klienti WebSphere MQ MQI, kteří jsou připojeni k těmto systémům.

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 MQCBC_VERSION_2.

Znaková sada a kódování: Data v MQCBC musí být ve znakové sadě, která je dána atributem správce front *CodedCharSetId* a kódováním lokálního správce front uvedeného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, bude struktura ve znakové sadě a kódování klienta.

Pole pro MQCBC

Abecední seznam polí pro strukturu MQCBC.

Struktura MQCBC obsahuje následující pole; pole jsou popsána v abecedním pořadí:

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.

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

Správce front neprovádí žádná rozhodnutí na základě obsahu tohoto pole a je předáván beze změny z pole “CallbackArea (MQPTR)” na stránce 265 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.

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

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

Byla vyvolána funkce obslužné rutiny událostí pro události výběrového vysílání; obslužná rutina událostí odešle události výběrového vysílání produktu WebSphere MQ místo 'normálních' událostí produktu WebSphere MQ .

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

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.

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 (MQPTR)” na stránce 311 ve struktuře MQCTLO, což je parametr volání MQCTL používaný 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.

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.

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.

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.

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.

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

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.

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. Tabulka 479 na stránce 263 Vypisuje hodnoty, které můžete nastavit k úpravě chování správce front při návratu z obslužné rutiny událostí.

Tabulka 479. 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.

Počáteční hodnoty a jazyková deklarace pro MQCBC
Struktura kontextu zpětného volání-počáteční hodnoty

Pro strukturu **MQCBC** nejsou žádné počáteční hodnoty. Struktura se předává jako parametr rutině zpětného volání. Správce front inicializuje strukturu; aplikace ji nikdy neinicializují.

Deklarace C
Kontextová struktura zpětného volání-deklarace jazyka C

```
typedef struct tagMQCBC MQCBC;
struct tagMQCBC {
   MQCHAR4   StrucId;                /* Structure identifier */
   MQLONG    Version;                /* Structure version number */
   MQLONG    CallType;               /* Why Function was called */
   MQHOBJ    Hobj;                   /* Object Handle */
   MQPTR     CallbackArea;           /* Callback data passed to the function */
   MQPTR     ConnectionArea;         /* MQCTL data area passed to the function */
   MQLONG    CompCode;               /* Completion Code */
   MQLONG    Reason;                 /* Reason Code */
   MQLONG    State;                  /* Consumer State */
   MQLONG    DataLength;             /* Message Data Length */
   MQLONG    BufferLength;           /* Buffer Length */
   MQLONG    Flags;                  /* Flags containing information about
                                        this consumer */
  /* Ver:1 */
   MQLONG    ReconnectDelay;         /* Number of milliseconds before */
  /* Ver:2 */  };                    /* reconnect attempt */
```

Deklarace COBOL

```
** MQCBC structure
 10  MQCBC.
** Structure Identifier
 15  MQCBC-STRUCID                    PIC X(4).
** Structure Version
 15  MQCBC-VERSION                    PIC S9(9) BINARY.
 ** Call Type
 15  MQCBC-CALLTYPE                   PIC S9(9) BINARY.
** Object Handle
```

```
 15  MQCBC-HOBJ                          PIC S9(9) BINARY.
** Callback User Area
 15  MQCBC-CALLBACKAREA                  POINTER
** Connection Area
 15  MQCBC-CONNECTIONAREA                POINTER
** Completion Code
 15  MQCBC-COMPCODE                      PIC S9(9) BINARY.
** Reason Code
 15  MQCBC-REASON                        PIC S9(9) BINARY.
** Consumer State
 15  MQCBC-STATE                         PIC S9(9) BINARY.
** Data Length
 15  MQCBC-DATALENGTH                    PIC S9(9) BINARY.
** Buffer Length
 15  MQCBC-BUFFERLENGTH                  PIC S9(9) BINARY.
** Flags
 15  MQCBC-FLAGS                         PIC S9(9) BINARY.
** Ver:1 **
** Number of milliseconds before reconnect attempt
 15  MQCBC-RECONNECTDELAY PIC S9(9) BINARY.
** Ver:2 **
```

Deklarace PL/I

```
dcl
 1 MQCBC based,
  3 StrucId             char(4),       /* Structure identifier */
  3 Version             fixed bin(31), /* Structure version */
  3 CallType            fixed bin(31), /* Callback type */
  3 Hobj                fixed bin(31), /* Object Handle */
  3 CallbackArea        pointer,       /* User area passed to the function */
  3 ConnectionArea      pointer,       /* Connection User Area */
  3 CompCode            fixed bin(31); /* Completion Code */
  3 Reason              fixed bin(31); /* Reason Code */
  3 State               fixed bin(31); /* Consumer State */
  3 DataLength          fixed bin(31); /* Message Data Length */
  3 BufferLength        fixed bin(31); /* Message Buffer length */
  3 Flags               fixed bin(31); /* Consumer Flags */
/* Ver:1 */
  3 ReconnectDelay      fixed bin(31); /* Number of milliseconds before */
/* Ver:2 */                           /* reconnect attempt */
```

Deklarace High Level Assembler

```
    MQCBC                 DSECT
    MQCBC                 DS  0F    Force fullword alignment
    MQCBC_STRUCID         DS  CL4   Structure identifier
    MQCBC_VERSION         DS  F     Structure version number
    MQCBC_CALLTYPE        DS  F     Why Function was called
    MQCBC_HOBJ            DS  F     Object Handle
    MQCBC_CALLBACKAREA    DS  A     Callback data passed to the function
    MQCBC_CONNECTIONAREA  DS  A     MQCTL Data area passed to the function
    MQCBC_COMPCODE        DS  F     Completion Code
    MQCBC_REASON          DS  F     Reason Code
    MQCBC_STATE           DS  F     Consumer State
    MQCBC_DATALENGTH      DS  F     Message Data Length
    MQCBC_BUFFERLENGTH    DS  F     Buffer Length
    MQCBC_FLAGS           DS  F     Flags containing information about this consumer
    MQCBC_RECONNECTDELAY  DS  F     Number of milliseconds before reconnect
    MQCBC_LENGTH          EQU *-MQCBC
                          ORG       MQCBC
    MQCBC_AREA            DS  CL(MQCBC_LENGTH)
```

MQCBD-Deskriptor zpětného volání

Následující tabulka shrnuje pole ve struktuře. Struktura určující funkci zpětného volání.

Tabulka 480. Pole v MQCBD		
Pole	**Popis**	**Téma**
StrucID	Identifikátor struktury	StrucID

Tabulka 480. Pole v MQCBD (pokračování)		
Pole	**Popis**	**Téma**
Version	Číslo verze struktury	verze
CallbackType	Typ funkce zpětného volání	CallbackType
Options	Volby, které řídí spotřebu zpráv	Volby
Callback Area	Pole pro funkci zpětného volání k použití	CallbackArea
CallbackFunction	Zda je funkce vyvolána jako volání rozhraní API	CallbackFunction
CallbackName	Zda je funkce vyvolána jako dynamicky propojený program	CallbackName
MaxMsgLength	Délka nejdelší zprávy, kterou lze číst	MaxMsgLength

Přehled pro MQCBD

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSa klienti WebSphere MQ MQI připojené k těmto systémům.

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 MQCBD_VERSION_1.

Znaková sada a kódování: Data v MQCBD musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front poskytnutého funkcí MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQCBD

Abecední seznam polí pro strukturu MQCBD.

Struktura MQCBD obsahuje následující pole; pole jsou popsána v abecedním pořadí:

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 nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě z pole "CallbackArea (MQPTR)" na stránce 258 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.

V systému z/OS musí 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: Když používáte CICS s produktem WebSphere MQ V7.0.1, asynchronní spotřeba je podporována, pokud:

- Apar PK66866 se používá 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.

Systémy UNIX
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 produktu WebSphere MQ , pokud má program používat volání rozhraní API produktu WebSphere MQ . Všimněte si však, že pole "Objekt Hobj (MQHOBJ)" na stránce 261 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.

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:

- 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.

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 (MQLONG)" na stránce 261 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.

Volby (MQLONG)
Struktura deskriptoru zpětného volání-pole Volby

Lze zadat libovolný, nebo všechny následující hodnoty. Je-li požadována více než jedna volba, hodnoty mohou být:

- Přidáno společně (nepřidávejte stejnou konstantu více než jednou), nebo
- zkombinovat pomocí bitové operace OR (jestliže 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 se připojení (pro aplikaci CICS nebo IMS) nachází 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 (MQLONG)" na stránce 258 , 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQCBD

Struktura deskriptoru zpětného volání-počáteční hodnoty

Tabulka 481. Počáteční hodnoty polí v MQCBD

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQCBD_STRUCTION_ID	'CBD¬'
Version	MQCBD_VERSION_1	1
CallBackType	MQCBT_MESSAGE_CONSUMER	1
Options	MQCBDO_NONE	0
CallbackArea	Není	Nulový ukazatel nebo prázdné znaky null
CallbackFunction	Není	Nulový ukazatel nebo prázdné znaky null
CallbackName	Není	Nulový řetězec nebo prázdné znaky
MaxMsgLength	MQCBD_FULL_MSG_LENGTH	-1

Tabulka 481. Počáteční hodnoty polí v MQCBD (pokračování)

Název pole	Název konstanty	Hodnota konstanty

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Null řetězce nebo mezery označuje řetězec znaků null v programovacím jazyce C a prázdné znaky v jiných programovacích jazycích.
3. V programovacím jazyce C-proměnná makraHodnota MQCBD_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCBD MyCBD = {MQCBD_DEFAULT};
```

Deklarace C
Struktura deskriptoru zpětného volání-deklarace jazyka C

```
typedef struct tagMQCBD MQCBD;
 struct tagMQCBD {
   MQCHAR4    StrucId;                /* Structure identifier */
   MQLONG     Version;                /* Structure version number */
   MQLONG     CallBackType;           /* Callback function type */
   MQLONG     Options;                /* Options controlling message
                                         consumption */
   MQPTR      CallbackArea;           /* User data passed to the function */
   MQPTR      CallbackFunction;       /* Callback function pointer */
   MQCHAR128  CallbackName;           /* Callback name */
   MQLONG     MaxMsgLength;           /* Maximum message length */
 };
```

Deklarace COBOL

```
** MQCBCD structure
 10  MQCBD.
** Structure Identifier
 15  MQCBD-STRUCID                    PIC X(4).
** Structure Version
 15  MQCBD-VERSION                    PIC S9(9) BINARY.
 ** Callback Type
 15  MQCBD-CALLBACKTYPE               PIC S9(9) BINARY.
** Options
 15  MQCBD-OPTIONS                    PIC S9(9) BINARY.
** Callback User Area
 15  MQCBD-CALLBACKAREA               POINTER
** Callback Function Pointer
 15  MQCBD-CALLBACKFUNCTION           FUNCTION-POINTER
** Callback Program Name
 15  MQCBD-CALLBACKNAME               PIC X(128)
** Maximum Message Length
 15  MQCDB-MAXMSGLENGTH               PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQCBD based,
   3 StrucId             char(4),       /* Structure identifier*/
   3 Version             fixed bin(31), /* Structure version*/
   3 CallbackType        fixed bin(31), /* Callback function type */
   3 Options             fixed bin(31), /* Options */
   3 CallbackArea        pointer,       /* User area passed to the function */
   3 CallbackFunction    pointer,       /* Callback Function Pointer */
   3 CallbackName        char(128),     /* Callback Program Name */
   3 MaxMsgLength        fixed bin(31); /* Maximum Message Length */
```

MQCHARV-Řetězec proměnné délky

Následující tabulka shrnuje pole ve struktuře.

Pole	Popis	Téma
VSPtr	Ukazatel na řetězec proměnné délky	VSPtr
VSOffset	Odsazení v bajtech proměnné délky proměnné od začátku struktury, která obsahuje tuto strukturu MQCHARV	VSOffset
VSLength	Délka řetězce proměnné délky adresovaná polem VSPtr nebo VSOffset (v bajtech).	Délka VSLID
VSBufSize	Velikost vyrovnávací paměti adresovaná v poli VSPtr nebo VSOffset (v bajtech).	VSBufSize
VSCCSID	Identifikátor znakové sady řetězce s proměnnou délkou adresovaný polem VSPtr nebo VSOffset.	VSCCSID

Přehled pro MQCHARV

Dostupnost: AIX, HP-UX, Solaris, Linux, IBM i, Windowsa klienti WebSphere MQ MQI, kteří jsou připojeni k těmto systémům.

Účel: Použijte strukturu MQCHARV k popisu řetězce proměnné délky.

Znaková sada a kódování: Data ve struktuře MQCHARV musí být v kódování lokálního správce front, který je dán rozhraním MQENC_NATIVE a znaková sada pole VSCCSID v rámci struktury. Je-li aplikace spuštěna jako klient MQ , musí být struktura v kódování klienta. Některé znakové sady mají reprezentaci, která závisí na daném kódování. Je-li VSCCSID 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. Mějte na paměti, že COBOL nepodporuje datový typ ukazatele ve všech prostředích. Z tohoto důvodu lze data také adresovat pomocí polí, která obsahují posunutí dat od začátku struktury obsahující MQCHARV.

Programování COBOL

Chcete-li portportovat aplikaci mezi prostředími, musíte zjistit, zda je datový typ ukazatele dostupný ve všech zamýšlených prostředích. Pokud tomu tak není, aplikace musí adresovat data pomocí polí offsetu místo polí ukazatele.

V těchto prostředích, kde ukazatele nejsou podporovány, můžete deklarovat pole ukazatele jako bajtové řetězce odpovídající délky, přičemž počáteční hodnotou je řetězec bajtů se všemi znaky null. Neměňte tuto počáteční hodnotu, pokud používáte pole offsetu. Jednou z možností, 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 kopii, která má být použita.

Pole pro MQCHARV

Struktura MQCHARV obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

VSCCSID (MQLONG)
Jedná se o identifikátor znakové sady proměnné délky proměnné adresované poli VSPtr nebo VSOffset.

Počáteční hodnota tohoto pole je MQCCSI_APPL, která je definována produktem MQ , aby označovala, že by měla být změněna na identifikátor skutečné znakové sady aktuálního procesu. V důsledku toho není hodnota MQCCSI_APPL nikdy přidružena k řetězci s proměnnou délkou. Počáteční hodnotu tohoto pole lze změnit definováním odlišné hodnoty pro konstantu MQCCSI_APPL pro danou kompilační jednotku vhodnými prostředky pro programovací jazyk vaší aplikace.

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 (UCS-2 CCSID 1200 a 13488) 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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQCHARV

Počáteční hodnoty polí v MQCHARV

Název pole	Název konstanty	Hodnota konstanty
VSPtr	Není	Nulový ukazatel nebo bajty null.
VSOffset	Není	0
VSBufSize	DÉLKA MQVS_USE_VSLENGTH	0
VSLength	Není	0
VSCCSID	MQCCSI_APPL	-3

Poznámka: V programovacím jazyce C obsahuje proměnná makra MQCHARV_DEFAULT hodnoty uvedené výše. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCHARV MyVarStr = {MQCHARV_DEFAULT};
```

Deklarace C

```
typedef struct tagMQCHARV MQCHARV;
struct tagMQCHARV {
  MQPTR     VSPtr;              /* Address of variable length string */
  MQLONG    VSOffset;           /* Offset of variable length string */
  MQLONG    VSBufSize;          /* Size of buffer */
  MQLONG    VSLength;           /* Length of variable length string */
  MQLONG    VSCCSID;            /* CCSID of variable length string */
};
```

Deklarace 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.
```

Deklarace PL/I

```
dcl
 1 MQCHARV based,
  3 VSPtr       pointer,       /* Address of variable length string */
  3 VSOffset    fixed bin(31), /* Offset of variable length string */
  3 VSBufSize   fixed bin(31), /* Size of buffer */
  3 VSLength    fixed bin(31), /* Length of variable length string */
  3 VSCCSID     fixed bin(31); /* CCSID of variable length string */
```

Deklarace High Level Assembler

```
MQCHARV             DSECT
MQCHARV_VSPTR       DS   F     Address of variable length string
MQCHARV_VSOFFSET    DS   F     Offset of variable length string
MQCHARV_VSBUFSIZE   DS   F     Size of buffer
MQCHARV_VSLENGTH    DS   F     Length of variable length string
MQCHARV_VSCCSID     DS   F     CCSID of variable length string
*
```

```
MQCHARV_LENGTH      EQU  *-MQCHARV
                    ORG  MQCHARV
MQCHARV_AREA        DS   CL(MQCHARV_LENGTH)
```

Předefinování proměnné MQCCSI_APPL

Následující příklady ukazují, jak lze přepsat hodnotu proměnné MQCCSI_APPL v různých programovacích jazycích. Hodnotu MQCCSI_APPL můžete změnit odstraněním nutnosti nastavit VSCCSID pro každý řetězec proměnné délky zvlášť.

V těchto příkladech je CCSID nastaven na 1208; změňte tuto hodnotu na požadovanou hodnotu.
Tato hodnota se stane výchozí hodnotou, kterou lze přepsat nastavením hodnoty VSCCSID v libovolné specifické instanci MQCHARV.

Použití C

```
#define MQCCSI_APPL 1208
#include <cmqc.h>
```

Použití jazyka COBOL

```
COPY CMQXYZV REPLACING -3 BY 1208.
```

Využití PL/I

```
%MQCCSI_APPL = '1208';
%include syslib(cmqp);
```

Použití v asembleru System/390

```
MQCCSI_APPL EQU 1208
         CMQA LIST=NO
```

MQCIH-záhlaví mostu CICS

Následující tabulka shrnuje pole ve struktuře.

Tabulka 482. Pole v MQCIH

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQCIH	StrucLength
Encoding	Vyhrazené	Kódování
CodedCharSetId	Vyhrazené	CodedCharSetId
Format	Název formátu produktu MQ pro data následující MQCIH	Formát
Flags	Příznaky	Příznaky
ReturnCode	Návratový kód z mostu	ReturnCode
CompCode	Kód dokončení MQ nebo CICS EIBRESP	CompCode
Reason	Kód příčiny nebo zpětné vazby produktu MQ nebo CICS EIBRESP2	Příčina

Tabulka 482. Pole v MQCIH (pokračování)		
Pole	**Popis**	**Téma**
UOWControl	Ovládací prvek jednotky práce	UOWŘídicí prvek
GetWaitInterval	Interval čekání na volání MQGET vydaný úlohou mostu	IntervalGetWait
LinkType	Typ odkazu	LinkType
OutputDataLength	Délka výstupních dat COMMAREA	OutputDataDélka
FacilityKeepTime	Čas uvolnění zařízení mostu	FacilityKeep
ADSDescriptor	Odeslat/přijmout deskriptor ADS	ADSDescriptor
ConversationalTask	Určení, zda úloha může být konverzační	ConversationalTask
TaskEndStatus	Stav na konci úlohy	TaskEndStav
Facility	Token zařízení mostu	Poskytovaná služba
Function	Název volání MQ nebo funkce CICS EIBFN	funkce
AbendCode	Kód nestandardního konce	AbendCode
Authenticator	Heslo nebo přístupový lístek	Ověřovatel
Reserved1	Vyhrazené	Reserved1
ReplyToFormat	Název formátu MQ zprávy odpovědi	ReplyToFormát
RemoteSysId	ID vzdáleného systému CICS , které se má použít	RemoteSysID
RemoteTransId	CICS RTRANSID, který se má použít	RemoteTransID
TransactionId	Transakce pro připojení	TransactionId
FacilityLike	Atributy emulované terminálu	FacilityLike
AttentionId	Klíč AID	AttentionId
StartCode	Počáteční kód transakce	StartCode
CancelCode	Kód nekonečného transakce	CancelCode
NextTransactionId	Další transakce k připojení	NextTransaction
Reserved2	Vyhrazené	Reserved2
Reserved3	Vyhrazené	Reserved3
Poznámka: Zbývající pole nejsou přítomna, pokud *Version* je menší než MQCIH_VERSION_2.		
CursorPosition	Pozice kurzoru	CursorPosition
ErrorOffset	Posun chyby ve zprávě	ErrorOffset
InputItem	Vyhrazené	InputItem
Reserved4	Vyhrazené	Reserved4

Přehled pro MQCIH

Dostupnost: AIX, HP-UX, z/OS, Solaris, Linux, Windowsa WebSphere MQ Klient MQI připojený k těmto systémům.

Účel: Struktura MQCIH popisuje informace, které mohou být přítomny na začátku zprávy odeslané do mostu CICS prostřednictvím produktu WebSphere MQ pro systém z/OS.

Název formátu: MQFMT_CICS.

Verze: Aktuální verze MQCIH je MQCIH_VERSION_2. Pole, která existují pouze v poslední verzi struktury, jsou identifikována jako taková v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejvíce posledních verzí MQCIH s počáteční hodnotou pole *Version* nastavenou na hodnotu MQCIH_VERSION_2.

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 vlastní frontu mostu CICS , 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 je v některém z podporovaných znakových sad a kódování. Přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu mostu CICS , převádí strukturu MQCIH.
- Data zprávy aplikace následující za strukturou MQCIH musí být ve stejné znakové sadě a kódování jako struktura MQCIH. Ve struktuře MQCIH nelze použít pole *CodedCharSetId* a *Encoding* k určení znakové sady a kódování dat zprávy aplikace.

  Pokud data nejsou jedním z vestavěných formátů podporovaných správcem front, musíte data uživatelské procedury pro převod dat převést na základě data převodu dat.

Použití: Pokud aplikace vyžaduje hodnoty, které jsou stejné jako počáteční hodnoty zobrazené v produktu Tabulka 484 na stránce 285a je-li most spuštěn s parametrem AUTH=LOCAL nebo AUTH=IDENTIFY, můžete strukturu MQCIH ze zprávy vynechat. Ve všech ostatních případech musí být struktura přítomna.

Most přijímá strukturu MQCIH typu version-1 nebo version-2 , ale pro transakce 3270 je nutné použít strukturu version-2 .

Aplikace musí zajistit, aby pole dokumentovaná jako pole požadavku měly odpovídající hodnoty ve zprávě odeslané do mostu; tato pole jsou vstupem do mostu.

Pole dokumentovaná jako pole odezvy jsou nastavena pomocí mostu CICS 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. Tabulka 483 na stránce 276 ukazuje, která pole jsou nastavena pro různé hodnoty *ReturnCode*.

Tabulka 483. Obsah polí s informacemi o chybě ve struktuře MQCIH pro MQCIH

ReturnCode	***Function***	***CompCode***	***Reason***	***AbendCode***
MQCRC_OK	-	-	-	-
CHYBA 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	EIBFN CICS	CICS EIBRESP	CICS EIBRESP2	-
SOUBOR MQCRC_BRIDGE_ABEND MQCRC_APPLICATION_ABEND	-	-	-	ABKÓD CICS

Pole pro MQCIH

Struktura MQCIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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 .

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.

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.

Ověřovatel (MQCHAR8)
Hodnota tohoto pole je heslo nebo přístupový lístek.

Je-li pro most CICS aktivní ověření identifikátoru uživatele, produkt *Authenticator* se používá 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.

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.

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 a je převzato z jakéhokoli předchozího záhlaví WebSphere MQ .

CompCode (MQLONG)
Toto pole je polem odezvy. Jeho počáteční hodnota je MQCC_OK

Hodnota vrácená v tomto poli závisí na *ReturnCode*; viz Tabulka 483 na stránce 276.

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.

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.

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 předchozích záhlaví produktu WebSphere MQ .

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.

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 systému CICS , aby přidělil novou funkci 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.

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.

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.

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 CICS vezme synchronizační bod, když se program dokončí, pokud je dodán do jiné oblasti CICS . Most neurčuje, do kterého regionu CICS má být požadavek odeslán; to je řízeno definicí programu CICS nebo zařízením pro vyrovnávání pracovní zátěže.

Formát (MQCHAR8)
V tomto poli je zobrazen název formátu produktu WebSphere 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.

Funkce (MQCHAR4)
Toto pole je polem odezvy. Délka tohoto pole je dána proměnnou MQ_FUNCTION_LENGTH. Počáteční hodnota tohoto pole je MQCFUNC_NONE.

Hodnota vrácená v tomto poli závisí na *ReturnCode*; viz Tabulka 483 na stránce 276. Následující hodnoty jsou možné v případě, že produkt *Function* obsahuje název volání WebSphere MQ :

MQCFUNC_MQCONN
Volání MQCONN.

FUNKCE MQCFUNC_MQGET
Volání MQGET.

MQCFUNC_MQINQ
Volání MQINQ.

MQCFUNC_MQOPEN
Volání MQOPEN.

MQCFUNC_MQPUT
Volání MQPUT.

MQCFUNC_MQPUT1
Volání MQPUT1 .

MQCFUNC_NONE
Žádné telefonát.

Ve všech případech jsou pro programovací jazyk C také definovány konstanty MQCFUNC_ * _ARRAY; tyto konstanty mají stejné hodnoty jako odpovídající konstanty MQCFUNC_ *, ale jsou to pole znaků místo řetězců.

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 most CICS bude čekat na čas uvedený při spuštění mostu.

MQWI_UNLIMITED
Neomezený interval čekání.

InputItem (MQLONG)
Toto pole je rezervované pole. Hodnota musí být 0.

Toto pole není přítomno, pokud *Version* je menší než MQCIH_VERSION_2.

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.

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.

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.

If the length of the user data in the request message is smaller than *OutputDataLength*, the DATALENGTH option of the LINK command is used, enabling the LINK to be function-shipped efficiently to another CICS region.

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.

Příčina (MQLONG)
Toto pole je polem odezvy. Jeho počáteční hodnota je MQRC_NONE.

Hodnota vrácená v tomto poli závisí na *ReturnCode*; viz Tabulka 483 na stránce 276.

ID RemoteSys(MQCHAR4)
Toto pole uvádí identifikátor systému CICS pro systém CICS , který zpracovává požadavek.

Je-li toto pole prázdné, bude 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 k frontě požadavků produktu WebSphere MQ .
- Musí být přístupný prostřednictvím odkazů CICS ISC ze systému CICS na 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.

Formát ReplyTo(MQCHAR8)
Hodnota tohoto pole je název formátu WebSphere MQ zprávy odpovědi, která je odeslána jako odpověď na aktuální zprávu.

Pravidla pro kódování tohoto pole jsou stejná jako pravidla pro kódování pole *Format* v 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.

Reserved1 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

Reserved2 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

Reserved3 (MQCHAR8)
Toto pole je rezervované pole. Hodnota musí být 8 mezer.

Reserved4 (MQLONG)
Toto pole je rezervované pole. Hodnota musí být 0.

Toto pole není přítomno, pokud *Version* je menší než MQCIH_VERSION_2.

ReturnCode (MQLONG)
Hodnota tohoto pole je návratový kód z mostu CICS popisující výsledek zpracování prováděného mostem. Toto pole je polem odezvy, s počáteční hodnotou MQCRC_OK.

Pole *Function*, *CompCode*, *Reason* a *AbendCode* mohou obsahovat další informace (viz Tabulka 483 na stránce 276). Hodnota je jedna z následujících možností:

UKONČENÍ MQCRC_APPLICATION_ABEND
(5, X'005 ') Aplikace skončila abnormálně.

MQCRC_BRIDGE_ABEND
(4, X'004 ') CICS Bridge skončil abnormálně.

CHYBA MQCRC_BRIDGE_ERROR
(3, X'003 ') Most CICS detekoval chybu.

MQCRC_BRIDGE_TIMEOUT
(8, X'008 ') Druhá nebo pozdější zpráva v rámci aktuální jednotky práce, která nebyla přijata ve stanoveném čase.

CHYBA MQCRC_CICS_EXEC_ERROR
(1, X'001 ') EXEC CICS detekovala chybu.

CHYBA MQCRC_MQ_API_ERROR
(2, X'002 ') Volání MQ zjistilo chybu.

MQCRC_OK
(0, X'000 ') Bez chyby.

MQCRC_PROGRAM_NOT_AVAILABLE
(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.

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 produkt CICS Transaction Server verze 1.2je toto pole pouze polem požadavku; jeho hodnota v odpovědi není definována.

V případě produktu CICS Transaction Server verze 1.3 a následných 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.

StrucId (MQCHAR4)
Toto pole je polem požadavku s počáteční hodnotou proměnné MQCIH_STRUC_ID.

Hodnota musí být:

ID_KONSTRUKCE_MQCIH_
Identifikátor struktury záhlaví informací 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.

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 informací CICS .

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.

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 na CICS jako lokální transakci s počátečním programem, který je CSQCBP00. Toto pole je použito pouze v případě, že hodnota *UOWControl* má hodnotu MQCUOWC_FIRST nebo MQCUOWC_ONLY.

UOWControl (MQLONG)
Toto pole je pole požadavku, které řídí zpracování jednotky práce provedené pomocí mostu CICS . Počáteční hodnota tohoto pole je MQCUOWC_ONLY.

Můžete požadovat, aby most spustil jednu transakci, nebo jeden nebo více programů v rámci transakce. Pole označuje, zda most CICS spustí pracovní jednotku, 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).

Verze (MQLONG)
Toto pole je pole požadavku. Jeho počáteční hodnota je MQCIH_VERSION_2.

Hodnota musí být jedna z následujících:

MQCIH_VERSION_1
Struktura záhlaví informací Version-1 CICS .

MQCIH_VERSION_2
Struktura informačního záhlaví produktu 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 CICS .

Počáteční hodnoty a deklarace jazyka pro MQCIH

Tabulka 484. Počáteční hodnoty polí v MQCIH pro MQCIH

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_KONSTRUKCE_MQCIH_	'CIH¬'
Version	MQCIH_VERSION_2	2
StrucLength	MQCIH_LENGTH_2	180
Encoding	Není	0
CodedCharSetId	Není	0
Format	MQFMT_NONE	Mezery
Flags	MQCIH_NONE	0
ReturnCode	MQCRC_OK	0
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
UOWControl	POUZE MQCUOWC_ONLY	273
GetWaitInterval	MQCGWI_DEFAULT	-2
LinkType	MQCLT_PROGRAM	1
OutputDataLength	MQCODL_AS_INPUT	-1
FacilityKeepTime	Není	0
ADSDescriptor	MQCADSD_NONE	0
ConversationalTask	MQCCT_NO	0
TaskEndStatus	MQCTES_NOSYNC	0
Facility	MQCFAC_NONE	Hodnoty null
Function	MQCFUNC_NONE	Mezery
AbendCode	Není	Mezery
Authenticator	Není	Mezery
Reserved1	Není	Mezery
ReplyToFormat	MQFMT_NONE	Mezery
RemoteSysId	Není	Mezery
RemoteTransId	Není	Mezery
TransactionId	Není	Mezery

Tabulka 484. Počáteční hodnoty polí v MQCIH pro MQCIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
FacilityLike	Není	Mezery
AttentionId	Není	Mezery
StartCode	MQCSC_NONE	Mezery
CancelCode	Není	Mezery
NextTransactionId	Není	Mezery
Reserved2	Není	Mezery
Reserved3	Není	Mezery
CursorPosition	Není	0
ErrorOffset	Není	0
InputItem	Není	0
Reserved4	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQCIH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCIH MyCIH = {MQCIH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQCIH MQCIH;
struct tagMQCIH {
  MQCHAR4  StrucId;             /* Structure identifier */
  MQLONG   Version;             /* Structure version number */
  MQLONG   StrucLength;         /* Length of MQCIH structure */
  MQLONG   Encoding;            /* Reserved */
  MQLONG   CodedCharSetId;      /* Reserved */
  MQCHAR8  Format;              /* MQ format name of data that follows
                                   MQCIH */
  MQLONG   Flags;               /* Flags */
  MQLONG   ReturnCode;          /* Return code from bridge */
  MQLONG   CompCode;            /* MQ completion code or CICS EIBRESP */
  MQLONG   Reason;              /* MQ reason or feedback code, or CICS
                                   EIBRESP2 */
  MQLONG   UOWControl;          /* Unit-of-work control */
  MQLONG   GetWaitInterval;     /* Wait interval for MQGET call issued
                                   by bridge task */
  MQLONG   LinkType;            /* Link type */
  MQLONG   OutputDataLength;    /* Output COMMAREA data length */
  MQLONG   FacilityKeepTime;    /* Bridge facility release time */
  MQLONG   ADSDescriptor;       /* Send/receive ADS descriptor */
  MQLONG   ConversationalTask;  /* Whether task can be conversational */
  MQLONG   TaskEndStatus;       /* Status at end of task */
  MQBYTE8  Facility;            /* Bridge facility token */
  MQCHAR4  Function;            /* MQ call name or CICS EIBFN
                                   function */
  MQCHAR4  AbendCode;           /* Abend code */
  MQCHAR8  Authenticator;       /* Password or passticket */
  MQCHAR8  Reserved1;           /* Reserved */
  MQCHAR8  ReplyToFormat;       /* MQ format name of reply message */
  MQCHAR4  RemoteSysId;         /* Reserved */
  MQCHAR4  RemoteTransId;       /* Reserved */
  MQCHAR4  TransactionId;       /* Transaction to attach */
  MQCHAR4  FacilityLike;        /* Terminal emulated attributes */
  MQCHAR4  AttentionId;         /* AID key */
```

```
  MQCHAR4  StartCode;           /* Transaction start code */
  MQCHAR4  CancelCode;          /* Abend transaction code */
  MQCHAR4  NextTransactionId;   /* Next transaction to attach */
  MQCHAR8  Reserved2;           /* Reserved */
  MQCHAR8  Reserved3;           /* Reserved */
  MQLONG   CursorPosition;      /* Cursor position */
  MQLONG   ErrorOffset;         /* Offset of error in message */
  MQLONG   InputItem;           /* Reserved */
  MQLONG   Reserved4;           /* Reserved */
};
```

Deklarace COBOL

```
**   MQCIH structure
  10 MQCIH.
**    Structure identifier
   15 MQCIH-STRUCID            PIC X(4).
**    Structure version number
   15 MQCIH-VERSION            PIC S9(9) BINARY.
**    Length of MQCIH structure
   15 MQCIH-STRUCLENGTH        PIC S9(9) BINARY.
**    Reserved
   15 MQCIH-ENCODING           PIC S9(9) BINARY.
**    Reserved
   15 MQCIH-CODEDCHARSETID     PIC S9(9) BINARY.
**    MQ format name of data that follows MQCIH
   15 MQCIH-FORMAT             PIC X(8).
**    Flags
   15 MQCIH-FLAGS              PIC S9(9) BINARY.
**    Return code from bridge
   15 MQCIH-RETURNCODE         PIC S9(9) BINARY.
**    MQ completion code or CICS EIBRESP
   15 MQCIH-COMPCODE           PIC S9(9) BINARY.
**    MQ reason or feedback code, or CICS EIBRESP2
   15 MQCIH-REASON             PIC S9(9) BINARY.
**    Unit-of-work control
   15 MQCIH-UOWCONTROL         PIC S9(9) BINARY.
**    Wait interval for MQGET call issued by bridge task
   15 MQCIH-GETWAITINTERVAL    PIC S9(9) BINARY.
**    Link type
   15 MQCIH-LINKTYPE           PIC S9(9) BINARY.
**    Output COMMAREA data length
   15 MQCIH-OUTPUTDATALENGTH   PIC S9(9) BINARY.
**    Bridge facility release time
   15 MQCIH-FACILITYKEEPTIME   PIC S9(9) BINARY.
**    Send/receive ADS descriptor
   15 MQCIH-ADSDESCRIPTOR      PIC S9(9) BINARY.
**    Whether task can be conversational
   15 MQCIH-CONVERSATIONALTASK PIC S9(9) BINARY.
**    Status at end of task
   15 MQCIH-TASKENDSTATUS      PIC S9(9) BINARY.
**    Bridge facility token
   15 MQCIH-FACILITY           PIC X(8).
**    MQ call name or CICS EIBFN function
   15 MQCIH-FUNCTION           PIC X(4).
**    Abend code
   15 MQCIH-ABENDCODE          PIC X(4).
**    Password or passticket
   15 MQCIH-AUTHENTICATOR      PIC X(8).
**    Reserved
   15 MQCIH-RESERVED1          PIC X(8).
**    MQ format name of reply message
   15 MQCIH-REPLYTOFORMAT      PIC X(8).
**    Reserved
   15 MQCIH-REMOTESYSID        PIC X(4).
**    Reserved
   15 MQCIH-REMOTETRANSID      PIC X(4).
**    Transaction to attach
   15 MQCIH-TRANSACTIONID      PIC X(4).
**    Terminal emulated attributes
   15 MQCIH-FACILITYLIKE       PIC X(4).
**    AID key
   15 MQCIH-ATTENTIONID        PIC X(4).
**    Transaction start code
   15 MQCIH-STARTCODE          PIC X(4).
**    Abend transaction code
   15 MQCIH-CANCELCODE         PIC X(4).
**    Next transaction to attach
   15 MQCIH-NEXTTRANSACTIONID  PIC X(4).
```

```
**    Reserved
   15 MQCIH-RESERVED2          PIC X(8).
**    Reserved
   15 MQCIH-RESERVED3          PIC X(8).
**    Cursor position
   15 MQCIH-CURSORPOSITION     PIC S9(9) BINARY.
**    Offset of error in message
   15 MQCIH-ERROROFFSET        PIC S9(9) BINARY.
**    Reserved
   15 MQCIH-INPUTITEM          PIC S9(9) BINARY.
**    Reserved
   15 MQCIH-RESERVED4          PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQCIH based,
  3 StrucId             char(4),       /* Structure identifier */
  3 Version             fixed bin(31), /* Structure version number */
  3 StrucLength         fixed bin(31), /* Length of MQCIH structure */
  3 Encoding            fixed bin(31), /* Reserved */
  3 CodedCharSetId      fixed bin(31), /* Reserved */
  3 Format              char(8),       /* MQ format name of data that
                                          follows MQCIH */
  3 Flags               fixed bin(31), /* Flags */
  3 ReturnCode          fixed bin(31), /* Return code from bridge */
  3 CompCode            fixed bin(31), /* MQ completion code or CICS
                                          EIBRESP */
  3 Reason              fixed bin(31), /* MQ reason or feedback code, or
                                          CICS EIBRESP2 */
  3 UOWControl          fixed bin(31), /* Unit-of-work control */
  3 GetWaitInterval     fixed bin(31), /* Wait interval for MQGET call
                                          issued by bridge task */
  3 LinkType            fixed bin(31), /* Link type */
  3 OutputDataLength    fixed bin(31), /* Output COMMAREA data length */
  3 FacilityKeepTime    fixed bin(31), /* Bridge facility release time */
  3 ADSDescriptor       fixed bin(31), /* Send/receive ADS descriptor */
  3 ConversationalTask fixed bin(31), /* Whether task can be
                                          conversational */
  3 TaskEndStatus       fixed bin(31), /* Status at end of task */
  3 Facility            char(8),       /* Bridge facility token */
  3 Function            char(4),       /* MQ call name or CICS EIBFN
                                          function */
  3 AbendCode           char(4),       /* Abend code */
  3 Authenticator       char(8),       /* Password or passticket */
  3 Reserved1           char(8),       /* Reserved */
  3 ReplyToFormat       char(8),       /* MQ format name of reply
                                          message */
  3 RemoteSysId         char(4),       /* Reserved */
  3 RemoteTransId       char(4),       /* Reserved */
  3 TransactionId       char(4),       /* Transaction to attach */
  3 FacilityLike        char(4),       /* Terminal emulated attributes */
  3 AttentionId         char(4),       /* AID key */
  3 StartCode           char(4),       /* Transaction start code */
  3 CancelCode          char(4),       /* Abend transaction code */
  3 NextTransactionId  char(4),       /* Next transaction to attach */
  3 Reserved2           char(8),       /* Reserved */
  3 Reserved3           char(8),       /* Reserved */
  3 CursorPosition      fixed bin(31), /* Cursor position */
  3 ErrorOffset         fixed bin(31), /* Offset of error in message */
  3 InputItem           fixed bin(31), /* Reserved */
  3 Reserved4           fixed bin(31); /* Reserved */
```

Deklarace High Level Assembler

```
MQCIH                   DSECT
MQCIH_STRUCID           DS   CL4  Structure identifier
MQCIH_VERSION           DS   F    Structure version number
MQCIH_STRUCLENGTH       DS   F    Length of MQCIH structure
MQCIH_ENCODING          DS   F    Reserved
MQCIH_CODEDCHARSETID    DS   F    Reserved
MQCIH_FORMAT            DS   CL8  MQ format name of data that follows
*                                 MQCIH
MQCIH_FLAGS             DS   F    Flags
MQCIH_RETURNCODE        DS   F    Return code from bridge
MQCIH_COMPCODE          DS   F    MQ completion code or CICS EIBRESP
```

```
MQCIH_REASON              DS   F    MQ reason or feedback code, or CICS
*                                   EIBRESP2
MQCIH_UOWCONTROL          DS   F    Unit-of-work control
MQCIH_GETWAITINTERVAL     DS   F    Wait interval for MQGET call issued
*                                   by bridge task
MQCIH_LINKTYPE            DS   F    Link type
MQCIH_OUTPUTDATALENGTH    DS   F    Output COMMAREA data length
MQCIH_FACILITYKEEPTIME    DS   F    Bridge facility release time
MQCIH_ADSDESCRIPTOR       DS   F    Send/receive ADS descriptor
MQCIH_CONVERSATIONALTASK  DS   F    Whether task can be conversational
MQCIH_TASKENDSTATUS       DS   F    Status at end of task
MQCIH_FACILITY            DS   XL8  Bridge facility token
MQCIH_FUNCTION            DS   CL4  MQ call name or CICS EIBFN function
MQCIH_ABENDCODE           DS   CL4  Abend code
MQCIH_AUTHENTICATOR       DS   CL8  Password or passticket
MQCIH_RESERVED1           DS   CL8  Reserved
MQCIH_REPLYTOFORMAT       DS   CL8  MQ format name of reply message
MQCIH_REMOTESYSID         DS   CL4  Reserved
MQCIH_REMOTETRANSID       DS   CL4  Reserved
MQCIH_TRANSACTIONID       DS   CL4  Transaction to attach
MQCIH_FACILITYLIKE        DS   CL4  Terminal emulated attributes
MQCIH_ATTENTIONID         DS   CL4  AID key
MQCIH_STARTCODE           DS   CL4  Transaction start code
MQCIH_CANCELCODE          DS   CL4  Abend transaction code
MQCIH_NEXTTRANSACTIONID   DS   CL4  Next transaction to attach
MQCIH_RESERVED2           DS   CL8  Reserved
MQCIH_RESERVED3           DS   CL8  Reserved
MQCIH_CURSORPOSITION      DS   F    Cursor position
MQCIH_ERROROFFSET         DS   F    Offset of error in message
MQCIH_INPUTITEM           DS   F    Reserved
MQCIH_RESERVED4           DS   F    Reserved
*
MQCIH_LENGTH              EQU  *-MQCIH
                          ORG  MQCIH
MQCIH_AREA                DS   CL(MQCIH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQCIH
  StrucId            As String*4 'Structure identifier'
  Version            As Long     'Structure version number'
  StrucLength        As Long     'Length of MQCIH structure'
  Encoding           As Long     'Reserved'
  CodedCharSetId     As Long     'Reserved'
  Format             As String*8 'MQ format name of data that follows'
                                 'MQCIH'
  Flags              As Long     'Flags'
  ReturnCode         As Long     'Return code from bridge'
  CompCode           As Long     'MQ completion code or CICS EIBRESP'
  Reason             As Long     'MQ reason or feedback code, or CICS'
                                 'EIBRESP2'
  UOWControl         As Long     'Unit-of-work control'
  GetWaitInterval    As Long     'Wait interval for MQGET call issued'
                                 'by bridge task'
  LinkType           As Long     'Link type'
  OutputDataLength   As Long     'Output COMMAREA data length'
  FacilityKeepTime   As Long     'Bridge facility release time'
  ADSDescriptor      As Long     'Send/receive ADS descriptor'
  ConversationalTask As Long     'Whether task can be conversational'
  TaskEndStatus      As Long     'Status at end of task'
  Facility           As MQBYTE8  'Bridge facility token'
  Function           As String*4 'MQ call name or CICS EIBFN function'
  AbendCode          As String*4 'Abend code'
  Authenticator      As String*8 'Password or passticket'
  Reserved1          As String*8 'Reserved'
  ReplyToFormat      As String*8 'MQ format name of reply message'
  RemoteSysId        As String*4 'Reserved'
  RemoteTransId      As String*4 'Reserved'
  TransactionId      As String*4 'Transaction to attach'
  FacilityLike       As String*4 'Terminal emulated attributes'
  AttentionId        As String*4 'AID key'
  StartCode          As String*4 'Transaction start code'
  CancelCode         As String*4 'Abend transaction code'
  NextTransactionId  As String*4 'Next transaction to attach'
  Reserved2          As String*8 'Reserved'
  Reserved3          As String*8 'Reserved'
  CursorPosition     As Long     'Cursor position'
  ErrorOffset        As Long     'Offset of error in message'
  InputItem          As Long     'Reserved'
```

```
  Reserved4          As Long     'Reserved'
End Type
```

MQCMHO-Vytvoření voleb popisovače zprávy

Následující tabulka shrnuje pole ve struktuře.

Tabulka 485. Pole v MQCMHO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby

Přehled pro MQCMHO

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS a klienti WebSphere MQ .

Účel: Struktura **MQCMHO** umožňuje aplikacím určit volby, které řídí způsob vytváření obslužných rutin zpráv. Struktura je vstupním parametrem na volání **MQCRTMH** .

Znaková sada a kódování: Data v souboru **MQCMHO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole pro MQCMHO

Struktura MQCMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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 JMS.
- není jedním z následujících klíčových slov, v libovolné směsi malých nebo velkých písmen:
  - AND
  - BETWEEN
  - Esc
  - NEPRAVDA
  - IN
  - IS

  - Jako
  - NE
  - NULL
  - NEBO
  - 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQCMHO

Tabulka 486. Počáteční hodnoty polí v MQCMHO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQCMHO_STRUCTION_ID	'CMHO'

Tabulka 486. Počáteční hodnoty polí v MQCMHO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
Version	MQCMHO_VERSION_1	1
Options	MQCMHO_DEFAULT_VALIDATION	0

Notes:

1. V programovacím jazyce C-proměnná makraMQCMHO_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCMHO MyCMHO = {MQCMHO_DEFAULT};
```

Deklarace C

```
struct tagMQCMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQCRTMH */
};
```

Deklarace COBOL

```
**   MQCMHO structure
  10 MQCMHO.
**    Structure identifier
   15 MQCMHO-STRUCID        PIC X(4).
**    Structure version number
   15 MQCMHO-VERSION        PIC S9(9) BINARY.
**    Options that control the action of MQCRTMH
   15 MQCMHO-OPTIONS        PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQCMHO based,
   3 StrucId        char(4),       /* Structure identifier */
   3 Version        fixed bin(31), /* Structure version number */
   3 Options        fixed bin(31), /* Options that control the action of MQCRTMH */
```

Deklarace High Level Assembler

```
MQCMHO                DSECT
MQCMHO_STRUCID        DS   CL4   Structure identifier
MQCMHO_VERSION        DS   F     Structure version number
MQCMHO_OPTIONS        DS   F     Options that control the action of
*                                MQCRTMH
MQCMHO_LENGTH         EQU  *-MQCMHO
MQCMHO_AREA           DS   CL(MQCMHO_LENGTH)
```

MQCNO-Volby připojení

Následující tabulka shrnuje pole ve struktuře.

Tabulka 487. Pole v MQCNO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId

Tabulka 487. Pole v MQCNO (pokračování)		
Pole	**Popis**	**Téma**
Version	Číslo verze struktury	verze
Options	Volby, které řídí akci MQCONNX	Volby
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQCNO_VERSION_2.		
ClientConnOffset	Offset struktury MQCD pro připojení klienta	PosunutíClientConn
ClientConnPtr	Adresa struktury MQCD pro připojení klienta	ClientConnPtr
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQCNO_VERSION_3.		
ConnTag	Značka připojení správce front	ConnTag
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQCNO_VERSION_4.		
SSLConfigPtr	Adresa struktury MQSCO pro připojení klienta	SSLConfigPtr
SSLConfigOffset	Offset struktury MQSCO pro připojení klienta	SSLConfigOffset
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQCNO_VERSION_5.		
ConnectionId	Jedinečné ID připojení	ConnectionId
SecurityParmsOffset	Parametry zabezpečení	PosunutíSecurityParms
SecurityParmsPtr	Parametry zabezpečení	SecurityParmsPtr

Související úlohy
Použití MQCONNX

Přehled pro MQCNO

Availability: Všechny verze s výjimkou MQCNO_VERSION_4: AIX, HP-UX, IBM i, Solaris, Linux, Windowsa WebSphere MQ MQI MQI připojené k těmto systémům.

Účel: Struktura MQCNO umožňuje aplikaci zadat volby související s připojením k lokálnímu správci front. Struktura je vstupním/výstupním parametrem pro volání MQCONNX. Další informace o použití sdílených popisovačů a volání MQCONNX naleznete v tématu Sdílená připojení (nezávislá na podprocesech) s produktem MQCONNX.

Verze: Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQCNO, ale s počáteční hodnotou pole *Version* nastavenou na hodnotu MQCNO_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí nastavit pole *Version* na číslo verze požadované verze.

Znaková sada a kódování: Data ve struktuře MQCNO musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient WebSphere MQ MQI, musí být struktura ve znakové sadě a kódování klienta.

Pole pro MQCNO

Struktura MQCNO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

Produkt *ClientConnOffset* používejte pouze v případě, že je aplikace, která vydala volání MQCONNX, spuštěna jako klient WebSphere MQ MQI. 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 produkty *ClientConnOffset* a *ClientConnPtr* pouze tehdy, je-li aplikace, která vydala volání MQCONNX, spuštěna jako klient WebSphere MQ MQI. 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 klient WebSphere MQ MQI, ale neposkytuje strukturu MQCD, použije se k výběru definice kanálu proměnná prostředí MQSERVER . Není-li parametr MQSERVER nastaven, použije se tabulka kanálů klienta.

Není-li aplikace spuštěna jako klient WebSphere MQ MQI, 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 1000 .

Pole v MQCD	**Hodnota**
ChannelName	Název kanálu.
Version	Číslo verze struktury. Nesmí být menší než MQCD_VERSION_7.
TransportType	Libovolný podporovaný typ transportu.
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.

Pole v MQCD	Hodnota
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	SSL CipherSpec.
SSLPeerNamePtr	Adresa názvu partnera SSL.
SSLPeerNameLength	Délka názvu partnera SSL.
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.

Bez ohledu na to, jakou techniku zvolíte, můžete použít pouze jeden z produktů *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.

ConnectionId (MQBYTE24)
ConnectionId je jedinečný 24bajtový identifikátor, který umožňuje produktu WebSphere 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 bude jedinečný mezi všemi ostatními identifikátory, které generuje produkt WebSphere MQ , například 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.

ConnTag (MQBYTE128)
ConnTag je značka, kterou správce front přidružuje k prostředkům, které jsou ovlivněny aplikací během tohoto připojení. Každá aplikace nebo instance aplikace musí pro značku použít jinou hodnotu, aby správce front mohl správně serializovat přístup k ovlivněným prostředkům. Toto pole je vstupním polem a jeho počáteční hodnota je MQCT_NONE.

Další podrobnosti o hodnotách, které mají být použity různými aplikacemi, najdete v popisech voleb MQCNO_ * _CONN_TAG_ *. Tato značka přestane být platná při ukončení aplikace nebo při vyvolání volání MQDISC.

Poznámka: Hodnoty značek připojení začínající řetězcem MQ v malých, nižších nebo smíšených případech v kódování ASCII nebo EBCDIC jsou vyhrazeny pro použití produkty IBM . Nepoužívejte hodnoty značky připojení začínající těmito písmeny.

Pokud vyžadujete žádnou značku, použijte následující speciální hodnotu:

MQCT_NONE
Hodnota je binární nula pro délku pole.

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

Toto pole se používá při připojování ke správci front z/OS . V jiných prostředích zadejte hodnotu MQCT_NONE.

Délka tohoto pole je dána hodnotou MQ_CONN_TAG_LENGTH. Toto pole je ignorováno, pokud *Version* je menší než MQCNO_VERSION_3.

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 vypnuto v definici správce front nastavením atributu *MQIAccounting* na hodnotu MQMON_OFF, je při nastavení tohoto příznaku povolena kolekce dat evidence MQI.

MQCNO_ACCOUNTING_MQI_DISABLED
Je-li shromažďování dat monitorování vypnuto 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 vypnuto v definici správce front nastavením atributu *MQIAccounting* na hodnotu MQMON_OFF, nastavení tohoto příznaku povolí 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 vypnuto v definici správce front nastavením atributu *MQIAccounting* na hodnotu MQMON_OFF, nastavení tohoto příznaku vypne shromažďování dat evidence pro fronty, které určují správce front v poli *MQIAccounting* příslušné definice fronty.

Pokud není definován žádný z těchto parametrů, účtování pro připojení je definováno v atributech správce front.

Volby vazeb

Následující volby řídí typ vazby produktu WebSphere 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` (nebo odpovídající položku registru systému Windows) k výběru skutečného typu 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.

- Je-li zadána proměnná prostředí MQ_CONNECT_TYPE, může být dodána jedna z následujících voleb pro změnu chování MQCONN nebo MQCONNX, je-li zadáno MQCNO_STANDARD_BINDING. (Výjimkou je, je-li hodnota MQCNO_FASTPATH_BINDING zadána s hodnotou MQ_CONNECT_TYPE nastaveným na hodnotu LOCAL nebo STANDARD , aby umožnila snížení úrovně zkrácené cesty administrátorem bez související změny aplikace:

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 .
- V systému IBM i musí být úloha spuštěna pod profilem uživatele, který patří do skupiny QMQMADM . Program také nesmí být nestandardně ukončen, jinak může dojít k nepředvídatelným výsledkům.
- Na systémech UNIX musí 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 WebSphere MQ Object Authority Manager (OAM) stále používá skutečné ID uživatele pro kontrolu oprávnění.

- V systému Windowsmusí být program členem skupiny `mqm` . Vázání zrychleného přístupu není podporováno pro 64bitové aplikace.

Volba MQCNO_FASTPATH_BINDING je podporována v následujících prostředích: AIX, HP-UX, IBM i, Solaris, Linux, and Windows. V systému z/OSje volba přijata, ale ignorována.

Další informace o důsledcích použití důvěryhodných aplikací naleznete v tématu Omezení pro důvěryhodné aplikace .

CQCNO_SHARED_BINDING

S parametrem MQCNO_SHARED_BINDING bude aplikace a agent local-queue-manager sdílet některé 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:

- Hodnota MQCNO_CLIENT_BINDING byla v systému z/OS odmítnuta s chybou MQRC_OPTIONS_ERROR.
- 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 prostředí Java nebo .NET, protože mají vlastní mechanizmy pro výběr typu vazby.
- 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.

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:

- Hodnota MQCNO_LOCAL_BINDING je v systému z/OSignorována.
- 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 prostředí Java nebo .NET, protože mají vlastní mechanizmy pro výběr typu vazby.
- 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.

V systému AIX, HP-UX, Solaris, Linuxa Windowsmůžete použít proměnnou prostředí `MQ_CONNECT_TYPE` s typem vazby zadaným polem *`Options`* , abyste mohli řídit typ použité vazby. 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 proměnnou prostředí podporovanou hodnotu, použije se vazba fastpath pouze v případě, že proměnná prostředí a pole *Options both* určují vazbu zkrácené cesty.

Volby značek připojení

Tyto volby jsou podporovány pouze při připojování ke správci front z/OS a řídí použití značky připojení *ConnTag*. Můžete uvést pouze jednu z těchto voleb:

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 patří lokální správce front.

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.

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 jinde ve skupině sdílení front, do níž patří lokální správce front.

- Aplikace musí být spuštěny ve stejném adresním prostoru MVS , aby bylo možné sdílet značku připojení. Je-li aplikace používající značku připojení aplikací klienta, MQCNO_RESTRICT_CONN_TAG_Q_MGR není povolen.

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 ve skupině sdílení front již používána, 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 být spuštěny ve stejném adresním prostoru MVS , aby bylo možné sdílet 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

Tyto volby jsou podporovány v následujících prostředích: AIX, HP-UX, IBM i, Solaris, Linux , a 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:

- 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 třídami produktu WebSphere MQ pro prostředí 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 označuje, 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 produktu starších než verze 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 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 je ignorováno, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v produktu "MQCSP-parametry zabezpečení" na stránce 307.

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 je ignorováno, pokud je *Verze* menší než MQCNO_VERSION_5.

Struktura MQCSP je definována v produktu "MQCSP-parametry zabezpečení" na stránce 307.

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.

Produkt *SSLConfigOffset* používejte pouze v případě, že je aplikace, která vydala volání MQCONNX, spuštěna jako klient WebSphere MQ MQI. 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.

SSLConfigPtr (PMQSCO)
SSLConfigPtr je vstupní pole. 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 tehdy, je-li aplikace, která volala volání MQCONNX, spuštěna jako klient WebSphere MQ MQI a protokol kanálu je TCP/IP. Pokud aplikace není spuštěna jako klient WebSphere 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 SSL pro připojení klienta. Když jsou informace o zabezpečení SSL zadány tímto způsobem, proměnné prostředí MQSSLKEYR a MQSSLCRYP jsou ignorovány; všechny informace související s SSL v tabulce definic kanálů klienta (CCDT) se také ignorují.

Informace SSL lze zadat pouze na:

- První volání MQCONNX procesu typu klient, nebo

- Následné volání MQCONNX při uzavření všech předchozích připojení SSL/TLS ke správci front pomocí funkce MQDISC.

Toto jsou jediné stavy, v nichž lze inicializovat prostředí SSL celého procesu. Je-li volání MQCONNX vydáno se zadáním informací o zabezpečení SSL, když prostředí SSL již existuje, informace o zabezpečení SSL na volání se budou ignorovat a připojení se provede s použitím existujícího prostředí SSL; 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 do *SSLConfigPtr*, nebo zadáním posunu v *SSLConfigOffset*; podrobnosti o tom, jak to provést, najdete 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.

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
Struktura connect-options Version-1 .

MQCNO_VERSION_2
Struktura voleb připojení Version-2 .

MQCNO_VERSION_3
Struktura connect-options Version-3 .

MQCNO_VERSION_4
Struktura connect-options Version-4 .

MQCNO_VERSION_5
Struktura connect-options Version-5 .

Tato verze struktury MQCNO rozšiřuje MQCNO_VERSION_3 na systémech z/OSa MQCNO_VERSION_4 na všechny ostatní platformy.

Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech polí. Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQCNO_CURRENT_VERSION
Aktuální verze struktury voleb připojení.

Počáteční hodnoty a deklarace jazyka pro MQCNO

Tabulka 488. Počáteční hodnoty polí v MQCNO pro MQCNO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQCNO_STRUCTION_ID	'CNO¬'

Tabulka 488. Počáteční hodnoty polí v MQCNO pro MQCNO (pokračování)

Název pole	Název konstanty	Hodnota konstanty
Version	MQCNO_VERSION_1	1
Options	MQCNO_NONE	0
ClientConnOffset	Není	0
ClientConnPtr	Není	Nulový ukazatel nebo bajty null
ConnTag	MQCT_NONE	Hodnoty null
SSLConfigPtr	Není	Nulový ukazatel nebo bajty null
SSLConfigOffset	Není	0
ConnectionId	Není	Nulový ukazatel nebo bajty null
SecurityParmsOffset	Není	Nulový ukazatel nebo bajty null
SecurityParmsPtr	Není	Nulový ukazatel nebo bajty null

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQCNO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCNO MyCNO = {MQCNO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQCNO MQCNO;
struct tagMQCNO {
  MQCHAR4    StrucId;           /* Structure identifier */
  MQLONG     Version;           /* Structure version number */
  MQLONG     Options;           /* Options that control the action of
                                   MQCONNX */
  MQLONG     ClientConnOffset;  /* Offset of MQCD structure for client
                                   connection */
  MQPTR      ClientConnPtr;     /* Address of MQCD structure for client
                                   connection */
  MQBYTE128  ConnTag;           /* Queue-manager connection tag */
  PMQSCO     SSLConfigPtr;      /* Address of MQSCO structure for client
                                   connection */
  MQLONG     SSLConfigOffset;   /* Offset of MQSCO structure for client
                                   connection */
  MQBYTE24   ConnectionId;      /* Unique connection identifier */
  MQLONG     SecurityParmsOffset /* Security fields */
  PMQCSP     SecurityParmsPtr   /* Security parameters */
 };
```

Deklarace COBOL

```
**   MQCNO structure
  10 MQCNO.
**    Structure identifier
   15 MQCNO-STRUCID          PIC X(4).
**    Structure version number
   15 MQCNO-VERSION          PIC S9(9) BINARY.
**    Options that control the action of MQCONNX
   15 MQCNO-OPTIONS          PIC S9(9) BINARY.
**    Offset of MQCD structure for client connection
   15 MQCNO-CLIENTCONNOFFSET PIC S9(9) BINARY.
**    Address of MQCD structure for client connection
```

```
   15 MQCNO-CLIENTCONNPTR    POINTER.
**    Queue-manager connection tag
   15 MQCNO-CONNTAG          PIC X(128).
**    Address of MQSCO structure for client connection
   15 MQCNO-SSLCONFIGPTR     POINTER.
**    Offset of MQSCO structure for client connection
   15 MQCNO-SSLCONFIGOFFSET  PIC S9(9) BINARY.
**    Unique connection identifier
   15 MQCNO-CONNECTIONID     PIC X(24).
**    Offset of MQCSP structure for security parameters
   15 MQCNO-SECURITYPARMSOFFSET PIC S9(9) BINARY.
**    Address of MQCSP structure for security parameters
   15 MQCNO-SECURITYPARMSPTR POINTER.
```

Deklarace PL/I

```
dcl
 1 MQCNO based,
  3 StrucId          char(4),       /* Structure identifier */
  3 Version          fixed bin(31), /* Structure version number */
  3 Options          fixed bin(31), /* Options that control the action
                                       of MQCONNX */
  3 ClientConnOffset fixed bin(31), /* Offset of MQCD structure for
                                       client connection */
  3 ClientConnPtr    pointer,       /* Address of MQCD structure for
                                       client connection */
  3 ConnTag          char(128),     /* Queue-manager connection tag */
  3 SSLConfigPtr     pointer,       /* Address of MQSCO structure for
                                       client connection */
  3 SSLConfigOffset  fixed bin(31), /* Offset of MQSCO structure for
                                       client connection */
  3 ConnectionId     char(24),      /* Unique connection identifier
  3 SecurityParmsOffset fixed bin(31); /* Offset of MQCSP structure for
                                       security parameters */
  3 SecurityParmsPtr pointer,       /* Address of MQCSP structure for
                                       security parameters */
```

Deklarace High Level Assembler

```
MQCNO                   DSECT
MQCNO_STRUCID           DS   CL4    Structure identifier
MQCNO_VERSION           DS   F      Structure version number
MQCNO_OPTIONS           DS   F      Options that control the action of
*                                   MQCONNX
MQCNO_CLIENTCONNOFFSET  DS   F      Offset of MQCD structure for client
*                                   connection
MQCNO_CLIENTCONNPTR     DS   F      Address of MQCD structure for client
*                                   connection
MQCNO_CONNTAG           DS   XL128  Queue-manager connection tag
*
MQCNO_CONNECTIONID      DS   XL24   Unique connection identifier
*
MQCNO_SSLCONFIGOFFSET   DS   F      Offset of MQCSP structure for security
*                                   parameters
MQCNO_SSLCONFIGPTR      DS   F      Address of MQCSP structure for security
*                                   parameters
MQCNO_LENGTH            EQU  *-MQCNO
                        ORG  MQCNO
MQCNO_AREA              DS   CL(MQCNO_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQCNO
  StrucId          As String*4  'Structure identifier'
  Version          As Long      'Structure version number'
  Options          As Long      'Options that control the action of'
                                'MQCONNX'
  ClientConnOffset As Long      'Offset of MQCD structure for client'
                                'connection'
  ClientConnPtr    As MQPTR     'Address of MQCD structure for client'
                                'connection'
```

```
   ConnTag          As MQBYTE128 'Queue-manager connection tag'
   SSLConfigPtr     As MQPTR     'Address of MQSCO structure for client'
                                 'connection'
   SSLConfigOffset  As Long      'Offset of MQSCO structure for client'
                                 'connection'
   ConnectionId     As MQBYTE24  'Unique connection identifier'
   SecurityParmsOffset As Long   'Offset of MQCSP structure for security'
                                 'parameters'
   SecurityParmsPtr As MQPTR     'Address of MQCSP structure for security'
                                 'parameters'
   End Type
```

MQCSP-parametry zabezpečení

Následující tabulka shrnuje pole ve struktuře.

Tabulka 489. Pole v MQCSP		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
AuthenticationType	Typ ověření	AuthenticationType
Reserved1	Požadováno pro zarovnání ukazatele v IBM i	Reserved1
CSPUserIdPtr	Adresa ID uživatele	CSPUserIdPtr
CSPUserIdOffset	Offset ID uživatele	CSPUserIdOffset
CSPUserIdLength	Délka ID uživatele	CSPUserIdDélka
Reserved2	Požadováno pro zarovnání ukazatele v IBM i	Reserved2
CSPPasswordPtr	Adresa hesla	CSPPasswordPtr
CSPPasswordOffset	Posunutí hesla	CSPPasswordOffset
CSPPasswordLength	Délka hesla	CSPPasswordLength

Přehled pro MQCSP

Dostupnost: Všechny produkty WebSphere MQ .

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 ve struktuře MQCSP musí být ve znakové sadě a kódování lokálního správce front; tyto údaje jsou dány atributem správce front *CodedCharSetId* a MQENC_NATIVE.

Pole pro MQCSP

Struktura MQCSP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

CSPPasswordLength (MQLONG)
Toto pole je délka hesla, které se má použít při ověření.

Maximální délka hesla je závislá na platformě, viz ID uživatelů. 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.

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.

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.

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.

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.

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.

Reserved1 (MQBYTE4)
Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

Reserved2 (MQBYTE8)
Vyhrazené pole, které je povinné pro zarovnání ukazatele na IBM i.

Toto je vstupní pole. Počáteční hodnota tohoto pole má hodnotu null.

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.

Počáteční hodnoty a jazyková deklarace pro MQCSP

Tabulka 490. Počáteční hodnoty polí v MQCSP pro MQCSP

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQCSP_STRUCT	'CSP¬'
Version	MQCSP_VERSION_1	1
AuthenticationType	Není	MQCSP_AUTH_NONE
Reserved1	Není	Nulový řetězec nebo prázdné znaky
CSPUserIdPtr	Není	Nulový ukazatel nebo bajty null
CSPUserIdOffset	Není	0
CSPUserIdLength	Není	0
Reserved2	Není	Nulový řetězec nebo prázdné znaky
CSPPasswordPtr	Není	Nulový ukazatel nebo bajty null
CSPPasswordOffset	Není	0
CSPPasswordLength	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQCSP_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCSP MyCSP = {MQCSP_DEFAULT};
```

Deklarace C

```
typedef struct tagMQCSP MQCSP;
struct tagMQCSP {
  MQCHAR4    StrucId;            /* Structure identifier */
  MQLONG     Version;            /* Structure version number */
  MQLONG     AuthenticationType; /* Type of authentication */
  MQBYTE4    Reserved1;          /* Required for IBM i pointer
                                    alignment */
  MQPTR      CSPUserIdPtr;       /* Address of user ID */
```

```
  MQLONG     CSPUserIdOffset;    /* Offset of user ID */
  MQLONG     CSPUserIdLength;    /* Length of user ID */
  MQBYTE8    Reserved2;          /* Required for IBM i pointer
                                    alignment */
                                 alignment */
  MQPTR      CSPPasswordPtr;     /* Address of password */
  MQLONG     CSPPasswordOffset;  /* Offset of password */
  MQLONG     CSPPasswordLength;  /* Length of password */
  };
```

Deklarace COBOL

```
**   MQCSP structure
  10 MQCSP.
**    Structure identifier
   15 MQCSP-STRUCID           PIC X(4).
**    Structure version number
   15 MQCSP-VERSION           PIC S9(9) BINARY.
**    Type of authentication
   15 MQCSP-AUTHENTICATIONTYPE PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
   15 MQCSP-RESERVED1         PIC X(4).
**    Address of user ID
   15 MQCSP-CSPUSERIDPTR      POINTER.
**    Offset of user ID
   15 MQCSP-CSPUSERIDOFFSET   PIC S9(9) BINARY.
**    Length of user ID
   15 MQCSP-CSPUSERIDLENGTH   PIC S9(9) BINARY.
**    Required for IBM i pointer alignment
   15 MQCSP-RESERVED2         PIC X(4).
**    Address of password
   15 MQCSP-CSPPASSWORDPTR    POINTER.
**    Offset of password
   15 MQCSP-CSPPASSWORDOFFSET PIC S9(9) BINARY.
**    Length of password
   15 MQCSP-CSPPASSWORDLENGTH PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQCSP based,
  3 StrucId           char(4),       /* Structure identifier */
  3 Version           fixed bin(31), /* Structure version number */
  3 AuthenticationType fixed bin(31),  /* Type of authentication */
  3 Reserved1         char(4),     /* Required for IBM i pointer
                                      alignment */
  3 CSPUserIdPtr      pointer,      /* Address of user ID */
  3 CSPUserIdOffset   fixed bin(31), /* Offset of user ID */
  3 CSPUserIdLength   fixed bin(31), /* Length of user ID */
  3 Reserved2         char(8),     /* Required for IBM i pointer
                                      alignment */
  3 CSPPasswordPtr    pointer,      /* Address of password */
  3 CSPPasswordOffset fixed bin(31), /* Offset of user ID */
  3 CSPPasswordLength fixed bin(31); /* Length of user ID */
```

Deklarace jazyka Visual Basic

```
Type MQCSP
  StrucId           As String*4   'Structure identifier'
  Version           As Long       'Structure version number'
  AuthenticationType As Long      'Type of authentication'
  Reserved1         As MQBYTE4    'Required for IBM i pointer'
                                  'alignment'
  CSPUserIdPtr      As MQPTR      'Address of user ID'
  CSPUserIdOffset   As Long       'Offset of user ID'
  CSPUserIdLength   As Long       'Length of user ID'
  Reserved2         As MQBYTE8    'Required for IBM i pointer'
                                  'alignment'
  CSPPasswordPtr    As MQPTR      'Address of password'
  CSPPasswordOffset As Long       'Offset of password'
  CSPPasswordLength As Long       'Length of password'
End Type
```

MQCTLO-Struktura voleb zpětného volání řídicího prvku

Následující tabulka shrnuje pole ve struktuře. Struktura určující funkci zpětného volání řízení.

Tabulka 491. Pole v MQCTLO		
Pole	**Popis**	**Téma**
StrucID	Identifikátor struktury	StrucID
Version	Číslo verze struktury	verze
Options	Volby	Volby
Reserved	Vyhrazené pole	Volby
ConnectionArea	Pole pro funkci zpětného volání k použití	ConnectionArea

Přehled pro MQCTLO

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSa klienti WebSphere MQ MQI připojené k těmto systémům. Přehled struktury MQCTLO.

Účel: Struktura MQCTLO se používá k určení voleb souvisejících s funkcí zpětného volání řízení.

Struktura je vstupním a výstupním parametrem na volání “MQCTL-Řízení zpětných volání” na stránce 635 .

Verze: Aktuální verze MQCTLO je MQCTLO_VERSION_1.

Znaková sada a kódování: Data v aplikaci MQCTLO musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front poskytnutého funkcí MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQCTLO

Abecední seznam polí pro strukturu MQCTLO.

Struktura MQCTLO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

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 nezakládá žádná rozhodnutí založená na obsahu tohoto pole a je předávána v nezměněné podobě do pole “ConnectionArea (MQPTR)” na stránce 260 ve struktuře MQCBC, což je vstupní parametr zpětného 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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQCTLO
Struktura řídicích voleb-počáteční hodnoty

Tabulka 492. Počáteční hodnoty polí v MQCTLO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQCTLO_STRUCTION_ID	'CTLO'
Version	MQCTLO_VERSION_1	1
Options	MQCTLO_NONE	Hodnoty null
Reserved	Vyhrazené pole	
ConnectionArea	Není	Nulový ukazatel nebo bajty null

Tabulka 492. Počáteční hodnoty polí v MQCTLO (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**

Notes:

1. V programovacím jazyce C-proměnná makraHodnota MQCTLO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQCTLO MyCTLO = {MQCTLO_DEFAULT};
```

Deklarace C
Struktura řídicích voleb-deklarace jazyka C

```
typedef struct tagMQCTLO MQCTLO;
 struct tagMQCTLO {
  MQCHAR4   StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQLONG    Options;        /* Options that control the action of MQCTL */
  MQLONG    Reserved;       /* Reserved field */


MQPTR     ConnectionArea; /* Connection work area passed to the function */
 };
```

Deklarace COBOL

```
** MQCTLO structure
 10  MQCTLO.
** Structure Identifier
 15  MQCTLO-STRUCID                    PIC X(4).
** Structure Version
 15  MQCTLO-VERSION                    PIC S9(9) BINARY.
 ** Options
 15  MQCTLO-OPTIONS                    PIC S9(9) BINARY.
** Reserved
 15  MQCTLO-RESERVED                   PIC S9(9) BINARY.
** ConnectionArea
 15  MQCTLO-CONNECTIONAREA             POINTER
```

Deklarace PL/I

```
dcl
 1 MQCTLO based,
  3 StrucId             char(4),       /* Structure identifier */
  3 Version             fixed bin(31), /* Structure version */
  3 Options             fixed bin(31), /* Options */
  3 Reserved            fixed bin(31),
  3 ConnectionArea      pointer;       /* Connection work area */
```

MQDH-záhlaví distribuce

Následující tabulka shrnuje pole ve struktuře.

Tabulka 493. Pole v MQDH		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQDH plus následující záznamy	StrucLength

Tabulka 493. Pole v MQDH (pokračování)		
Pole	**Popis**	**Téma**
Encoding	Číselné kódování dat, která jsou uvedena v poli záznamů MQPMR	Kódování
CodedCharSetId	Identifikátor znakové sady dat, která následuje pole záznamů MQPMR	CodedCharSetId
Format	Název formátu dat, který následuje pole záznamů MQPMR	Formát
Flags	Obecné příznaky	Příznaky
PutMsgRecFields	Příznaky určující, která pole MQPMR jsou přítomna	PutMsgRecFields
RecsPresent	Počet přítomných záznamů objektů	RecsPresent
ObjectRecOffset	Odstup prvního záznamu objektu od začátku zařízení MQDH	PosunutíObjectRec
PutMsgRecOffset	Odstup prvního záznamu vložení-zprávy od začátku zařízení MQDH	PutMsgRecOffset

Přehled pro MQDH

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Okna, plus klienti WebSphere MQ připojené k těmto systémům.

Účel: Struktura MQDH popisuje další data, která se nacházejí ve zprávě, když se jedná o zprávu rozdělovníku uloženou v přenosové frontě. 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.

Tuto strukturu používají specializované aplikace, které vložila zprávy přímo do přenosových front, nebo které odebírají zprávy z přenosových front (například: agenti kanálů zpráv).

Aplikace, které chtějí vložit zprávy do distribučních seznamů, nesmí používat tuto strukturu. Místo toho musí použít strukturu MQOD k definování cílů v distribučním seznamu a struktury MQPMO pro uvedení vlastností zpráv nebo příjmu informací o zprávách odeslaných do jednotlivých míst určení.

Název formátu: MQFMT_DIST_HEADER.

Znaková sada a kódování: Data ve znakové sadě MQDH musí být ve znakové sadě atributu správce front produktu *CodedCharSetId* a kódování lokálního správce front daného parametrem MQENC_NATIVE.

Nastavte znakovou sadu a kódování objektu MQDH do polí *CodedCharSetId* a *Encoding* v následujícím umístění:

- 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 generovat více než jednu takovou zprávu a umístit ji do různých přenosových front. V takovém případě struktury MQDH v těchto zprávách identifikují různé podmnožiny cílů definovaných v seznamu distribucí otevřeném aplikací.

Aplikace, která umístí zprávu do distribuční fronty přímo do přenosové fronty, musí odpovídat výše popsané posloupnosti a musí zajistit, aby struktura MQDH byla správná. Pokud struktura MQDH není platná, může správce front selhat při volání MQPUT nebo MQPUT1 s kódem příčiny MQRC_DH_ERROR.

Zprávy ve frontě můžete uložit do fronty v podobě distribučního seznamu pouze v případě, že jste frontu definovali jako schopnou podporovat zprávy distribučního seznamu (viz atribut fronty *DistLists* popsaný v části "Atributy pro fronty" na stránce 787). 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 pro MQDH

Struktura MQDH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ , kteří jsou připojeni k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

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.

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 podmínky 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.

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.

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.

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.

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.

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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQDH

Tabulka 494. Počáteční hodnoty polí v MQDH pro MQDH

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQDH_	'DH¬¬'

Tabulka 494. Počáteční hodnoty polí v MQDH pro MQDH (pokračování)

Název pole	Název konstanty	Hodnota konstanty
Version	MQDH_VERSION_1	1
StrucLength	Není	0
Encoding	Není	0
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQDHF_NONE	0
PutMsgRecFields	MQPMRF_NONE	0
RecsPresent	Není	0
ObjectRecOffset	Není	0
PutMsgRecOffset	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQDH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDH MyDH = {MQDH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQDH MQDH;
struct tagMQDH {
  MQCHAR4  StrucId;          /* Structure identifier */
  MQLONG   Version;          /* Structure version number */
  MQLONG   StrucLength;      /* Length of MQDH structure plus following
                                MQOR and MQPMR records */
  MQLONG   Encoding;         /* Numeric encoding of data that follows
                                the MQOR and MQPMR records */
  MQLONG   CodedCharSetId;   /* Character set identifier of data that
                                follows the MQOR and MQPMR records */
  MQCHAR8  Format;           /* Format name of data that follows the
                                MQOR and MQPMR records */
  MQLONG   Flags;            /* General flags */
  MQLONG   PutMsgRecFields;  /* Flags indicating which MQPMR fields are
                                present */
  MQLONG   RecsPresent;      /* Number of MQOR records present */
  MQLONG   ObjectRecOffset;  /* Offset of first MQOR record from start
                                of MQDH */
  MQLONG   PutMsgRecOffset;  /* Offset of first MQPMR record from start
                                of MQDH */
};
```

Deklarace COBOL

```
**   MQDH structure
  10 MQDH.
**    Structure identifier
   15 MQDH-STRUCID         PIC X(4).
**    Structure version number
   15 MQDH-VERSION         PIC S9(9) BINARY.
**    Length of MQDH structure plus following MQOR and MQPMR records
   15 MQDH-STRUCLENGTH     PIC S9(9) BINARY.
**    Numeric encoding of data that follows the MQOR and MQPMR records
   15 MQDH-ENCODING        PIC S9(9) BINARY.
**    Character set identifier of data that follows the MQOR and MQPMR
```

```
**    records
   15 MQDH-CODEDCHARSETID  PIC S9(9) BINARY.
**    Format name of data that follows the MQOR and MQPMR records
   15 MQDH-FORMAT          PIC X(8).
**    General flags
   15 MQDH-FLAGS           PIC S9(9) BINARY.
**    Flags indicating which MQPMR fields are present
   15 MQDH-PUTMSGRECFIELDS PIC S9(9) BINARY.
**    Number of MQOR records present
   15 MQDH-RECSPRESENT     PIC S9(9) BINARY.
**    Offset of first MQOR record from start of MQDH
   15 MQDH-OBJECTRECOFFSET PIC S9(9) BINARY.
**    Offset of first MQPMR record from start of MQDH
   15 MQDH-PUTMSGRECOFFSET PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQDH based,
  3 StrucId         char(4),       /* Structure identifier */
  3 Version         fixed bin(31), /* Structure version number */
  3 StrucLength     fixed bin(31), /* Length of MQDH structure plus
                                      following MQOR and MQPMR
                                      records */
  3 Encoding        fixed bin(31), /* Numeric encoding of data that
                                      follows the MQOR and MQPMR
                                      records */
  3 CodedCharSetId  fixed bin(31), /* Character set identifier of data
                                      that follows the MQOR and MQPMR
                                      records */
  3 Format          char(8),       /* Format name of data that follows
                                      the MQOR and MQPMR records */
  3 Flags           fixed bin(31), /* General flags */
  3 PutMsgRecFields fixed bin(31), /* Flags indicating which MQPMR
                                      fields are present */
  3 RecsPresent     fixed bin(31), /* Number of MQOR records present */
  3 ObjectRecOffset fixed bin(31), /* Offset of first MQOR record from
                                      start of MQDH */
  3 PutMsgRecOffset fixed bin(31); /* Offset of first MQPMR record from
                                      start of MQDH */
```

Deklarace jazyka Visual Basic

```
Type MQDH
  StrucId         As String*4 'Structure identifier'
  Version         As Long     'Structure version number'
  StrucLength     As Long     'Length of MQDH structure plus following'
                              'MQOR and MQPMR records'
  Encoding        As Long     'Numeric encoding of data that follows'
                              'the MQOR and MQPMR records'
  CodedCharSetId  As Long     'Character set identifier of data that'
                              'follows the MQOR and MQPMR records'
  Format          As String*8 'Format name of data that follows the'
                              'MQOR and MQPMR records'
  Flags           As Long     'General flags'
  PutMsgRecFields As Long     'Flags indicating which MQPMR fields are'
                              'present'
  RecsPresent     As Long     'Number of MQOR records present'
  ObjectRecOffset As Long     'Offset of first MQOR record from start'
                              'of MQDH'
  PutMsgRecOffset As Long     'Offset of first MQPMR record from start'
                              'of MQDH'
End Type
```

Záhlaví MQDLH-Dead-letter

Následující tabulka shrnuje pole ve struktuře.

Tabulka 495. Pole v MQDLH		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Reason	Byla doručena zpráva o příčině na frontě nedoručených zpráv.	Příčina
DestQName	Název původní cílové fronty	DestQName
DestQMgrName	Název původního cílového správce front	DestQMgrName
Encoding	Číselné kódování dat, která následuje za MQDLH	Kódování
CodedCharSetId	Identifikátor znakové sady, která následuje za MQDLH	CodedCharSetId
Format	Formátovat název dat, která následuje za MQDLH	Formát
PutApplType	Typ aplikace, která vložila zprávu do fronty nedoručených zpráv	PutApplType
PutApplName	Název aplikace, která vložila zprávu do fronty nedoručených zpráv	PutApplName
PutDate	Datum, kdy byla zpráva vložena do fronty nedoručených zpráv	PutDate
PutTime	Čas, kdy byla zpráva vložena do fronty nedoručených zpráv	PutTime

Přehled pro MQDLH

Dostupnost: Všechny platformy WebSphere MQ .

Účel: Struktura MQDLH popisuje informace, které deřadí data zpráv aplikací ve frontě nedoručených zpráv (undelivered-message). Zpráva může být doručena do fronty nedoručených zpráv, protože správce front nebo agent kanálu zpráv jej přesměroval do fronty, nebo protože aplikace zadala zprávu přímo do fronty.

Název formátu: MQFMT_DEAD_LETTER_HEADER.

Znaková sada a kódování: Pole ve struktuře MQDLH se nacházejí ve znakové sadě a kódování zadané v polích *CodedCharSetId* a *Encoding* . Tyto hodnoty 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, které jsou platné v názvech front.

Pokud používáte třídy WMQ pro prostředí Java/JMS a kódová stránka definovaná v deskriptoru MQMD není podporována virtuálním počítačem Java, bude MQDLH zapsáno ve znakové sadě UTF-8 .

Použití: Aplikace, které vložila zprávy přímo do fronty nedoručených zpráv, musí obsahovat předponu dat zprávy se strukturou MQDLH a inicializovat pole s příslušnými hodnotami. Správce front však nevyžaduje, aby byla přítomna struktura MQDLH nebo že pro pole byly zadány platné hodnoty.

Pokud je zpráva příliš dlouhá na vložení do fronty nedoručených zpráv, musí aplikace provést jednu z následujících možností:

- Ořízněte data zprávy tak, aby se vešly do fronty nedoručených zpráv.
- Zaznamenejte zprávu do pomocné paměti a umístěte zprávu o výjimce do fronty nedoručených zpráv, která bude označovat toto.

- Vyřazovat zprávu a vrátit chybu původci. Je-li zpráva (nebo může být) kritická zpráva, udělejte to pouze tehdy, je-li známo, že původce stále má kopii zprávy; například zpráva přijatá agentem kanálu zpráv z komunikačního kanálu.

Který z výše uvedených hodnot je vhodný (je-li nějaký) závisí na návrhu aplikace.

Správce front provádí speciální zpracování, pokud je zpráva, která je segmentem, vložena se strukturou MQDLH na přední straně; viz popis struktury MQMDE pro další podrobnosti.

Vložení zpráv do fronty nedoručených zpráv: Je-li zpráva vložena do fronty nedoručených zpráv, musí být struktura MQMD použitá pro volání MQPUT nebo MQPUT1 identická s názvem MQMD asociovaným se zprávou (obvykle MQMD, který je vrácen voláním MQGET), s výjimkou následujících položek:

- Nastavte pole *CodedCharSetId* a *Encoding* na jakoukoli znakovou sadu a kódování se používají pro pole ve struktuře MQDLH.
- Chcete-li označit, že data začínají strukturou MQDLH, nastavte pole *Format* na hodnotu MQFMT_DEAD_LETTER_HEADER.
- Nastavte pole kontextu (*AccountingToken*, *ApplIdentityData* , *ApplOriginData*, *PutApplName*, *PutApplType* , *PutDate*, *PutTime*, *UserIdentifier*) pomocí kontextové volby, která odpovídá okolnostem:
  - Aplikace, která vkládá 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 zpráv na jejich výchozí hodnoty.
  - Serverová aplikace uváděná do fronty nedoručených zpráv, která právě přijala, musí použít volbu MQPMO_PASS_ALL_CONTEXT k zachování původních kontextových informací.
  - Serverová aplikace uváděná do fronty nedoručených zpráv *reply* na zprávu, kterou právě obdržela, musí používat volbu MQPMO_PASS_IDENTITY_CONTEXT; to zachová informace o identitě, ale nastaví informace o původu tak, aby to bylo v aplikaci serveru.
  - Agent oznamovacího kanálu, který vloží do fronty nedoručených zpráv zprávu, kterou obdrží z komunikačního kanálu, musí použít volbu MQPMO_SET_ALL_CONTEXT k zachování původních kontextových informací.

V samotné struktuře MQDLH nastavte pole takto:

- Nastavte pole *CodedCharSetId*, *Encoding*a *Format* na hodnoty, které popisují data, která následují za strukturou MQDLH, obvykle hodnoty z původního deskriptoru zpráv.
- Nastavte pole kontextu *PutApplType*, *PutApplName*, *PutDate*a *PutTime* na hodnoty odpovídající aplikaci, která vkládá zprávu do fronty nedoručených zpráv. Tyto hodnoty se nevztahují k původní zprávě.
- Podle potřeby nastavte jiná pole.

Ujistěte se, že všechna pole mají platné hodnoty a že znaková pole jsou doplněna mezerami do definované délky pole; neukončujte data znaků předčasně pomocí znaku hex 00, protože správce front nekonvertuje null a následné znaky na mezery ve struktuře MQDLH.

Získávání zpráv z fronty nedoručených zpráv: Aplikace, které získávají zprávy z fronty nedoručených zpráv, musí ověřit, zda zprávy začínají strukturou MQDLH. Aplikace může určit, zda je struktura MQDLH přítomna tak, že prozkoumá pole *Format* v deskriptoru zprávy MQMD; má-li pole hodnotu MQFMT_DEAD_LETTER_HEADER, data zprávy začínají strukturou MQDLH. Počítejte také s tím, že zprávy, které aplikace získají z fronty nedoručených zpráv, mohou být zkráceny, pokud byly pro frontu původně příliš dlouhé.

Pole pro MQDLH

Struktura MQDLH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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, HP-UX, z/OS, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ MQI, kteří jsou připojeni k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

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.

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.

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.

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.

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 zprávy” na stránce 383.

Pokud správce front přesměrovává zprávu do fronty nedoručených zpráv, *PutApplName* obsahuje prvních 28 znaků názvu správce front, 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.

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 zprávy" na stránce 383).

Pokud správce front přesměrovává zprávu do fronty nedoručených zpráv, bude mít parametr *PutApplType* hodnotu MQAT_QMGR.

Počáteční hodnota tohoto pole je `0`.

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.

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 zprávy” na stránce 383 .

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 557).

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 557).

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.

Počáteční hodnota tohoto pole je MQRC_NONE.

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.

Počáteční hodnoty a deklarace jazyka pro MQDLH

Tabulka 496. Počáteční hodnoty polí v MQDLH pro MQDLH

Název pole	**Název konstanty**	**Hodnota konstanty**
StrucId	ID_STRUKTURY MQDLH_STRUCTION_ID	'DLH¬'
Version	MQDLH_VERSION_1	1
Reason	MQRC_NONE	0
DestQName	Není	Nulový řetězec nebo prázdné znaky
DestQMgrName	Není	Nulový řetězec nebo prázdné znaky
Encoding	Není	0
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
PutApplType	Není	0
PutApplName	Není	Nulový řetězec nebo prázdné znaky
PutDate	Není	Nulový řetězec nebo prázdné znaky
PutTime	Není	Nulový řetězec nebo prázdné znaky

Tabulka 496. Počáteční hodnoty polí v MQDLH pro MQDLH (pokračování)

Název pole	Název konstanty	Hodnota konstanty

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.
3. V programovacím jazyce C-proměnná makraParametr MQDLH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDLH MyDLH = {MQDLH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQDLH MQDLH;
struct tagMQDLH {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Reason;          /* Reason message arrived on dead-letter
                                (undelivered-message) queue */
  MQCHAR48  DestQName;       /* Name of original destination queue */
  MQCHAR48  DestQMgrName;    /* Name of original destination queue
                                manager */
  MQLONG    Encoding;        /* Numeric encoding of data that follows
                                MQDLH */
  MQLONG    CodedCharSetId;  /* Character set identifier of data that
                                follows MQDLH */
  MQCHAR8   Format;          /* Format name of data that follows
                                MQDLH */
  MQLONG    PutApplType;     /* Type of application that put message on
                                dead-letter (undelivered-message)
                                queue */
  MQCHAR28  PutApplName;     /* Name of application that put message on
                                dead-letter (undelivered-message)
                                queue */
  MQCHAR8   PutDate;         /* Date when message was put on dead-letter
                                (undelivered-message) queue */
  MQCHAR8   PutTime;         /* Time when message was put on the
                                dead-letter (undelivered-message)
                                queue */
};
```

Deklarace COBOL

```
**   MQDLH structure
  10 MQDLH.
**    Structure identifier
   15 MQDLH-STRUCID        PIC X(4).
**    Structure version number
   15 MQDLH-VERSION        PIC S9(9) BINARY.
**    Reason message arrived on dead-letter (undelivered-message) queue
   15 MQDLH-REASON         PIC S9(9) BINARY.
**    Name of original destination queue
   15 MQDLH-DESTQNAME      PIC X(48).
**    Name of original destination queue manager
   15 MQDLH-DESTQMGRNAME   PIC X(48).
**    Numeric encoding of data that follows MQDLH
   15 MQDLH-ENCODING       PIC S9(9) BINARY.
**    Character set identifier of data that follows MQDLH
   15 MQDLH-CODEDCHARSETID PIC S9(9) BINARY.
**    Format name of data that follows MQDLH
   15 MQDLH-FORMAT         PIC X(8).
**    Type of application that put message on dead-letter
**    (undelivered-message) queue
   15 MQDLH-PUTAPPLTYPE    PIC S9(9) BINARY.
**    Name of application that put message on dead-letter
**    (undelivered-message) queue
   15 MQDLH-PUTAPPLNAME    PIC X(28).
```

```
**    Date when message was put on dead-letter (undelivered-message)
**    queue
   15 MQDLH-PUTDATE        PIC X(8).
**    Time when message was put on the dead-letter (undelivered-message)
**    queue
   15 MQDLH-PUTTIME        PIC X(8).
```

Deklarace PL/I

```
dcl
 1 MQDLH based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 Reason         fixed bin(31), /* Reason message arrived on
                                     dead-letter (undelivered-message)
                                     queue */
  3 DestQName      char(48),      /* Name of original destination
                                     queue */
  3 DestQMgrName   char(48),      /* Name of original destination queue
                                     manager */
  3 Encoding       fixed bin(31), /* Numeric encoding of data that
                                     follows MQDLH */
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                     that follows MQDLH */
  3 Format         char(8),       /* Format name of data that follows
                                     MQDLH */
  3 PutApplType    fixed bin(31), /* Type of application that put
                                     message on dead-letter
                                     (undelivered-message) queue */
  3 PutApplName    char(28),      /* Name of application that put
                                     message on dead-letter
                                     (undelivered-message) queue */
  3 PutDate        char(8),       /* Date when message was put on
                                     dead-letter (undelivered-message)
                                     queue */
  3 PutTime        char(8);       /* Time when message was put on the
                                     dead-letter (undelivered-message)
                                     queue */
```

Deklarace High Level Assembler

```
MQDLH                 DSECT
MQDLH_STRUCID         DS   CL4   Structure identifier
MQDLH_VERSION         DS   F     Structure version number
MQDLH_REASON          DS   F     Reason message arrived on dead-letter
*                                (undelivered-message) queue
MQDLH_DESTQNAME       DS   CL48  Name of original destination queue
MQDLH_DESTQMGRNAME    DS   CL48  Name of original destination queue
*                                manager
MQDLH_ENCODING        DS   F     Numeric encoding of data that follows
*                                MQDLH
MQDLH_CODEDCHARSETID  DS   F     Character set identifier of data that
*                                follows MQDLH
MQDLH_FORMAT          DS   CL8   Format name of data that follows MQDLH
MQDLH_PUTAPPLTYPE     DS   F     Type of application that put message on
*                                dead-letter (undelivered-message) queue
MQDLH_PUTAPPLNAME     DS   CL28  Name of application that put message on
*                                dead-letter (undelivered-message) queue
MQDLH_PUTDATE         DS   CL8   Date when message was put on
*                                dead-letter (undelivered-message) queue
MQDLH_PUTTIME         DS   CL8   Time when message was put on the
*                                dead-letter (undelivered-message) queue
*
MQDLH_LENGTH          EQU  *-MQDLH
                      ORG  MQDLH
MQDLH_AREA            DS   CL(MQDLH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQDLH
  StrucId        As String*4  'Structure identifier'
  Version        As Long      'Structure version number'
  Reason         As Long      'Reason message arrived on dead-letter'
```

```
                              '(undelivered-message) queue'
   DestQName      As String*48 'Name of original destination queue'
   DestQMgrName   As String*48 'Name of original destination queue'
                              'manager'
   Encoding       As Long      'Numeric encoding of data that follows'
                              'MQDLH'
   CodedCharSetId As Long      'Character set identifier of data that'
                              'follows MQDLH'
   Format         As String*8  'Format name of data that follows MQDLH'
   PutApplType    As Long      'Type of application that put message on'
                              'dead-letter (undelivered-message) queue'
   PutApplName    As String*28 'Name of application that put message on'
                              'dead-letter (undelivered-message) queue'
   PutDate        As String*8  'Date when message was put on dead-letter'
                              '(undelivered-message) queue'
   PutTime        As String*8  'Time when message was put on the'
                              'dead-letter (undelivered-message) queue'
End Type
```

MQDMHO-Odstranění voleb zpracování zpráv

Následující tabulka shrnuje pole ve struktuře.

Tabulka 497. Pole v MQDMHO

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby

Přehled pro MQDMHO

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

Účel: Struktura **MQDMHO** umožňuje aplikacím zadávat volby, které řídí způsob odstranění manipulátorů zpráv. Struktura je vstupním parametrem na volání **MQDLTMH** .

Znaková sada a kódování: Data v souboru **MQDMHO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole pro MQDMHO

Struktura MQDMHO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Volby (MQLONG)

Hodnota musí být:

MQDMHO_NONE

Nejsou uvedeny žádné volby.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je **MQDMHO_NONE**.

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**.

Počáteční hodnoty a deklarace jazyka pro MQDMHO

Tabulka 498. Počáteční hodnoty polí v MQDMHO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQDMHO_STRUCTION_ID	'DMHO'
Version	MQDMHO_VERSION_1	1
Options	MQDMHO_NONE	0

Notes:

1. V programovacím jazyce C-proměnná makraHodnota MQDMHO_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDMHO MyDMHO = {MQDMHO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQDMHO;
struct tagMQDMHO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQDLTMH */
};
```

Deklarace COBOL

```
**   MQDMHO structure
  10 MQDMHO.
**    Structure identifier
   15 MQDMHO-STRUCID       PIC X(4).
**    Structure version number
   15 MQDMHO-VERSION       PIC S9(9) BINARY.
**    Options that control the action of MQDLTMH
   15 MQDMHO-OPTIONS       PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQDMHO based,
  3 StrucId        char(4),       /* Structure identifier */
```

```
  3 Version        fixed bin(31), /* Structure version number */
  3 Options        fixed bin(31), /* Options that control the action of MQDLTMH */
```

Deklarace High Level Assembler

```
MQDMHO                DSECT
MQDMHO_STRUCID        DS   CL4   Structure identifier
MQDMHO_VERSION        DS   F     Structure version number
MQDMHO_OPTIONS        DS   F     Options that control the action of
*                                MQDLTMH
MQDMHO_LENGTH         EQU  *-MQDMHO
MQDMHO_AREA           DS   CL(MQDMHO_LENGTH)
```

MQDMPO-Odstranění voleb vlastností zprávy

Následující tabulka shrnuje pole ve struktuře. Struktura MQDMPO struktury-odstranění vlastností vlastností zprávy

Tabulka 499. Pole v MQDMPO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby řízení akce MQDMPO	Volby

Přehled pro MQDMPO

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

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 (MQENC_NATIVE).

Pole pro MQDMPO

Struktura voleb vlastností pro odstranění zprávy-pole

Struktura MQDMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQDMPO

Struktura voleb odstranění vlastností zprávy-počáteční hodnoty

Tabulka 500. Počáteční hodnoty polí v MQDPMO

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_KONSTRUKCE_MQDMPO_	'DMPO'
Version	MQDMPO_VERSION_1	1
Options	Volby, které řídí akci příkazu MQDLTMP	MQDMPO_NONE

Notes:

1. V programovacím jazyce C-proměnná makraHodnota MQDMPO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQDMPO MyDMPO = {MQDMPO_DEFAULT};
```

Deklarace C
Struktura volby odstranění vlastností zprávy-deklarace jazyka C

```
typedef struct tagMQDMPO MQDMPO;
struct tagMQDMPO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQDLTMP */
};
```

Deklarace COBOL
Struktura voleb pro odstranění vlastností zprávy-deklarace jazyka COBOL

```
**   MQDMPO structure
  10 MQDMPO.
**    Structure identifier
   15 MQDMPO-STRUCID               PIC X(4).
**    Structure version number
   15 MQDMPO-VERSION               PIC S9(9) BINARY.
**    Options that control the action of MQDLTMP
   15 MQDMPO-OPTIONS               PIC S9(9) BINARY.
```

Deklarace PL/I
Struktura volby odstranění vlastností zprávy-deklarace jazyka PL/I

```
Dcl
 1 MQDPMO based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 Options        fixed bin(31), /* Options that control the action
                                     of MQDLTMP */
```

Deklarace High Level Assembler
Struktura volby odstranění vlastností zprávy-deklarace jazyka asembleru

```
MQDMPO                    DSECT
MQDMPO_STRUCID            DS   CL4  Structure identifier
MQDMPO_VERSION            DS   F    Structure version number
MQDMPO_OPTIONS            DS   F    Options that control the
*                                   action of MQDLTMP
MQDMPO_LENGTH             EQU  *-MQDMPO
MQDMPO_AREA               DS   CL(MQDMPO_LENGTH)
```

MQEPH-záhlaví vloženého PCF

Následující tabulka shrnuje pole ve struktuře.

Tabulka 501. Pole v MQEPH

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQEPH plus struktury MQCFH a struktury parametrů, které za ním následují	StrucLength
Encoding	Číselné kódování dat za poslední strukturu parametru PCF	Kódování
CodedCharSetId	Identifikátor znakové sady, která následuje poslední strukturu parametru PCF	CodedCharSetId
Format	Název formátu dat, který následuje poslední strukturu parametru PCF	Formát
Flags	Příznaky	Příznaky
PCFHeader	Hlavička Programovatelného příkazového formátu (PCF)	Záhlaví PCFHeader

Přehled pro MQEPH

Dostupnost: Všechny platformy WebSphere MQ .

Účel: Struktura MQEPH popisuje další data, která se vyskytují ve zprávě, když je tato zpráva programovatelná zpráva ve formátu příkazu (PCF). Pole *PCFHeader* definuje parametry PCF, které následují za touto strukturou, a to vám umožňuje sledovat data zprávy PCF s ostatními záhlavími.

Název formátu: MQFMT_EMBEDDED_PCF

Znaková sada a kódování: Data v aplikaci MQEPH musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE.

Nastavte znakovou sadu a kódování MQEPH do polí *CodedCharSetId* a *Encoding* 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 pro jakýkoli 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 pro MQEPH

Struktura MQEPH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

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.

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.

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 502. Počáteční hodnoty polí v MQCFH		
Název pole	**Název konstanty**	**Hodnota konstanty**
Type	MQCFT_NONE	0
StrucLength	MQCFH_STRU_LENGH	36
Version	MQCFH_VERSION_3	3
StrucLength	Není	0
Command	MQCMD_NONE	0
MsgSeqNumber	Není	1
Control	MQCFC_LAST	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
ParameterCount	Není	0

Aplikace musí změnit Type z MQCFT_NONE na platný typ struktury pro použití vloženého záhlaví PCF.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQEPH

Tabulka 503. Počáteční hodnoty polí v MQEPH pro MQEPH

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_KONSTRUKCE_MQEPH_	'EPH¬'
Version	MQEPH_VERSION_1	1
StrucLength	SPRAVA_STRUKTUROVANÉ_STRUKTUROVANÉHO_SYSTÉMU	68
Encoding	Není	0
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQEPH_NONE	0
PCFHeader	Názvy a hodnoty, jak jsou definovány v produktu Tabulka 502 na stránce 334	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQEPH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQEPH MyEPH = {MQEPH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQEPH MQEPH;
struct tagMQDH {
  MQCHAR4  StrucId;          /* Structure identifier */
  MQLONG   Version;          /* Structure version number */
  MQLONG   StrucLength;      /* Total length of MQEPH including the MQCFH
                                and parameter structures that follow it */
  MQLONG   Encoding;         /* Numeric encoding of data that follows last
                                PCF parameter structure */
  MQLONG   CodedCharSetId;   /* Character set identifier of data that
                                follows last PCF parameter structure */
  MQCHAR8  Format;           /* Format name of data that follows last PCF
                                parameter structure */
  MQLONG   Flags;            /* Flags */
  MQCFH    PCFHeader;        /* Programmable command format header */
 };
```

Deklarace COBOL

```
**   MQEPH structure
  10 MQEPH.
**    Structure identifier
   15 MQEPH-STRUCID         PIC X(4).
**    Structure version number
   15 MQEPH-VERSION         PIC S9(9) BINARY.
**    Total length of MQEPH structure including the MQCFH
**    and parameter structures that follow it
   15 MQEPH-STRUCLENGTH     PIC S9(9) BINARY.
**    Numeric encoding of data that follows last
**    PCF structure
   15 MQEPH-ENCODING        PIC S9(9) BINARY.
**    Character set identifier of data that
**    follows last PCF parameter structure
   15 MQEPH-CODEDCHARSETID  PIC S9(9) BINARY.
**    Format name of data that follows last PCF
```

```
**    parameter structure
   15 MQEPH-FORMAT          PIC X(8).
**    Flags
   15 MQEPH-FLAGS           PIC S9(9) BINARY.
**    Programmable command format header
   15 MQEPH-PCFHEADER.
**      Structure type
     20 MQEPH-PCFHEADER-TYPE           PIC S9(9) BINARY.
**      Structure length
     20 MQEPH-PCFHEADER-STRUCLENGTH    PIC S9(9) BINARY.
**      Structure version number
     20 MQEPH-PCFHEADER-VERSION        PIC S9(9) BINARY.
**      Command identifier
     20 MQEPH-PCFHEADER-COMMAND        PIC S9(9) BINARY.
**      Message sequence number
     20 MQEPH-PCFHEADER-MSGSEQNUMBER   PIC S9(9) BINARY.
**      Control options
     20 MQEPH-PCFHEADER-CONTROL        PIC S9(9) BINARY.
**      Completion code
     20 MQEPH-PCFHEADER-COMPCODE       PIC S9(9) BINARY.
**      Reason code qualifying completion code
     20 MQEPH-PCFHEADER-REASON         PIC S9(9) BINARY.
**      Count of parameter structures
     20 MQEPH-PCFHEADER-PARAMETERCOUNT PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQEPH based,
  3 StrucId         char(4),       /* Structure identifier */
  3 Version         fixed bin(31), /* Structure version number */
  3 StrucLength     fixed bin(31), /* Total Length of MQEPH including the
                                      MQCFH and parameter structures that
                                      follow it
  3 Encoding        fixed bin(31), /* Numeric encoding of data that follows
                                      last PCF parameter structure
  3 CodedCharSetId  fixed bin(31), /* Character set identifier of data that
                                      follows last PCF parameter structure
  3 Format          char(8),       /* Format name of data that follows last
                                      PCF parameter structure */
  3 Flags           fixed bin(31), /* Flags */
  3 PCFHeader,                     /* Programmable command format header
   5 Type           fixed bin(31), /* Structure type */
   5 StrucLength    fixed bin(31), /* Structure length */
   5 Version        fixed bin(31), /* Structure version number */
   5 Command        fixed bin(31), /* Command identifier */
   5 MsgseqNumber   fixed bin(31), /* Message sequence number */
   5 Control        fixed bin(31), /* Control options */
   5 CompCode       fixed bin(31), /* Completion code */
   5 Reason         fixed bin(31), /* Reason code qualifying completion code */
   5 ParameterCount fixed bin(31); /* Count of parameter structures */
```

Deklarace High Level Assembler

```
MQEPH                          DSECT
MQEPH_STRUCID                  DS   CL4   Structure identifier
MQEPH_VERSION                  DS   F     Structure version number
MQEPH_STRUCLENGTH              DS   F     Total length of MQEPH including the
*                                         MQCFH and parameter structures that
                                          follow it
MQEPH_ENCODING                 DS   F     Numeric encoding of data that follows
*                                         last PCF parameter structure
MQEPH_CODEDCHARSETID           DS   F     Character set identifier of data that
*                                         follows last PCF parameter structure
MQEPH_FORMAT                   DS   CL8   Format name of data that follows last
*                                         PCF parameter structure
MQEPH_FLAGS                    DS   F     Flags
MQEPH_PCFHEADER                DS   0F    Force fullword alignment
MQEPH_PCFHEADER_TYPE           DS   F     Structure type
MQEPH_PCFHEADER_STRUCLENGTH    DS   F     Structure length
MQEPH_PCFHEADER_VERSION        DS   F     Structure version number
MQEPH_PCFHEADER_COMMAND        DS   F     Command identifier
MQEPH_PCFHEADER_MSGSEQNUMBER   DS   F     Structure length
MQEPH_PCFHEADER_CONTROL        DS   F     Control options
```

```
MQEPH_PCFHEADER_COMPCODE         DS   F     Completion code
MQEPH_PCFHEADER_REASON           DS   F     Reason code qualifying completion code
MQEPH_PCFHEADER_PARAMETER_COUNT  DS   F     Count of parameter structures
MQEPH_PCFHEADER_LENGTH           EQU  *-MQEPH_PCFHEADER
                                 ORG  MQEPH_PCFHEADER
MQEPH_PCFHEADER_AREA             DS   CL(MQEPH_PCFHEADER_LENGTH)
*
MQEPH_LENGTH                     EQU  *-MQEPH
                                 ORG  MQEPH
MQEPH_AREA                       DS   CL(MQEPH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQEPH
  StrucId         As String*4 'Structure identifier'
  Version         As Long     'Structure version number'
  StrucLength     As Long     'Total length of MQEPH structure including the MQCFH'
                              'and parameter structures that follow it'
  Encoding        As Long     'Numeric encoding of data that follows last'
                              'PCF parameter structure'
  CodedCharSetId  As Long     'Character set identifier of data that'
                              'follows last PCF parameter structure'
  Format          As String*8 'Format name of data that follows last PCF'
                              'parameter structure'
  Flags           As Long     'Flags'
  PCFHeader       As MQCFH    'Programmable command format header'
 End Type

Global MQEPH_DEFAULT As MQEPH
```

MQGMO-Získat-volby zprávy

Následující tabulka shrnuje pole ve struktuře.

Tabulka 504. Pole v produktu MQGMO

Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby, které řídí akci MQGET	MQGMO-pole Volby
WaitInterval	Interval čekání	WaitInterval
Signal1	signál	Signal1
Signal2	Identifikátor signálu	Signal2
ResolvedQName	Vyřešený název cílové fronty	ResolvedQName
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQGMO_VERSION_2.		
MatchOptions	Volby, které řídí výběrová kritéria použitá pro MQGET	MatchOptions
GroupStatus	Příznak označující, zda je načtená zpráva ve skupině	GroupStatus
SegmentStatus	Příznak označující, zda je načtená zpráva segmentem logické zprávy	SegmentStatus
Segmentation	Příznak označující, zda je pro načtenou zprávu povolena další segmentace	Segmentace
Reserved1	Vyhrazené	Reserved1
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQGMO_VERSION_3.		

Tabulka 504. Pole v produktu MQGMO (pokračování)		
Pole	**Popis**	**Téma**
MsgToken	Token zpráv	MsgToken
ReturnedLength	Délka vrácených dat zprávy (bajty)	ReturnedLength
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQGMO_VERSION_4.		
Reserved2	Vyhrazené	Reserved2
MsgHandle	Manipulátor se zprávou, která má být naplněna vlastnostmi zprávy načítané z fronty.	MsgHandle

Přehled pro MQGMO

Dostupnost: Všechny platformy WebSphere MQ .

Účel: Struktura MQGMO umožňuje aplikaci řídit způsob, jakým jsou zprávy odebírány z front. Struktura je vstupním/výstupním parametrem na volání MQGET.

Verze: Aktuální verze produktu MQGMO je MQGMO_VERSION_4. Některá pole jsou k dispozici pouze v určitých verzích MQGMO. Pokud potřebujete portovat aplikace mezi několika prostředími, musíte se ujistit, že je verze MQGMO konzistentní ve všech prostředích. Pole, která existují pouze v určitých verzích struktury, jsou identifikována jako “MQGMO-Získat-volby zprávy” na stránce 337 a v popisech polí.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi produktu MQGMO, která je podporována prostředím, ale s počáteční hodnotou pole *Version* nastavenou na hodnotu 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 produktu MQGMO musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQGMO

Struktura MQGMO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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 odpovídající volba shody *nejsou* , jsou ignorována při výběru zprávy, která má být

vrácena. 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 použitá kritéria výběru 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 *je* aktuální skupina nebo logická zpráva, lze vrátit pouze další zprávu ve skupině nebo dalšímu segmentu v logické zprávě a tuto změnu nelze změnit zadáním voleb MQMO_ *.

V obou výše uvedených 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 449 .

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 hodnotě *any* identifikátoru zprávy v produktu MQMD pro zprávu. Proto uvedení MQMO_MATCH_MSG_ID s hodnotou MQMI_NONE je stejné jako *není* určující 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 je parametr MQMO_MATCH_CORREL_ID s parametrem MQCI_NONE stejný jako *není* určující položku 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 hodnoty

MQMO_MATCH_GROUP_ID s hodnotou MQGI_NONE je stejné jako *není* určující položku 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 podporována v systému z/OS.

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 definována 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.

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.

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 hodnotu INDXTYPE a na systému 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.

Volby (MQLONG)

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 505. Pravidla pro aktivaci volání MQGET ve sdílené frontě.

Počet volání příkazu MQGET čekajících na aktivaci		**Výsledek**
S volbou BROWSE	**Bez volby BROWSE**[1]	
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:

[1] 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.

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 žádné *MsgId* ani *CorrelId* se neshodují, je volání MQGET znovu vydáno každých 200 milisekund, dokud nevyvolá vhodná zpráva do fronty nebo dokud nevyprší interval čekání.

  Tato metoda způsobí neočekávané zatížení zpracování a není účinnou metodou načítání zpráv, když se zprávy přidávají zřídka. Chcete-li se vyhnout této režii pro případ procházení, zadejte *MsgId* (pokud není indexován nebo indexován produktem *MsgId*) nebo *CorrelId* (je-li indexován produktem *CorrelId*) shodující se s voláním MQGET .

MQGMO_WAIT je ignorován, pokud je zadán s MQGMO_BROWSE_MSG_UNDER_CURSOR nebo MQGMO_MSG_UNDER_CURSOR; není vyvolána žádná chyba.

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.

MQGMO_SET_SIGNAL
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 indexován nebo indexován produktem *MsgId*) nebo *CorrelId* (je-li indexován produktem *CorrelId*) shodující se s voláním MQGET .

Tato volba je podporována pouze v systému z/OS .

MQGMO_FAIL_IF_QUIESCING
Pokud je správce front ve stavu uvedení do klidového stavu, vynutí selhání volání MQGET .

V systému z/OStato volba také vynutí selhání volání MQGET v případě, že připojení (pro aplikaci CICS nebo IMS) je 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.

  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 souvisí s účastí 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. V systému z/OSse požadavek na získání nachází v rámci transakce. Ve všech ostatních prostředích se 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* , je-li načtená zpráva 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, HP-UX, z/OS, IBM i, Solaris a Linux, a klienti WebSphere MQ MQI 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. V systému z/OSse požadavek na získání nachází v rámci transakce. Ve všech ostatních prostředích se 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

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 systému IMS a systému CICS může být nutné, abyste po zálohování jednotky práce obsahující zprávu označené MQGMO_MARK_SKIP_BACKOUTzazálohovali další volání produktu WebSphere MQ . Chcete-li potvrdit novou jednotku práce obsahující označenou zprávu, je třeba zadat volání produktu WebSphere MQ . Volání může být libovolné volání WebSphere MQ .

1. Pokud jste v systému IMSnepoužili IMS APAR PN60855 a provozujete aplikaci IMS MPP nebo BMP.
2. V systému CICS, pokud spouštíte jakoukoli aplikaci.

V obou případech zadejte všechny volání produktu WebSphere MQ před potvrzením nové jednotky práce, která obsahuje 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:

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.

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 787).

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 .

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_MARK_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

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:

MQGMO_LOCK
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_LOCK není možné, když používáte klienta IBM WebSphere MQ v systému HP Integrity NonStop Server ke správci front produktu z/OS , když je koordinován TMF.

MQGMO_UNLOCK
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 datové zprávy: Následující volby se vztahují ke zpracování dat zprávy, když je zpráva přečtena z fronty:

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 855 .

- 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 zprávy" na stránce 383 .

Volby skupiny a segmentu: 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*, *MsgSeqNumber*a *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-Získat-volby zprávy” na stránce 337 , kde jsou uvedeny další podrobnosti.

Tabulka 506 na stránce 356 ukazuje hodnoty polí *MsgId*, *CorrelId*, *GroupId*, *MsgSeqNumber*a *Offset* , o 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 506. 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
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 popisovač 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 parametr 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 jsou 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-Získat-volby zprávy” na stránce 337 , 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*, *Offset*a *MatchOptions* na odpovídající hodnoty a pak vydat volání MQGET s MQGMO_SYNCPOINT nebo MQGMO_NO_SYNCPOINT nastaveným, 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 507 na stránce 357 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 507. 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

Tabulka 507. Výsledek, když volání MQGET nebo MQCLOSE není konzistentní s informacemi o skupině a segmentu (pokračování)

Aktuální volání je	Předchozí volání bylo MQGET s MQGMO_LOGICAL_ORDER	Předchozí volání bylo MQGET bez MQGMO_LOGICAL_ORDER
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*, *MsgSeqNumber*a *Offset* v MQMDa volby MQMO_* v produktu *MatchOptions* v MQGMObyly nastaveny správně před každým voláním MQGET .

Například aplikace, která chce *předat* 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 výše).

- 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 CFLEVEL (4).
- V systémech AIX, HP-UX, IBM i, Solaris, Linux, Windows a klientů WebSphere MQ MQI připojených k těmto systémům je tato volba podporována pro všechny lokální fronty.

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.

V případě *trvalých* zpráv 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.

Pro *přechodné* zprávy 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 jsou většinu polí v deskriptoru zprávy stejné pro všechny segmenty v logické zprávě; obvykle se jedná pouze o pole *MsgId*, *Offset*a *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 správce front nastaví tak, aby označoval, že znovu 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.

- V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty musí být objekt CFSTRUCT, na kterém je mapa fronty, na CFLEVEL (3) nebo CFLEVEL (4).

- V systémech AIX, HP-UX, IBM i, Solaris, Linux, Windows a klientů WebSphere MQ MQI připojených k těmto systémům je tato volba podporována pro všechny lokální fronty.

MQGMO_ALL_MSGS_AVAILABLE
Zprávy ve skupině budou k dispozici pro načtení pouze v případě, že jsou k dispozici *všechny* zprávy ve skupině. Pokud fronta obsahuje skupiny zpráv s některými 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* , když neexistuje žádná aktuální skupina nebo logická zpráva. Pokud *je* aktuální skupina nebo logická zpráva, 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 *always* má 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_* .

- V systému z/OSje tato volba podporována pro soukromé a sdílené fronty, ale fronta musí mít typ indexu MQIT_GROUP_ID. Pro sdílené fronty musí být objekt CFSTRUCT, na kterém je mapa fronty, na CFLEVEL (3) nebo CFLEVEL (4).
- V systémech AIX, HP-UX, IBM i, Solaris, Linux, Windows a klientů WebSphere MQ MQI připojených k těmto systémům je tato volba podporována pro všechny lokální fronty.

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 MQGMO_ALL_SEGMENTS_AVAILABLE efekt *pouze* , když neexistuje žádná aktuální logická zpráva. Pokud *je* aktuální logická zpráva, 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 *always* má 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. Pokud zde *je* směs typů sestav přítomných pro segmenty určité logické zprávy, tyto zprávy sestavy musí být načteny jeden 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 musí být objekt CFSTRUCT, na kterém je mapa fronty, na CFLEVEL (3) nebo CFLEVEL (4).
- V systémech AIX, HP-UX, IBM i, Solaris, Linux, Windows a klientů WebSphere MQ MQI připojených k těmto systémům je tato volba podporována pro všechny lokální fronty.

Volby vlastností: Následující volby souvisejí s vlastnostmi 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 zadána volba *MsgHandle*, je ignorována.

MQGMO_PROPERTIES_COMPATIBILITY
Pokud zpráva obsahuje vlastnost s předponou **"mcd."**, **"jms."**, **"usr."**nebo **"mqext."**, jsou všechny vlastnosti zprávy doručeny do aplikace v záhlaví MQRFH2 . Jinak budou všechny vlastnosti zprávy, kromě vlastností obsažených v deskriptoru (či rozšíření) zprávy, zahozeny a nebudou nadále pro aplikaci přístupné.

Výchozí volba: Pokud není požadována žádná z uvedených 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.

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.

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**.

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.

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.

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.

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.

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 se nastavují v bitech 2 až 31 ECB, oblast definovaná v makru mapování z/OS 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.

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.

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 zprávy” na stránce 383 .) 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 250 milisekund vyššími 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`.

Počáteční hodnoty a jazyková prohlášení pro MQGMO

Tabulka 508. Počáteční hodnoty polí v MQGMO pro MQGMO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQGMO_STRUCTION_ID	`'GMO¬'`
Version	MQGMO_VERSION_1	`1`
Options	MQGMO_NO_WAIT	`0`
WaitInterval	Není	`0`
Signal1	Není	Ukazatel Null na systému z/OS; `0` jinak
Signal2	Není	`0`
ResolvedQName	Není	Nulový řetězec nebo prázdné znaky
MatchOptions	MQMO_MATCH_MSG_ID + MQMO_MATCH_CORREL_ID	`3`
GroupStatus	MQGS_NOT_IN_GROUP	`'¬'`
SegmentStatus	SEGMENT MQSS_NOT_SEGMENT	`'¬'`
Segmentation	MQSEG_BLOKOVÁNO	`'¬'`
Reserved1	Není	`'¬'`
MsgToken	MQMTOK_NONE	Hodnoty null
ReturnedLength	MQRL_UNDEFINED	`-1`
Reserved2	Není	`'¬'`
MsgHandle	MQM_NONE	`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.
3. V programovacím jazyce C-proměnná makraHodnota MQGMO_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQGMO MyGMO = {MQGMO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQGMO MQGMO;
struct tagMQGMO {
```

```
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of */
                             MQGET */
  MQLONG    WaitInterval;    /* Wait interval */
  MQLONG    Signal1;         /* Signal */
  MQLONG    Signal2;         /* Signal identifier */
  MQCHAR48  ResolvedQName;   /* Resolved name of destination queue */
  /* Ver:1 */
  MQLONG    MatchOptions;    /* Options controlling selection */
                             criteria used for MQGET */
  MQCHAR    GroupStatus;     /* Flag indicating whether message */
                             retrieved is in a group */
  MQCHAR    SegmentStatus;   /* Flag indicating whether message */
                             retrieved is a segment of a logical */
                             message */
  MQCHAR    Segmentation;    /* Flag indicating whether further */
                             segmentation is allowed for the */
                             message retrieved */
  MQCHAR    Reserved1;       /* Reserved */
  /* Ver:2 */
  MQBYTE16  MsgToken;        /* Message token */
  MQLONG    ReturnedLength;  /* Length of message data returned */
                             (bytes) */
  /* Ver:3 */
  MQLONG    Reserved2;       /* Reserved */
  MQHMSG    MsgHandle;       /* Message handle */
  /* Ver:4 */
};
```

- V systému z/OSje pole *Signal1* deklarováno jako PMQLONG.

Deklarace COBOL

```
**   MQGMO structure
  10 MQGMO.
**    Structure identifier
   15 MQGMO-STRUCID        PIC X(4).
**    Structure version number
   15 MQGMO-VERSION        PIC S9(9) BINARY.
**    Options that control the action of MQGET
   15 MQGMO-OPTIONS        PIC S9(9) BINARY.
**    Wait interval
   15 MQGMO-WAITINTERVAL   PIC S9(9) BINARY.
**    Signal
   15 MQGMO-SIGNAL1        PIC S9(9) BINARY.
**    Signal identifier
   15 MQGMO-SIGNAL2        PIC S9(9) BINARY.
**    Resolved name of destination queue
   15 MQGMO-RESOLVEDQNAME  PIC X(48).
**    Options controlling selection criteria used for MQGET
   15 MQGMO-MATCHOPTIONS   PIC S9(9) BINARY.
**    Flag indicating whether message retrieved is in a group
   15 MQGMO-GROUPSTATUS    PIC X.
**    Flag indicating whether message retrieved is a segment of a
**    logical message
   15 MQGMO-SEGMENTSTATUS  PIC X.
**    Flag indicating whether further segmentation is allowed for the
**    message retrieved
   15 MQGMO-SEGMENTATION   PIC X.
**    Reserved
   15 MQGMO-RESERVED1      PIC X.
**    Message token
   15 MQGMO-MSGTOKEN       PIC X(16).
**    Length of message data returned (bytes)
   15 MQGMO-RETURNEDLENGTH PIC S9(9) BINARY.
**    Reserved
   15 MQGMO-RESERVED2      PIC S9(9) BINARY.
**    Message handle
   15 MQGMO-MSGHANDLE      PIC S9(18) BINARY.
```

- V systému z/OSje pole *Signal1* deklarováno jako POINTER.

Deklarace PL/I

```
dcl
```

```
 1 MQGMO based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 Options        fixed bin(31), /* Options that control the action of
                                     MQGET */
  3 WaitInterval   fixed bin(31), /* Wait interval */
  3 Signal1        fixed bin(31), /* Signal */
  3 Signal2        fixed bin(31), /* Signal identifier */
  3 ResolvedQName  char(48),      /* Resolved name of destination
                                     queue */
  3 MatchOptions   fixed bin(31), /* Options controlling selection
                                     criteria used for MQGET */
  3 GroupStatus    char(1),       /* Flag indicating whether message
                                     retrieved is in a group */
  3 SegmentStatus  char(1),       /* Flag indicating whether message
                                     retrieved is a segment of a logical
                                     message */
  3 Segmentation   char(1),       /* Flag indicating whether further
                                     segmentation is allowed for the
                                     message retrieved */
  3 Reserved1      char(1),       /* Reserved */
  3 MsgToken       char(16),      /* Message token */
  3 ReturnedLength fixed bin(31); /* Length of message data returned
                                     (bytes) */
  3 Reserved2      fixed bin(31); /* Reserved */
  3 MsgHandle      fixed bin(63); /* Message handle */
```

- V systému z/OSje pole *Signal1* deklarováno jako `pointer`.

Deklarace High Level Assembler

```
MQGMO                 DSECT
MQGMO_STRUCID         DS   CL4   Structure identifier
MQGMO_VERSION         DS   F     Structure version number
MQGMO_OPTIONS         DS   F     Options that control the action of
*                                MQGET
MQGMO_WAITINTERVAL    DS   F     Wait interval
MQGMO_SIGNAL1         DS   F     Signal
MQGMO_SIGNAL2         DS   F     Signal identifier
MQGMO_RESOLVEDQNAME   DS   CL48  Resolved name of destination queue
MQGMO_MATCHOPTIONS    DS   F     Options controlling selection criteria
*                                used for MQGET
MQGMO_GROUPSTATUS     DS   CL1   Flag indicating whether message
*                                retrieved is in a group
MQGMO_SEGMENTSTATUS   DS   CL1   Flag indicating whether message
*                                retrieved is a segment of a logical
*                                message
MQGMO_SEGMENTATION    DS   CL1   Flag indicating whether further
*                                segmentation is allowed for the message
*                                retrieved
MQGMO_RESERVED1       DS   CL1   Reserved
MQGMO_MSGTOKEN        DS   XL16  Message token
MQGMO_RETURNEDLENGTH  DS   F     Length of message data returned (bytes)
MQGMO_RESERVED2       DS   F     Reserved
MQGMO_MSGHANDLE       DS   D     Message handle
MQGMO_LENGTH          EQU  *-MQGMO
                      ORG  MQGMO
MQGMO_AREA            DS   CL(MQGMO_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQGMO
  StrucId        As String*4  'Structure identifier'
  Version        As Long      'Structure version number'
  Options        As Long      'Options that control the action of MQGET'
  WaitInterval   As Long      'Wait interval'
  Signal1        As Long      'Signal'
  Signal2        As Long      'Signal identifier'
  ResolvedQName  As String*48 'Resolved name of destination queue'
  MatchOptions   As Long      'Options controlling selection criteria'
                              'used for MQGET'
  GroupStatus    As String*1  'Flag indicating whether message'
                              'retrieved is in a group'
  SegmentStatus  As String*1  'Flag indicating whether message'
                              'retrieved is a segment of a logical'
                              'message'
```

```
  Segmentation   As String*1  'Flag indicating whether further'
                              'segmentation is allowed for the message'
                              'retrieved'
  Reserved1      As String*1  'Reserved'
  MsgToken       As MQBYTE16  'Message token'
  ReturnedLength As Long      'Length of message data returned (bytes)'
End Type
```

MQIIH-záhlaví informací IMS

Následující tabulka shrnuje pole ve struktuře.

Tabulka 509. Pole v MQIIH

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQIIH	StrucLength
Encoding	Vyhrazené	Kódování
CodedCharSetId	Vyhrazené	CodedCharSetId
Format	Název formátu produktu MQ pro data následující MQIIH	Formát
Flags	Příznaky	Příznaky
LTermOverride	Potlačení logického terminálu	LTermOverride
MFSMapName	Název mapy služeb formátu zpráv	MFSMapName
ReplyToFormat	Název formátu MQ zprávy odpovědi	ReplyToFormát
Authenticator	Heslo RACF™ nebo přístupový lístek	Ověřovatel
TranInstanceId	Identifikátor instance transakce	TranInstanceID
TranState	Stav transakce	TranState
CommitMode	Režim vázaného zpracování	CommitMode
SecurityScope	Rozsah zabezpečení	SecurityScope
Reserved	Vyhrazené	Vyhrazené

Přehled pro MQIIH

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

Účel: Struktura MQIIH popisuje informace, které musí být přítomny na začátku zprávy odeslané do mostu IMS prostřednictvím produktu WebSphere MQ pro systém z/OS.

Název formátu: MQFMT_IMS.

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 je v některém z podporovaných znakových sad a kódování. Přijímající agent kanálu zpráv připojený ke správci front, který vlastní frontu mostu IMS , konvertuje MQIIH.

- Data zprávy aplikace následující za strukturou 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.

  Pokud data nejsou jedním z vestavěných formátů podporovaných správcem front, musíte data uživatelské procedury pro převod dat převést na základě data převodu dat.

Pole pro MQIIH

Struktura MQIIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Ověřovatel (MQCHAR8)

Jedná se o heslo RACF nebo PassTicket. Je volitelný; je-li zadán, použije se s ID uživatele v kontextu zabezpečení MQMD k sestavení UTOKEN, které je odesláno na IMS , aby poskytl kontext 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.

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 .

CommitMode (MQCHAR)

Jedná se o režim vázaného zpracování IMS . Další informace o režimech vázaného zpracování 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.

Jakákoli transakce IMS zahájená jako výsledek příkazu commit mpde MQICM_SEND_THEN_COMMIT se spustí v režimu RESPONSE, bez ohledu na to, jak je transakce definována v definici systému IMS (parametr MSGTYPE v makru TRANSACT). Toto platí také pro transakce zahájené přepínačem transakce.

Počáteční hodnota tohoto pole je MQICM_COMMIT_THEN_SEND.

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 .

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 IMS neodpověděl na IOPCB ani na jinou transakci.

Počáteční hodnota tohoto pole je MQIIH_NONE.

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.

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 397.

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.

Vyhrazeno (MQCHAR)

Jedná se o vyhrazené pole; musí být prázdné.

SecurityScope (MQCHAR)

To označuje, že je požadováno zpracování zabezpečení 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.

StrucId (MQCHAR4)

Jedná se o identifikátor struktury. Hodnota musí být:

MQIIH_STRUCTION_ID
Identifikátor pro strukturu záhlaví informací 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.

StrucLength (MQLONG)

Toto je délka struktury MQIIH. Hodnota musí být:

MQIIH_LENGTH_1
Délka struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je MQIIH_LENGTH_1.

ID TranInstance(MQBYTE16)

Jedná se o identifikátor instance transakce. Toto pole je používáno pro výstupní zprávy z IMS, takže je na prvním vstupu ignorován. Pokud jste nastavili *TranState* na hodnotu MQITS_IN_CONVERSATION, musí být tato hodnota poskytnuta na dalším vstupu a všechny následné vstupy, abyste povolili IMS korelovat zprávy se správnou konverzací. 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)

Označuje stav konverzace 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 systémem 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 . Vrací data stavu transakce ve formuláři IMS , nikoli ve formě znaku.

Počáteční hodnota tohoto pole je MQITS_NOT_IN_CONVERSATION.

Verze (MQLONG)

Jedná se o číslo verze struktury. Hodnota musí být:

MQIIH_VERSION_1
Číslo verze pro strukturu záhlaví informací IMS .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQIIH_VERSION
Aktuální verze struktury záhlaví informací IMS .

Počáteční hodnota tohoto pole je MQIIH_VERSION_1.

Počáteční hodnoty a jazyková prohlášení pro MQIIH

Tabulka 510. Počáteční hodnoty polí v MQIIH pro MQIIH

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQIIH_STRUCTION_ID	'IIH?'
Version	MQIIH_VERSION_1	1
StrucLength	MQIIH_LENGTH_1	84
Encoding	Není	0
CodedCharSetId	Není	0
Format	MQFMT_NONE	Mezery
Flags	MQIIH_NONE	0
LTermOverride	Není	Mezery
MFSMapName	Není	Mezery
ReplyToFormat	MQFMT_NONE	Mezery
Authenticator	MQIAUT_NONE	Mezery
TranInstanceId	MQITII_NONE	Hodnoty null
TranState	MQITS_NOT_IN_CONVERSATION	'?'
CommitMode	MQICM_COMMIT_THEN_SEND	'0'
SecurityScope	KONTROLA MQISS_CHECK	'C'
Reserved	Není	'?'

Notes:

1. Symbol? zastupuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQIIH_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQIIH MyIIH = {MQIIH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQIIH MQIIH;
struct tagMQIIH {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    StrucLength;     /* Length of MQIIH structure */
  MQLONG    Encoding;        /* Reserved */
  MQLONG    CodedCharSetId;  /* Reserved */
  MQCHAR8   Format;          /* MQ format name of data that follows
                                MQIIH */
  MQLONG    Flags;           /* Flags */
  MQCHAR8   LTermOverride;   /* Logical terminal override */
  MQCHAR8   MFSMapName;      /* Message format services map name */
  MQCHAR8   ReplyToFormat;   /* MQ format name of reply message */
  MQCHAR8   Authenticator;   /* RACF password or passticket */
  MQBYTE16  TranInstanceId;  /* Transaction instance identifier */
  MQCHAR    TranState;       /* Transaction state */
  MQCHAR    CommitMode;      /* Commit mode */
  MQCHAR    SecurityScope;   /* Security scope */
  MQCHAR    Reserved;        /* Reserved */
};
```

Deklarace COBOL

```
**   MQIIH structure
  10 MQIIH.
**    Structure identifier
   15 MQIIH-STRUCID        PIC X(4).
**    Structure version number
   15 MQIIH-VERSION        PIC S9(9) BINARY.
**    Length of MQIIH structure
   15 MQIIH-STRUCLENGTH    PIC S9(9) BINARY.
**    Reserved
   15 MQIIH-ENCODING       PIC S9(9) BINARY.
**    Reserved
   15 MQIIH-CODEDCHARSETID PIC S9(9) BINARY.
**    MQ format name of data that follows MQIIH
   15 MQIIH-FORMAT         PIC X(8).
**    Flags
   15 MQIIH-FLAGS          PIC S9(9) BINARY.
**    Logical terminal override
   15 MQIIH-LTERMOVERRIDE  PIC X(8).
**    Message format services map name
   15 MQIIH-MFSMAPNAME     PIC X(8).
**    MQ format name of reply message
   15 MQIIH-REPLYTOFORMAT  PIC X(8).
**    RACF password or passticket
   15 MQIIH-AUTHENTICATOR  PIC X(8).
**    Transaction instance identifier
   15 MQIIH-TRANINSTANCEID PIC X(16).
**    Transaction state
   15 MQIIH-TRANSTATE      PIC X.
**    Commit mode
   15 MQIIH-COMMITMODE     PIC X.
**    Security scope
   15 MQIIH-SECURITYSCOPE  PIC X.
**    Reserved
   15 MQIIH-RESERVED       PIC X.
```

Deklarace PL/I

```
dcl
 1 MQIIH based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 StrucLength    fixed bin(31), /* Length of MQIIH structure */
  3 Encoding       fixed bin(31), /* Reserved */
  3 CodedCharSetId fixed bin(31), /* Reserved */
  3 Format         char(8),       /* MQ format name of data that follows
                                     MQIIH */
  3 Flags          fixed bin(31), /* Flags */
  3 LTermOverride  char(8),       /* Logical terminal override */
  3 MFSMapName     char(8),       /* Message format services map name */
```

```
  3 ReplyToFormat  char(8),       /* MQ format name of reply message */
  3 Authenticator  char(8),       /* RACF password or passticket */
  3 TranInstanceId char(16),      /* Transaction instance identifier */
  3 TranState      char(1),       /* Transaction state */
  3 CommitMode     char(1),       /* Commit mode */
  3 SecurityScope  char(1),       /* Security scope */
  3 Reserved       char(1);       /* Reserved */
```

Deklarace High Level Assembler

```
MQIIH                 DSECT
MQIIH_STRUCID         DS   CL4   Structure identifier
MQIIH_VERSION         DS   F     Structure version number
MQIIH_STRUCLENGTH     DS   F     Length of MQIIH structure
MQIIH_ENCODING        DS   F     Reserved
MQIIH_CODEDCHARSETID  DS   F     Reserved
MQIIH_FORMAT          DS   CL8   MQ format name of data that follows
*                                MQIIH
MQIIH_FLAGS           DS   F     Flags
MQIIH_LTERMOVERRIDE   DS   CL8   Logical terminal override
MQIIH_MFSMAPNAME      DS   CL8   Message format services map name
MQIIH_REPLYTOFORMAT   DS   CL8   MQ format name of reply message
MQIIH_AUTHENTICATOR   DS   CL8   RACF password or passticket
MQIIH_TRANINSTANCEID  DS   XL16  Transaction instance identifier
MQIIH_TRANSTATE       DS   CL1   Transaction state
MQIIH_COMMITMODE      DS   CL1   Commit mode
MQIIH_SECURITYSCOPE   DS   CL1   Security scope
MQIIH_RESERVED        DS   CL1   Reserved
*
MQIIH_LENGTH          EQU  *-MQIIH
                      ORG  MQIIH
MQIIH_AREA            DS   CL(MQIIH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQIIH
  StrucId        As String*4 'Structure identifier'
  Version        As Long     'Structure version number'
  StrucLength    As Long     'Length of MQIIH structure'
  Encoding       As Long     'Reserved'
  CodedCharSetId As Long     'Reserved'
  Format         As String*8 'MQ format name of data that follows MQIIH'
  Flags          As Long     'Flags'
  LTermOverride  As String*8 'Logical terminal override'
  MFSMapName     As String*8 'Message format services map name'
  ReplyToFormat  As String*8 'MQ format name of reply message'
  Authenticator  As String*8 'RACF password or passticket'
  TranInstanceId As MQBYTE16 'Transaction instance identifier'
  TranState      As String*1 'Transaction state'
  CommitMode     As String*1 'Commit mode'
  SecurityScope  As String*1 'Security scope'
  Reserved       As String*1 'Reserved'
End Type
```

MQIMPO-Dotaz na volby vlastností zprávy

Následující tabulka shrnuje pole ve struktuře. MQIMPO strukturu-dotaz na volby vlastností zprávy

Tabulka 511. Pole v MQIMPO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby řízení akce MQINQMP	Volby
RequestedEncoding	Kódování, do kterého se má dotazovaná vlastnost převést	RequestedEncoding

Tabulka 511. Pole v MQIMPO (pokračování)		
Pole	**Popis**	**Téma**
RequestedCCSID	Znaková sada dotazované vlastnosti	RequestedCCSID
ReturnedEncoding	Kódování vrácené hodnoty	ReturnedEncoding
ReturnedCCSID	Znaková sada vrácené hodnoty	ReturnedCCSID
Reserved1	Vyhrazené pole	ReturnedCCSID
ReturnedName	Název dotazované vlastnosti	ReturnedName
TypeString	Znázornění řetězce datového typu vlastnosti	TypeString

Přehled pro MQIMPO

Struktura voleb vlastností dotazové zprávy.

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

Účel: Struktura MQIMPO umožňuje aplikacím zadávat volby, které řídí, jak se mají dotazovat vlastnosti zpráv. Struktura je vstupním parametrem volání MQINQMP.

Znaková sada a kódování: Data ve struktuře MQIMPO musí být ve znakové sadě aplikace a kódování aplikace (MQENC_NATIVE).

Pole pro MQIMPO

Dotaz na struktury voleb vlastností zprávy-pole

Struktura MQIMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Volby (MQLONG)
Dotaz na strukturu voleb vlastností zprávy-pole Volby

Následující volby řídí akci MQINQMP. Můžete uvést jednu nebo více z těchto voleb, a pokud potřebujete více než jednu, mohou být tyto hodnoty:

- Přidáno společně (nepřidávejte stejnou konstantu více než jednou), nebo
- zkombinovat pomocí bitové operace OR (jestliže 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:

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
MQTYPE_INT8	MQTYPE_STRING, MQTYPE_INT16, MQTYPE_INT32, MQTYPE_INT64

Datový typ vlastnosti	Podporované cílové datové typy
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 WebSphere MQ vrátí kód příčiny MQRC_PROP_NUMBER_FORMAT_ERROR.
- Pokud se aplikace pokusí o převod, který není podporován, produkt WebSphere 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 WebSphere MQ vrací 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.

WebSphere 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.

WebSphere 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.

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.

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.

Reserved1 (MQCHAR)

Jedná se o vyhrazené pole. Počáteční hodnota tohoto pole je prázdný znak (4 bajtové pole).

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.

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.

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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQIMPO

Zjistit strukturu vlastností vlastností zprávy-počáteční hodnoty

Tabulka 512. Počáteční hodnoty polí v MQIPMO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQIMPO_STRUCT	'IMPO'
Version	MQIMPO_VERSION_1	1
Options	MQIMPO_INQ_FIRST	
RequestedEncoding	MQENC_NATIVE	
RequestedCCSID	MQCCSI_APPL	
ReturnedEncoding	MQENC_NATIVE	
ReturnedCCSID	0	
Reserved1	0	
ReturnedName	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	
TypeString	Nulový řetězec nebo prázdné znaky	

Notes:

1. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyce C-proměnná makraHodnota MQIMPO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQIMPO MyIMPO = {MQIMPO_DEFAULT};
```

Deklarace C
Dotaz na strukturu voleb vlastností zprávy-deklarace jazyka C

```
typedef struct tagMQIMPO MQIMPO;
struct tagMQIMPO {
  MQCHAR4  StrucId;            /* Structure identifier */
  MQLONG   Version;            /* Structure version number */
  MQLONG   Options;            /* Options that control the action of
                                  MQINQMP */
  MQLONG   RequestedEncoding;  /* Requested encoding of Value */
  MQLONG   RequestedCCSID;     /* Requested character set identifier
                                  of Value */
  MQLONG   ReturnedEncoding;   /* Returned encoding of Value */
  MQLONG   ReturnedCCSID;      /* Returned character set identifier
                                  of Value */
  MQCHAR   Reserved1           /* Reserved field */
  MQCHARV  ReturnedName;       /* Returned property name */
  MQCHAR8  TypeString;         /* Property data type as a string */
};
```

Deklarace COBOL
Dotaz na strukturu vlastností zprávy-deklarace jazyka COBOL

```
**   MQIMPO structure
  10 MQIMPO.
**    Structure identifier
   15 MQIMPO-STRUCID                PIC X(4).
**    Structure version number
   15 MQIMPO-VERSION                PIC S9(9) BINARY.
**    Options that control the action of MQINQMP
   15 MQIMPO-OPTIONS                PIC S9(9) BINARY.
**    Requested encoding of VALUE
   15 MQIMPO-REQUESTEDENCODING      PIC S9(9) BINARY.
**    Requested character set identifier of VALUE
   15 MQIMPO-REQUESTEDCCSID         PIC S9(9) BINARY.
**    Returned encoding of VALUE
   15 MQIMPO-RETURNEDENCODING       PIC S9(9) BINARY.
**    Returned character set identifier of VALUE
   15 MQIMPO-RETURNEDCCSID          PIC S9(9) BINARY.
**    Reserved field
   15 MQIMPO-RESERVED1
**    Returned property name
   15 MQIMPO-RETURNEDNAME.
**     Address of variable length string
    20 MQIMPO-RETURNEDNAME-VSPTR    POINTER.
**     Offset of variable length string
    20 MQIMPO-RETURNEDNAME-VSOFFSET PIC S9(9) BINARY.
**     CCSID of variable length string
    20 MQIMPO-RETURNEDNAME-VSCCSID  PIC S9(9) BINARY.
**    Property data type as string
   15 MQIMPO-TYPESTRING             PIC S9(9) BINARY.
```

Deklarace PL/I
Dotaz na strukturu vlastností zprávy-deklarace jazyka PL/I

```
dcl
 1 MQIMPO based,
  3 StrucId            char(4),       /* Structure identifier */
  3 Version            fixed bin(31), /* Structure version number */
  3 Options            fixed bin(31), /* Options that control the
                                         action of MQINQMP */
  3 RequestedEncoding  fixed bin(31), /* Requested encoding of
                                         Value */
  3 RequestedCCSID     fixed bin(31), /* Requested character set
                                         identifier of Value */
  3 ReturnedEncoding   fixed bin(31), /* Returned encoding of
                                         Value */
  3 ReturnedCCSID      fixed bin(31), /* Returned character set
                                         identifier of Value */
  3 Reserved1          fixed bin(31), /* Reserved field */
  3 ReturnedName,                     /* Returned property name */
   5 ReturnedName_VSPtr      pointer,       /* Address of returned
                                               name */
   5 5 ReturnedName_VSOffset fixed bin(31), /* Offset of returned
                                               name */
```

```
    5 5 ReturnedName_VSCCSID  fixed bin(31), /* CCSID of returned
                                                name */
  3 TypeString         char(8);       /* Property data type as
                                         string */
```

Deklarace High Level Assembler
Dotazovat strukturu vlastností zprávy-deklarace jazyka assembler

```
MQIMPO                       DSECT
MQIMPO_STRUCID               DS   CL4  Structure identifier
MQIMPO_VERSION               DS   F    Structure version number
MQIMPO_OPTIONS               DS   F    Options that control the
*                                      action of MQINQMP
MQIMPO_REQUESTEDENCODING     DS   F    Requested encoding of VALUE
MQIMPO_REQUESTEDCCSID        DS   F    Requested character set
*                                      identifier of VALUE
MQIMPO_RETURNEDENCODING      DS   F    Returned encoding of VALUE
MQIMPO_RETURNEDCCSID         DS   F    Returned character set
*                                      identifier of VALUE
MQIMPO_RESERVED1             DS   F    Reserved field
MQIMPO_RETURNEDNAME          DS   0F   Force fullword alignment
MQIMPO_RETURNEDNAME_VSPTR    DS   F    Address of returned name
MQIMPO_RETURNEDNAME_VSOFFSET DS   F    Offset of returned name
MQIMPO_RETURNEDNAME_VSLENGTH DS   F    Length of returned name
MQIMPO_RETURNEDNAME_VSCCSID  DS   F    CCSID of returned name
MQIMPO_RETURNEDNAME_LENGTH   EQU  *-MQIMPO_RETURNEDNAME
                             ORG  MQIMPO_RETURNEDNAME
MQIMPO_RETURNEDNAME_AREA     DS   CL(MQIMPO_RETURNEDNAME_LENGTH)
*
MQIMPO_TYPESTRING            DS   CL8  Property data type as string
MQIMPO_LENGTH                EQU  *-MQIMPO
MQIMPO_AREA                  DS   CL(MQIMPO_LENGTH)
```

MQMD-deskriptor zprávy

Následující tabulka shrnuje pole ve struktuře.

Tabulka 513. Pole v MQMD

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Report	Volby pro zprávy sestav	Sestava
MsgType	Typ zprávy	MsgType
Expiry	Životnost zprávy	Pole MQMD-Expiry
Feedback	Zpětná vazba nebo kód příčiny	Pole MQMD-Feedback
Encoding	Číselný kódování dat zprávy	Kódování
CodedCharSetId	Identifikátor znakové sady dat zprávy	CodedCharSetId
Format	Název formátu dat zprávy	Formát
Priority	Priorita zprávy	Priorita
Persistence	Trvalost zpráv	Trvání
MsgId	Identifikátor zprávy	MQMD- MsgId , pole
CorrelId	Identifikátor korelace	CorrelId
BackoutCount	Počítadlo odvolaných	BackoutCount
ReplyToQ	Název fronty odpovědí	ReplyToQ
ReplyToQMgr	Název správce front odpovědí	ReplyToQMgr

Tabulka 513. Pole v MQMD (pokračování)		
Pole	**Popis**	**Téma**
UserIdentifier	Identifikátor uživatele	UserIdentifier
AccountingToken	Token evidence	AccountingToken
ApplIdentityData	Údaje o žádosti vztahující se k totožnosti	ApplIdentityData
PutApplType	Typ aplikace, která vložila zprávu	PutApplType
PutApplName	Název aplikace, která vložila zprávu	PutApplName
PutDate	Datum, kdy byla zpráva vložena	PutDate
PutTime	Čas, kdy byla zpráva vložena	PutTime
ApplOriginData	Údaje o žádosti vztahující se k původu	ApplOriginData
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQMD_VERSION_2.		
GroupId	Identifikátor skupiny	GroupId
MsgSeqNumber	Pořadové číslo logické zprávy v rámci skupiny	MsgSeqNumber
Offset	Posunutí dat ve fyzické zprávě od začátku logické zprávy	Offset
MsgFlags	Příznaky zprávy	pole MQMD- MsgFlags
OriginalLength	Délka původní zprávy	OriginalLength

Přehled pro MQMD

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ MQI připojené k těmto systémům.

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 deskriptoru MQMD je MQMD_VERSION_2. Aplikace, které mají být přenositelné mezi několika prostředími, musí zajistit, aby požadovaná verze MQMD byla podporována ve všech příslušných prostředích. Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnovější verzi MQMD, která je podporována prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQMD_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí nastavit pole *Version* na číslo verze požadované verze.

Deklarace pro strukturu version-1 je k dispozici s názvem MQMD1.

Znaková sada a kódování: Data ve struktuře MQMD musí být ve znakové sadě a kódování lokálního správce front; tyto údaje jsou dány atributem správce front *CodedCharSetId* a MQENC_NATIVE. Je-li však aplikace spuštěna jako klient WebSphere MQ MQI, musí být tato 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: version-2 MQMD je ekvivalentní k použití MQMD version-1 a k určení dat zprávy se strukturou MQMDE. Nicméně, pokud mají všechny pole ve struktuře MQMDE své výchozí hodnoty, lze hodnotu MQMDE vynechat. Používá se version-1 MQMD plus MQMDE, jak je popsáno:

- Je-li v rámci volání MQPUT a MQPUT1 aplikace MQMD version-1 , může aplikace volitelně připojit data zprávy k datům zprávy MQMDE a nastavit pole *Format* v MQMD na MQFMT_MD_EXTENSION tak, aby bylo zřejmé, že je přítomen objekt 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 do ekvivalentních polí ve výstupu MQMDE na výstupu z volání MQPUT a MQPUT1 ; pokud aplikace tyto výstupní hodnoty vyžaduje, 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 *Format* v deskriptoru MQMD bude mít hodnotu MQFMT_MD_EXTENSION, aby bylo zřejmé, že je přítomen prvek MQMDE.

Výchozí hodnoty, které správce front používá pro pole v MQMDE, jsou stejné jako počáteční hodnoty těchto polí, zobrazené v Tabulka 518 na stránce 435.

Je-li zpráva v přenosové frontě, některá pole v produktu MQMD jsou nastavena na konkrétní hodnoty; podrobnosti viz “MQXQH-záhlaví přenosové fronty” na stránce 576 .

Kontext zprávy: Určitá pole v deskriptoru MQMD obsahují kontext zprávy. Existují dva typy kontextu zprávy: *kontext identity* a *kontext původu*. Typicky:

- Kontext identity souvisí s aplikací, která *původně* umístila zprávu
- Kontext původu souvisí s aplikací, která *nejnověji* umístili 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 původu obvykle má popisovaný význam, obsah obou typů kontextových polí v MQMD závisí na volbách MQPMO_ * _CONTEXT, které jsou určeny 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 umístila zprávu; závisí na návrhu sady aplikací.

Agent MCA (Message Channel Agent) nikdy nemění kontext zprávy. MCV, které přijímají zprávy ze vzdálených správců front, používají kontextovou volbu MQPMO_SET_ALL_CONTEXT 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 se kontext původu nevztahuje k žádné z jednotek MCAs, které odeslaly a obdržely zprávu. Kontext původu odkazuje na předchozí aplikaci, která vložila zprávu. Pokud všechny mezilehlé aplikace prošly kontextem zprávy, kontext původu odkazuje na původní aplikaci samotnou.

V popisech jsou popisována pole kontextu, jako kdyby byla použita, jak bylo popsáno výše. Další informace o kontextu zprávy viz téma Kontext zprávy.

Pole pro MQMD

Struktura MQMD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Tabulka 514. Pole v MQMD

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Report	Volby pro zprávy sestav	Sestava
MsgType	Typ zprávy	MsgType
Expiry	Životnost zprávy	Pole MQMD-Expiry
Feedback	Zpětná vazba nebo kód příčiny	Pole MQMD-Feedback

Tabulka 514. Pole v MQMD (pokračování)		
Pole	**Popis**	**Téma**
Encoding	Číselný kódování dat zprávy	Kódování
CodedCharSetId	Identifikátor znakové sady dat zprávy	CodedCharSetId
Format	Název formátu dat zprávy	Formát
Priority	Priorita zprávy	Priorita
Persistence	Trvalost zpráv	Trvání
MsgId	Identifikátor zprávy	MQMD- MsgId , pole
CorrelId	Identifikátor korelace	CorrelId
BackoutCount	Počítadlo odvolaných	BackoutCount
ReplyToQ	Název fronty odpovědí	ReplyToQ
ReplyToQMgr	Název správce front odpovědí	ReplyToQMgr
UserIdentifier	Identifikátor uživatele	UserIdentifier
AccountingToken	Token evidence	AccountingToken
ApplIdentityData	Údaje o žádosti vztahující se k totožnosti	ApplIdentityData
PutApplType	Typ aplikace, která vložila zprávu	PutApplType
PutApplName	Název aplikace, která vložila zprávu	PutApplName
PutDate	Datum, kdy byla zpráva vložena	PutDate
PutTime	Čas, kdy byla zpráva vložena	PutTime
ApplOriginData	Údaje o žádosti vztahující se k původu	ApplOriginData
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQMD_VERSION_2.		
GroupId	Identifikátor skupiny	GroupId
MsgSeqNumber	Pořadové číslo logické zprávy v rámci skupiny	MsgSeqNumber
Offset	Posunutí dat ve fyzické zprávě od začátku logické zprávy	Offset
MsgFlags	Příznaky zprávy	pole MQMD- MsgFlags
OriginalLength	Délka původní zprávy	OriginalLength

AccountingToken (MQBYTE32)

Jedná se o účtovací token, část **kontextu identity** zprávy. Další informace o kontextu zprávy viz “Přehled pro MQMD” na stránce 384; viz také Kontext zprávy.

AccountingToken umožňuje aplikaci správně účtovat za práci provedenou jako výsledek zprávy. Správce front považuje tyto informace za řetězec bitů a nekontroluje jeho obsah.

Správce front tyto informace generuje následujícím způsobem:

- 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.
  - V systému z/OS jsou informace o účtování nastaveny na:

- V případě dávky z/OS účtovací informace z karty JES JOB nebo z příkazu JES ACCT na 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 IMSse osmiznakový název PSB zřetězený s tokenem zotavení IMS o 16 znacích.

– V systému IBM ijsou informace o účtování nastaveny na účtovací kód úlohy.
– V systémech UNIX jsou informace o účtování nastaveny na číselný identifikátor uživatele, ve znacích ASCII.
– V systému Windowsjsou informace o účtování nastaveny na identifikátor zabezpečení (SID) systému Windows v komprimovaném formátu. Identifikátor SID jednoznačně identifikuje identifikátor uživatele uložený v poli *UserIdentifier* . Když je SID uloženo v poli *AccountingToken* , 6bajtová identifikační autorita (umístěná ve třetím a následujících bajtech SID) se vynechá. Je-li například SID systému Windows 28 bajtů dlouhé, jsou v poli *AccountingToken* uloženy 22 bajtů informací SID.

- Poslední bajt (bajt 32) účtovacího pole je nastaven na typ účtovacího tokenu (v tomto případě MQACTT_NT_SECURITY_ID, x '0b'):

  MQACCT_CICS_LUOW_ID
  Identifikátor LUOW CICS .

  MQACTT_NT_SECURITY_ID
  Identifikátor zabezpečení systému Windows .

  MQACTT_OS400_ACCOUNT_TOKEN
  Účtovací token IBM i .

  MQACTT_UNIX_NUMERIC_ID
  Číselný identifikátor systému UNIX .

  UŽIVATEL MQACTT_USER
  Uživatelem definovaný evidenční token.

  MQACTT_UNKNOWN
  Neznámý typ účtovacího tokenu.

  Typ účtovacího tokenu je nastaven na explicitní hodnotu pouze v následujících prostředích: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus WebSphere MQ MQI MQI připojené k těmto systémům. V jiných prostředích je typ účtovacího tokenu nastaven na hodnotu MQACTT_UNKNOWN. V těchto prostředích použijte pole *PutApplType* 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, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT, je-li zadán. Není-li zadán parametr MQPMO_SET_IDENTITY_CONTEXT ani MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a je to pole pouze pro výstup. Další informace o kontextu zprávy viz téma Kontext zprávy .

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat *AccountingToken* , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota *AccountingToken* , která je uchována se zprávou, pokud je zachována (viz popis MQPMO_RETAIN v souboru“Volby MQPMO (MQLONG)” na stránce 467 pro více podrobností o zachovaných publikacích), ale nepoužívá se jako *AccountingToken* , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu k přepsání *AccountingToken* 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; 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 *AccountingToken* 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.

Délka tohoto pole je dána hodnotou MQ_ACCOUNTING_TOKEN_LENGTH. Počáteční hodnota tohoto pole je MQACT_NONE.

Data ApplIdentity(MQCHAR32)

Tato část je součástí **kontextu identity** zprávy. Další informace o kontextu zprávy viz "Přehled pro MQMD" na stránce 384 a Kontext zprávy .

ApplIdentityData 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, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT, je-li zadán. Je-li přítomen znak null, správce front převede znak null a všechny následující znaky na mezery. Není-li zadán parametr MQPMO_SET_IDENTITY_CONTEXT ani MQPMO_SET_ALL_CONTEXT, je toto pole na vstupu ignorováno a je to pole pouze pro výstup. Další informace o kontextu zprávy viz téma Kontext zprávy.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat *ApplIdentityData* , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota *ApplIdentityData* , která je uchována se zprávou, je-li zachována (viz popis příkazu MQPMO_RETAIN pro více podrobností o zachovaných publikacích), ale nepoužívá se jako *ApplIdentityData* , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu k přepsání *ApplIdentityData* 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 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.

Data ApplOrigin(MQCHAR4)

Toto je část **kontextu původu** zprávy. Další informace o kontextu zprávy viz "Přehled pro MQMD" na stránce 384 a Kontext zprávy .

ApplOriginData 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 se jedná o vstupní/výstupní pole, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_ALL_CONTEXT. Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li parametr MQPMO_SET_ALL_CONTEXT zadán, bude 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 MQ_APPL_ORIGIN_DATA_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 4 prázdné znaky v jiných programovacích jazycích.

Když je zpráva publikována, ačkoli je nastavena *ApplOriginData* , je v odběru, který přijímá, prázdná.

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 787.

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.

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 753 .

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. Výjimka z tohoto pravidla je při vložení zprávy do fronty mostu IMS Bridge, která je vrácena v poli *CodedCharSetId* MQMD, je CCSID správce front.

Nesmíte používat následující speciální hodnotu:

MQCCSI_APPL
Výsledkem je nesprávná hodnota v poli *CodedCharSetId* v deskriptoru MQMD a při přijetí zprávy pomocí volání MQGET s volbou MQGMO_CONVERT způsobí návratový kód MQRC_SOURCE_CCSID_ERROR (nebo MQRC_FORMAT_ERROR for z/OS).

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-záhlaví vloženého PCF" na stránce 332.

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.

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.

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, identifikátor korelace *nebude* převeden mezi znakovými sadami při průbězích zpráv z jednoho správce front do jiného.

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 WebSphere MQ tento řetězec implementace specifický pro daný produkt obsahuje prvních 12 znaků názvu správce front a hodnoty odvozené 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 *není* určující parametr 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 mostem CICS 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* uvedeno, a výstupní pole, pokud je zadáno MQPMO_NEW_CORREL_ID *je* . 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.

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.

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 848 .

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.

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.

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 velikost původní doby 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.

Poznámka:

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.
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.
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.

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-kódy zpětné vazby mostu

Tyto kódy se používají při přijetí neočekávaného chybného kódu systému 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 bylo provedeno pomocí mostu IMS . V důsledku toho nebyla zpráva předána do systému IMS.

CHYBA MQFB_IMS_ERROR
Systém IMSvrátil neočekávanou chybu. Další informace o chybě naleznete v protokolu chyb produktu WebSphere MQ v systému, v němž je umístěn most IMS .

MQFB_IMS_FIRST
Když kód IMS-OTMA není 0x1A, IMSgenerované kódy zpětné vazby jsou 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 systémem IMS, pokud chybový kód není 0x1A.

MQFB_IMS_NACK_1A_REASON_FIRST
Má-li chybový kód hodnotu 0x1A, jsou kódy zpětné vazby generované systémem IMSv 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 systémem IMS, je-li kód chybového bajtu 0x1A

Kódy zpětné vazby mostu CICS: Následující kódy zpětné vazby mohou být generovány pomocí mostu CICS :

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
Most CICS byl nestandardně ukončen bez dokončení normálního zpracování chyb.

CHYBA MQFB_CICS_CCSID_ERROR
Identifikátor znakové sady není platný.

MQFB_CICS_CIH_ERROR
Struktura záhlaví informací 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 mostu CICS 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 mostu CICS 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.
- Při provádění jednotky práce došlo k nestandardstavu CICS .

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.

Kódy příčiny produktuWebSphere MQ: Pro zprávy o výjimce obsahuje produkt *Feedback* kód příčiny WebSphere 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 WebSphere MQ for z/OSnaleznete v tématu Kódy příčiny 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.

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 informačním záhlavím CICS MQCIH, za nímž následují data aplikace. Název formátu dat aplikace je dán polem *Format* ve struktuře MQCIH.

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 “Záhlaví MQDLH-Dead-letter” na stránce 319 . 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 “MQDH-záhlaví distribuce” na stránce 313 . 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, HP-UX, IBM i, Solaris, Linux, Okna, plus WebSphere MQ Klienti MQI 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 prvním záhlavím nebo zpráva nebude považována za zprávu přenosové cesty trasování. 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 převést pouze v systému 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 informačním záhlavím 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 MQGET s funkcí MQGMO_CONVERT naleznete v části “Formát (MQCHAR8)” na stránce 370 a “Formát ReplyTo(MQCHAR8)” na stránce 370.

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
Jedná se o řetězec proměnné IMS , který je řetězcem ve tvaru `llzzccc`, kde:

`ll`
je 2bajtová délka pole určující celkovou délku řetězcové položky 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 dohromady, přičemž každý řetězec má tvar `llzzccc`. Mezi po sobě jdoucími řetězci proměnných IMS nesmí 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 431 . 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 zprávy odkazu" na stránce 507 . Zprávy tohoto formátu lze převést, je-li v rámci příkazu MQGET zadána volba MQGMO_CONVERT.

Tento formát je podporován v následujících prostředích: AIX, HP-UX, IBM i, Solaris, Linux, Okna, plus WebSphere MQ Klienti MQI 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 557 . 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.

V systému z/OSzadejte volbu MQGMO_CONVERT v rámci 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) nebo ne.

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 576 .

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.

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.
- Na volání MQGET není zadán parametr MQMO_MATCH_GROUP_ID *není* .

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 vlastní identifikátory skupin*; aplikace proto musí 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.
- Je-li MQPMO_LOGICAL_ORDER *není* 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 506 na stránce 356. 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.

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-Získat-volby zprávy” na stránce 337 .) 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 nerozdělí zprávu uprostřed struktury záhlaví produktu WebSphere 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 všechny struktury záhlaví produktu WebSphere MQ , které mohou být přítomny), 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 produktu WebSphere 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/OStato volba není podporována, pokud je zpráva vložena do fronty, která má typ 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/OStato volba není podporována, pokud je zpráva vložena do fronty, která má typ 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 851.

Počáteční hodnota tohoto pole je MQMF_NONE. Toto pole je ignorováno, pokud *Version* je menší než MQMD_VERSION_2.

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.[2]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 informace o zachovaných příručkách naleznete v popisu MQPMO_RETAIN v části "Volby MQPMO (MQLONG)" na stránce 467 .

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 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*

[2] *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 produktu WebSphere 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.

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:

- volání MQGET s parametrem *CorrelId* na 64bitových distribuovaných platformách, kde index *CorrelId* odstraňuje potřebu provedení skutečného sekvenčního skenování.
- volání MQGET pomocí volby *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 506 na stránce 356 .

Zadání hodnoty MQMI_NONE, protože identifikátor zprávy má stejný účinek jako *not* určující MQMO_MATCH_MSG_ID, to znamená *any* odpovídá identifikátoru zprávy.

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.

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 je hodnota MQMO_MATCH_MSG_SEQ_NUMBER *není* určena.

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 506 na stránce 356. 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.

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.

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.

- Na volání MQGET je hodnota MQMO_MATCH_OFFSET *není* určena.

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 506 na stránce 356. 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.

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.

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 jsou přechodné zprávy přeživší správcem front v rámci skupiny sdílení front, ale nepřežijí selhání prostředku Coupling Facility 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. 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 *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.

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* , jak je definováno ve správci front *destination* , který vlastní konkrétní 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 *Priority* . 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* , jak je definováno 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 poskytnuto 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 systému z/OSzpráva s hodnotou MsgSeqNumber of 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.

Název funkce PutAppl(MQCHAR28)
Jedná se o název aplikace, která vložila zprávu, a je součástí *kontextu původu* zprávy. Obsah se liší mezi platformami a může se lišit mezi verzemi.

Další informace o kontextu zprávy viz “Přehled pro MQMD” na stránce 384 a Kontext zprávy .

Formát hodnoty *PutApplName* závisí na hodnotě *PutApplType* a může se změnit z jednoho vydání na druhý. Změny jsou vzácné, ale stane se, pokud se změní prostředí.

Když správce front nastaví toto pole (to znamená pro všechny volby kromě MQPMO_SET_ALL_CONTEXT), nastaví pole na hodnotu určenou prostředím:

- V systému z/OSsprávce front používá:
  – V případě dávky z/OS osmiznakový název úlohy z karty JES JOB

  - Pro TSO se jedná o 7znakový identifikátor uživatele TSO.
  - Pro CICS, osmiznakový identifikátor Applid, následovaný čtyřmístným tranID
  - Pro systém IMSse jedná o 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 CICS, osmiznakový název úlohy inicializátoru kanálu následovaný osmiznakovým názvem modulu, který vkládá do fronty nedoručených zpráv, za nímž následuje 8znakový identifikátor úlohy.

  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.
- V systémech Windows používá správce front:
  - Pro aplikaci CICS , název transakce CICS
  - V případě aplikace non-CICS se nejvíce vpravo 28 znaků úplného názvu spustitelného souboru
- V systému IBM i správce front používá plně kvalifikované jméno úlohy.
- Na systémech UNIX správce front používá:
  - Pro aplikaci CICS , název transakce CICS
  - V případě aplikace non-CICS se produkt MQ zeptá operačního systému na název procesu. Tento název je vrácen jako název souboru programu bez úplné cesty. Poté produkt MQ umístí tento název procesu do deskriptoru MQMD.PutApplName následujícím způsobem:

    AIX
    Je-li název menší nebo roven 28 bajtům, bude název vložen do pravého místa s mezerami.

    Je-li název větší než 28 bajtů, bude vloženo nejvíce vlevo 28 bajtů názvu.

    Linux a Solaris
    Je-li název menší nebo roven 15 bajtům, pak je název vložen do pravého místa s mezerami.

    Je-li název větší než 15 bajtů, pak se vloží zleva 15 bajtů názvu, doplní se doprava mezerami.

    HP-UX
    Je-li název menší nebo roven 14 bajtům, pak je název vložen do pravého místa s mezerami.

    Je-li název větší než 14 bajtů, pak se vloží nejlevější 14 bajtů názvu, doplní se doprava mezerami.

    Pokud například spustíte produkt `/opt/mqm/samp/bin/amqsput QNAME QMNAME`, bude mít název PutApplnázev `'amqsput'`. V tomto poli MQCHAR28 je k dispozici 21 znaků pro výplň. Mějte na zřeteli, že úplná cesta včetně produktu `/opt/mqm/samp/bin` není obsažena v názvu PutAppl.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_ALL_CONTEXT. Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Nulový znak a následující znaky jsou správcem front převáděny na mezery. Není-li parametr MQPMO_SET_ALL_CONTEXT zadán, bude toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Typ PutAppl(MQLONG)

Jedná se o typ aplikace, který tuto zprávu vložila, a je součástí **kontextu původu** zprávy. Další informace o kontextu zprávy viz “Přehled pro MQMD” na stránce 384 a Kontext zprávy.

PutApplType může mít jeden z následujících standardních typů. Můžete také definovat své vlastní typy, ale pouze s hodnotami v rozsahu MQAT_USER_FIRST až MQAT_USER_LAST.

MQAT_AIX
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_BROKER
Broker.

MQAT_CICS
Transakce CICS .

MOST MQAT_CICS_BRIDGE
Most CICS .

MQAT_CICS_VSE
Transakce CICS/VSE .

MQAT_DOS
Aplikace klienta WebSphere MQ MQI v systému PC DOS.

MQAT_DQM
Distribuovaný agent správce front.

MQAT_GUARDIAN
Aplikace Tandem Guardian (stejná hodnota jako MQAT_NSK).

MQAT_IMS
Aplikace IMS .

MOST MQAT_IMS_BRIDGE
Most IMS .

MQAT_JAVA
Java.

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_NOTES_AGENT
Aplikace agenta Lotus Notes .

MQAT_NSK
HP Integrity NonStop Server .

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_QMGR
Správce front.

MQAT_UNIX
Aplikace UNIX .

MQAT_VOS
Aplikace Stratus VOS.

MQAT_WINDOWS
16bitová aplikace systému Windows .

POČ MQAT_WINDOWS_NT
32bitovou aplikaci systému Windows .

MQAT_WLM
Aplikace správce pracovní zátěže systému z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Aplikace z/OS .

VÝCHOZÍ HODNOTA MQAT_DEFAULT
Výchozí typ aplikace.

Jedná se o výchozí typ aplikace pro platformu, na které je aplikace spuštěna.

Poznámka: Hodnota této konstanty je specifická pro prostředí. Z tohoto důvodu vždy kompilujte aplikaci pomocí záhlaví, zahrnutí nebo souboru COPY, které odpovídají platformě, na které bude aplikace spuštěna.

MQAT_UNKNOWN
Tuto hodnotu použijte, chcete-li označit, 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 určena volba kontextu MQPMO_NO_CONTEXT).

Když je načtena zpráva, lze pro tuto hodnotu testovat *PutApplType* , aby se rozhodlo, zda má zpráva kontext (doporučuje se, že *PutApplType* není nikdy nastaven na hodnotu MQAT_NO_CONTEXT, a to aplikací pomocí MQPMO_SET_ALL_CONTEXT, pokud jsou některé z ostatních kontextových polí neprázdné).

Když správce front vygeneruje tyto informace v důsledku vložení aplikace, je pole nastaveno na hodnotu určenou prostředím. V systému IBM ije nastavena hodnota MQAT_OS400; správce front nikdy nepoužívá produkt MQAT_CICS v systému IBM i.

Pro volání MQPUT a MQPUT1 se jedná o vstupní/výstupní pole, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_ALL_CONTEXT. Není-li parametr MQPMO_SET_ALL_CONTEXT zadán, bude toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Toto je výstupní pole pro volání MQGET. Počáteční hodnota tohoto pole je MQAT_NO_CONTEXT.

PutDate (MQCHAR8)

Jedná se o datum vložení zprávy a je součástí **kontextu původu** zprávy. Další informace o kontextu zprávy viz “Přehled pro MQMD” na stránce 384 a Kontext zprávy.

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.

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 se jedná o vstupní/výstupní pole, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_ALL_CONTEXT. Obsah pole nekontroluje správce front, s tím rozdílem, že všechny informace, které následují za znakem null uvnitř pole, jsou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li parametr MQPMO_SET_ALL_CONTEXT zadán, bude 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 MQ_PUT_DATE_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 8 prázdných znaků v jiných programovacích jazycích.

PutTime (MQCHAR8)

Jedná se o čas, kdy byla zpráva vložena, a je součástí **kontextu původu** zprávy. Další informace o kontextu zprávy viz "Přehled pro MQMD" na stránce 384 a Kontext zprávy.

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 *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.

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 se jedná o vstupní/výstupní pole, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_ALL_CONTEXT. Správce front nekontroluje obsah pole, kromě toho, že všechny informace, které následují za znakem null uvnitř pole, budou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li parametr MQPMO_SET_ALL_CONTEXT zadán, bude 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 MQ_PUT_TIME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 8 prázdných znaků v jiných programovacích jazycích.

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 753).

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.

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)

Je-li vyžadována více než jeden typ zprávy sestavy nebo jsou potřeba jiné volby sestavy, mohou být tyto hodnoty:

- Přidáno společně (nepřidávejte stejnou konstantu více než jednou), nebo
- zkombinovat pomocí bitové operace OR (jestliže 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 přes běžné kanály zpráv, je zpráva o výjimce generována *pouze* , pokud může být akce určená odesílající aplikací pro chybový stav 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 se generuje sestava COA, když zpráva dosáhne fronty IMS (potvrzení přijaté od IMS) a ne při vložení zprávy do fronty mostu MQ . To znamená, že pokud není produkt IMS aktivní, nebude generována žádná sestava COA, dokud nebude spuštěn systém IMS a do fronty IMS bude zařazena zpráva do fronty.

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/OSje to 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* provedena úspěšně. 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 přepnout na sestavy aktivity, je možné zapnout sestavy pomocí ukončení přeletu rozhraní API poskytnutého 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 nastavení volby sestavy na libovolné uživatelské zprávě, i když jsou zpracovány jiným správcem front než verze 6.0 nebo novější.
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 851.

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 požadována jedna nebo více těchto zpráv sestavy a *není* přijata, nelze použít obrácené pořadí, protože mohlo dojít k jedné 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_COD_ *, 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.

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
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 , ke kterému se zpráva vztahuje.
PutDate	Datum, kdy se odešle zpráva hlášení
PutTime	Čas odeslání zprávy sestavy

Pole v MQMD	Použitá hodnota
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 853.

Jedná se o výstupní pole pro volání MQGET a vstupní pole pro volání MQPUT a MQPUT1 . Počáteční hodnota tohoto pole je MQRO_NONE.

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.

UserIdentifier (MQCHAR12)

Tato část je součástí **kontextu identity** zprávy. Další informace o kontextu zprávy viz "Přehled pro MQMD" na stránce 384 a Kontext zprávy .

UserIdentifier uvádí identifikátor uživatele aplikace, která je původcem zprávy. Správce front považuje tyto informace za znaková data, ale nedefinuje její formát.

Po přijetí zprávy použijte příkaz *UserIdentifier* v poli *AlternateUserId* u parametru *ObjDesc* následné operace MQOPEN nebo MQPUT1 k provedení kontroly autorizace pro uživatele produktu *UserIdentifier* namísto aplikace, která má otevřeno.

Když správce front vygeneruje tyto informace pro volání MQPUT nebo MQPUT1 , postupujte takto:

- V systému z/OSsprávce front používá parametr *AlternateUserId* z parametru *ObjDesc* volání MQOPEN nebo MQPUT1 , pokud byla zadána volba MQOO_ALTERNATE_USER_AUTHORITY nebo MQPMO_ALTERNATE_USER_AUTHORITY. Nebyla-li příslušná volba uvedena, správce front použije identifikátor uživatele určený z prostředí.
- V jiných prostředích správce front vždy používá identifikátor uživatele určený z prostředí.

Když je identifikátor uživatele určen z prostředí:

- V systému z/OSsprávce front používá:
  - Pro MVS (dávka), identifikátor uživatele z karty JES JOB nebo spuštěná úloha
  - Pro TSO se identifikátor uživatele šířený do úlohy během odeslání úlohy

  - Pro CICS, identifikátor uživatele přidružený k úloze
  - Pro 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, které *have* vydalo 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
- V systému IBM isprávce front používá název profilu uživatele přidruženého k úloze aplikace.
- Na systémech UNIX správce front používá:
  - Přihlašovací jméno aplikace
  - Efektivní identifikátor uživatele procesu, pokud není k dispozici přihlášení
  - Identifikátor uživatele přidružený k transakci, pokud je aplikace transakce CICS .
- Na systémech Windows používá správce front prvních 12 znaků jména uživatele přihlášeného k přihlášení.

Toto pole je obvykle výstupní pole generované správcem front, ale pro volání MQPUT nebo MQPUT1 můžete toto pole zadat jako vstupní/výstupní pole a zadat pole UserIdentification místo toho, aby tyto informace mohly generovat správce front. Zadejte buď MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT v parametru PutMsg, a určete 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, je-li v parametru *PutMsgOpts* zadáno MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT, je-li zadán. Jakékoli informace, které následují za znakem null uvnitř pole, budou vyřazeny. Správce front převede znak null a všechny následující znaky na mezery. Není-li parametr MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT zadán, bude toto pole na vstupu ignorováno a je to pole pouze pro výstup.

Po úspěšném dokončení volání MQPUT nebo MQPUT1 bude toto pole obsahovat *UserIdentifier* , která byla přenesena spolu se zprávou, pokud byla vložena do fronty. To bude hodnota *UserIdentifier* , která je uchována se zprávou, je-li zachována (viz popis příkazu MQPMO_RETAIN pro více podrobností o zachovaných publikacích), ale nepoužívá se jako *UserIdentifier* , když je zpráva odeslána jako publikace odběratelům, protože poskytují hodnotu k přepsání *UserIdentifier* 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 hodnotou MQ_USER_ID_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 12 prázdných znaků v jiných programovacích jazycích.

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 produktu WebSphere MQ V6.0 a novějších a klientech WebSphere MQ MQI 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.

Počáteční hodnoty a deklarace jazyka pro MQMD

Tabulka 515. Počáteční hodnoty polí v deskriptoru MQMD pro MQMD

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQM_STRUCT	'MD'
Version	MQMD_VERSION_1	1
Report	MQRO_NONE	0
MsgType	MQM_DATAGRAM	8
Expiry	MQEI_UNLIMITED	-1
Feedback	MQFB_NONE	0
Encoding	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	MQCCSI_Q_MGR	0
Format	MQFMT_NONE	Mezery
Priority	MQPRI_PRIORITY_AS_Q_DEF	-1
Persistence	MQPER_PERSISTENCE_AS_Q_DEF	2
MsgId	MQMI_NONE	Hodnoty null
CorrelId	MQCI_NONE	Hodnoty null
BackoutCount	Není	0
ReplyToQ	Není	Nulový řetězec nebo prázdné znaky
ReplyToQMgr	Není	Nulový řetězec nebo prázdné znaky
UserIdentifier	Není	Nulový řetězec nebo prázdné znaky
AccountingToken	MQACT_NONE	Hodnoty null

Tabulka 515. Počáteční hodnoty polí v deskriptoru MQMD pro MQMD (pokračování)

Název pole	Název konstanty	Hodnota konstanty
ApplIdentityData	Není	Nulový řetězec nebo prázdné znaky
PutApplType	MQAT_NO_CONTEXT	0
PutApplName	Není	Nulový řetězec nebo prázdné znaky
PutDate	Není	Nulový řetězec nebo prázdné znaky
PutTime	Není	Nulový řetězec nebo prázdné znaky
ApplOriginData	Není	Nulový řetězec nebo prázdné znaky
GroupId	MQGI_NONE	Hodnoty null
MsgSeqNumber	Není	1
Offset	Není	0
MsgFlags	MQMF_NONE	0
OriginalLength	MQOL_UNDEFINED	-1

Notes:

1. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyce C-proměnná makraHodnota MQMD_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQMD MyMD = {MQMD_DEFAULT};
```

Deklarace C

```
typedef struct tagMQMD MQMD;
struct tagMQMD {
  MQCHAR4   StrucId;           /* Structure identifier */
  MQLONG    Version;           /* Structure version number */
  MQLONG    Report;            /* Options for report messages */
  MQLONG    MsgType;           /* Message type */
  MQLONG    Expiry;            /* Message lifetime */
  MQLONG    Feedback;          /* Feedback or reason code */
  MQLONG    Encoding;          /* Numeric encoding of message data */
  MQLONG    CodedCharSetId;    /* Character set identifier of message
                                  data */
  MQCHAR8   Format;            /* Format name of message data */
  MQLONG    Priority;          /* Message priority */
  MQLONG    Persistence;       /* Message persistence */
  MQBYTE24  MsgId;             /* Message identifier */
  MQBYTE24  CorrelId;          /* Correlation identifier */
  MQLONG    BackoutCount;      /* Backout counter */
  MQCHAR48  ReplyToQ;          /* Name of reply queue */
  MQCHAR48  ReplyToQMgr;       /* Name of reply queue manager */
  MQCHAR12  UserIdentifier;    /* User identifier */
  MQBYTE32  AccountingToken;   /* Accounting token */
  MQCHAR32  ApplIdentityData;  /* Application data relating to
                                  identity */
  MQLONG    PutApplType;       /* Type of application that put the
                                  message */
```

```
  MQCHAR28  PutApplName;       /* Name of application that put the
                                  message */
  MQCHAR8   PutDate;           /* Date when message was put */
  MQCHAR8   PutTime;           /* Time when message was put */
  MQCHAR4   ApplOriginData;    /* Application data relating to origin */
  MQBYTE24  GroupId;           /* Group identifier */
  MQLONG    MsgSeqNumber;      /* Sequence number of logical message
                                  within group */
  MQLONG    Offset;            /* Offset of data in physical message
                                  from start of logical message */
  MQLONG    MsgFlags;          /* Message flags */
  MQLONG    OriginalLength;    /* Length of original message */
};
```

Deklarace COBOL

```
**   MQMD structure
  10 MQMD.
**    Structure identifier
   15 MQMD-STRUCID          PIC X(4).
**    Structure version number
   15 MQMD-VERSION          PIC S9(9) BINARY.
**    Options for report messages
   15 MQMD-REPORT           PIC S9(9) BINARY.
**    Message type
   15 MQMD-MSGTYPE          PIC S9(9) BINARY.
**    Message lifetime
   15 MQMD-EXPIRY           PIC S9(9) BINARY.
**    Feedback or reason code
   15 MQMD-FEEDBACK         PIC S9(9) BINARY.
**    Numeric encoding of message data
   15 MQMD-ENCODING         PIC S9(9) BINARY.
**    Character set identifier of message data
   15 MQMD-CODEDCHARSETID   PIC S9(9) BINARY.
**    Format name of message data
   15 MQMD-FORMAT           PIC X(8).
**    Message priority
   15 MQMD-PRIORITY         PIC S9(9) BINARY.
**    Message persistence
   15 MQMD-PERSISTENCE      PIC S9(9) BINARY.
**    Message identifier
   15 MQMD-MSGID            PIC X(24).
**    Correlation identifier
   15 MQMD-CORRELID         PIC X(24).
**    Backout counter
   15 MQMD-BACKOUTCOUNT     PIC S9(9) BINARY.
**    Name of reply queue
   15 MQMD-REPLYTOQ         PIC X(48).
**    Name of reply queue manager
   15 MQMD-REPLYTOQMGR      PIC X(48).
**    User identifier
   15 MQMD-USERIDENTIFIER   PIC X(12).
**    Accounting token
   15 MQMD-ACCOUNTINGTOKEN  PIC X(32).
**    Application data relating to identity
   15 MQMD-APPLIDENTITYDATA PIC X(32).
**    Type of application that put the message
   15 MQMD-PUTAPPLTYPE      PIC S9(9) BINARY.
**    Name of application that put the message
   15 MQMD-PUTAPPLNAME      PIC X(28).
**    Date when message was put
   15 MQMD-PUTDATE          PIC X(8).
**    Time when message was put
   15 MQMD-PUTTIME          PIC X(8).
**    Application data relating to origin
   15 MQMD-APPLORIGINDATA   PIC X(4).
**    Group identifier
   15 MQMD-GROUPID          PIC X(24).
**    Sequence number of logical message within group
   15 MQMD-MSGSEQNUMBER     PIC S9(9) BINARY.
**    Offset of data in physical message from start of logical message
   15 MQMD-OFFSET           PIC S9(9) BINARY.
**    Message flags
   15 MQMD-MSGFLAGS         PIC S9(9) BINARY.
**    Length of original message
   15 MQMD-ORIGINALLENGTH   PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQMD based,
  3 StrucId          char(4),       /* Structure identifier */
  3 Version          fixed bin(31), /* Structure version number */
  3 Report           fixed bin(31), /* Options for report messages */
  3 MsgType          fixed bin(31), /* Message type */
  3 Expiry           fixed bin(31), /* Message lifetime */
  3 Feedback         fixed bin(31), /* Feedback or reason code */
  3 Encoding         fixed bin(31), /* Numeric encoding of message
                                       data */
  3 CodedCharSetId   fixed bin(31), /* Character set identifier of
                                       message data */
  3 Format           char(8),       /* Format name of message data */
  3 Priority         fixed bin(31), /* Message priority */
  3 Persistence      fixed bin(31), /* Message persistence */
  3 MsgId            char(24),      /* Message identifier */
  3 CorrelId         char(24),      /* Correlation identifier */
  3 BackoutCount     fixed bin(31), /* Backout counter */
  3 ReplyToQ         char(48),      /* Name of reply queue */
  3 ReplyToQMgr      char(48),      /* Name of reply queue manager */
  3 UserIdentifier   char(12),      /* User identifier */
  3 AccountingToken  char(32),      /* Accounting token */
  3 ApplIdentityData char(32),      /* Application data relating to
                                       identity */
  3 PutApplType      fixed bin(31), /* Type of application that put the
                                       message */
  3 PutApplName      char(28),      /* Name of application that put the
                                       message */
  3 PutDate          char(8),       /* Date when message was put */
  3 PutTime          char(8),       /* Time when message was put */
  3 ApplOriginData   char(4),       /* Application data relating to
                                       origin */
  3 GroupId          char(24),      /* Group identifier */
  3 MsgSeqNumber     fixed bin(31), /* Sequence number of logical
                                       message within group */
  3 Offset           fixed bin(31), /* Offset of data in physical
                                       message from start of logical
                                       message */
  3 MsgFlags         fixed bin(31), /* Message flags */
  3 OriginalLength   fixed bin(31); /* Length of original message */
```

Deklarace High Level Assembler

```
MQMD                   DSECT
MQMD_STRUCID           DS   CL4   Structure identifier
MQMD_VERSION           DS   F     Structure version number
MQMD_REPORT            DS   F     Options for report messages
MQMD_MSGTYPE           DS   F     Message type
MQMD_EXPIRY            DS   F     Message lifetime
MQMD_FEEDBACK          DS   F     Feedback or reason code
MQMD_ENCODING          DS   F     Numeric encoding of message data
MQMD_CODEDCHARSETID    DS   F     Character set identifier of message
*                                 data
MQMD_FORMAT            DS   CL8   Format name of message data
MQMD_PRIORITY          DS   F     Message priority
MQMD_PERSISTENCE       DS   F     Message persistence
MQMD_MSGID             DS   XL24  Message identifier
MQMD_CORRELID          DS   XL24  Correlation identifier
MQMD_BACKOUTCOUNT      DS   F     Backout counter
MQMD_REPLYTOQ          DS   CL48  Name of reply queue
MQMD_REPLYTOQMGR       DS   CL48  Name of reply queue manager
MQMD_USERIDENTIFIER    DS   CL12  User identifier
MQMD_ACCOUNTINGTOKEN   DS   XL32  Accounting token
MQMD_APPLIDENTITYDATA  DS   CL32  Application data relating to identity
MQMD_PUTAPPLTYPE       DS   F     Type of application that put the
*                                 message
MQMD_PUTAPPLNAME       DS   CL28  Name of application that put the
*                                 message
MQMD_PUTDATE           DS   CL8   Date when message was put
MQMD_PUTTIME           DS   CL8   Time when message was put
MQMD_APPLORIGINDATA    DS   CL4   Application data relating to origin
MQMD_GROUPID           DS   XL24  Group identifier
MQMD_MSGSEQNUMBER      DS   F     Sequence number of logical message
*                                 within group
MQMD_OFFSET            DS   F     Offset of data in physical message
*                                 from start of logical message
MQMD_MSGFLAGS          DS   F     Message flags
```

```
MQMD_ORIGINALLENGTH    DS   F     Length of original message
*
MQMD_LENGTH            EQU  *-MQMD
                       ORG  MQMD
MQMD_AREA              DS   CL(MQMD_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQMD
  StrucId          As String*4  'Structure identifier'
  Version          As Long      'Structure version number'
  Report           As Long      'Options for report messages'
  MsgType          As Long      'Message type'
  Expiry           As Long      'Message lifetime'
  Feedback         As Long      'Feedback or reason code'
  Encoding         As Long      'Numeric encoding of message data'
  CodedCharSetId   As Long      'Character set identifier of message'
                                'data'
  Format           As String*8  'Format name of message data'
  Priority         As Long      'Message priority'
  Persistence      As Long      'Message persistence'
  MsgId            As MQBYTE24  'Message identifier'
  CorrelId         As MQBYTE24  'Correlation identifier'
  BackoutCount     As Long      'Backout counter'
  ReplyToQ         As String*48 'Name of reply queue'
  ReplyToQMgr      As String*48 'Name of reply queue manager'
  UserIdentifier   As String*12 'User identifier'
  AccountingToken  As MQBYTE32  'Accounting token'
  ApplIdentityData As String*32 'Application data relating to identity'
  PutApplType      As Long      'Type of application that put the'
                                'message'
  PutApplName      As String*28 'Name of application that put the'
                                'message'
  PutDate          As String*8  'Date when message was put'
  PutTime          As String*8  'Time when message was put'
  ApplOriginData   As String*4  'Application data relating to origin'
  GroupId          As MQBYTE24  'Group identifier'
  MsgSeqNumber     As Long      'Sequence number of logical message'
                                'within group'
  Offset           As Long      'Offset of data in physical message'
                                'from start of logical message'
  MsgFlags         As Long      'Message flags'
  OriginalLength   As Long      'Length of original message'
End Type
```

MQMDE-Rozšíření deskriptoru zpráv

Následující tabulka shrnuje pole ve struktuře.

Tabulka 516. Pole v MQMDE

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQMDE	StrucLength
Encoding	Číselné kódování dat za MQMDE	Kódování
CodedCharSetId	Identifikátor znakové sady dat, která následuje MQMDE	CodedCharSetId
Format	Název formátu dat, která následuje MQMDE	Formát
Flags	Obecné příznaky	Příznaky
GroupId	Identifikátor skupiny	GroupId
MsgSeqNumber	Pořadové číslo logické zprávy v rámci skupiny	MsgSeqNumber

Tabulka 516. Pole v MQMDE (pokračování)		
Pole	**Popis**	**Téma**
Offset	Posunutí dat ve fyzické zprávě od začátku logické zprávy	Offset
MsgFlags	Příznaky zprávy	MsgFlags
OriginalLength	Délka původní zprávy	OriginalLength

Přehled pro MQMDE

Dostupnost: Všechny systémy WebSphere MQ a klienti produktu WebSphere MQ , kteří jsou připojeni k těmto systémům.

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: MQFMT_MD_EXTENSION.

Znaková sada a kódování: Data v MQMDE musí být ve znakové sadě a kódování lokálního správce front; tyto jsou předány atributem správce front *CodedCharSetId* a MQENC_NATIVE pro programovací jazyk C.

Nastavte znakovou sadu a kódování MQMDE do polí *CodedCharSetId* a *Encoding* 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.

Poznámka: V systému Windowsse aplikace kompilované pomocí Micro Focus COBOL používají hodnotu MQENC_NATIVE, která se liší od kódování správce front. Ačkoli číselná pole ve struktuře MQMD v rámci volání MQPUT, MQPUT1a MQGET musí být v kódování Micro Focus COBOL, musí být číselná pole ve struktuře MQMDE ve struktuře správce front. Tato hodnota je dána MQENC_NATIVE pro programovací jazyk C a má hodnotu 546.

Použití: Aplikace, které používají version-2 MQMD, 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

MQMDE zadaný na voláních MQPUT a MQPUT1: V případě volání MQPUT a MQPUT1 , pokud aplikace poskytuje version-1 MQMD, může aplikace volitelně připojit data zprávy k datům zprávy MQMDE a nastavit pole *Format* v MQMD na MQFMT_MD_EXTENSION tak, aby bylo zřejmé, 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 518 na stránce 435.

Pokud aplikace poskytuje version-2 MQMD *and* předpony dat zprávy aplikace s MQMDE, struktury se zpracují tak, jak jsou zobrazeny v Tabulka 517 na stránce 432.

Tabulka 517. Akce správce front, je-li hodnota MQMDE zadaná v MQPUT nebo MQPUT1 pro MQMDE			
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

Tabulka 517. Akce správce front, je-li hodnota MQMDE zadaná v MQPUT nebo MQPUT1 pro MQMDE (pokračování)

Verze MQMD	Hodnoty polí version-2	Hodnoty odpovídajících polí v MQMDE	Akce provedená správcem front
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
Poznámka: Pokud v systému z/OSaplikace specifikuje MQMD version-1 s MQMDE, ověřuje správce front hodnotu MQMDE pouze v případě, že má fronta *IndexType* MQIT_GROUP_ID.			

Je tu jeden speciální případ. Pokud aplikace používá version-2 MQMD k vložení zprávy, která je segment (tj. příznak MQMF_SEGMENT nebo MQMF_LAST_SEGMENT je nastaven) a název formátu v MQMD je MQFMT_DEAD_LETTER_HEADER, 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 do ekvivalentních polí ve výstupu MQMDE na výstupu z volání MQPUT a MQPUT1 ; pokud aplikace tyto výstupní hodnoty vyžaduje, musí použít version-2 MQMD.

MQMDE vrácený voláním MQGET: Při volání MQGET, pokud aplikace poskytuje MQMD version-1 , předpony správce front vrátí zprávu vrácenou s MQMDE, ale pouze, pokud jedno nebo více polí v MQMDE má nevýchozí hodnotu. Správce front nastaví pole *Format* v MQMD na hodnotu MQFMT_MD_EXTENSION, aby označilo, že je přítomen 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 volání MQGET vrátí hodnotu MQMDE, data v MQMDE se obvykle nacházejí 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 517 na stránce 432 , kde jsou uvedeny okolnosti, které mohou být příčinou).
- Zpráva byla přijata ze vzdáleného správce front připojeného pomocí spojení TCP a přijímací agent kanálu zpráv (MCA) nebyl správně nastaven.

Poznámka: V systému Windowsse aplikace kompilované pomocí Micro Focus COBOL používají hodnotu MQENC_NATIVE, která se liší od kódování správce front (viz výše).

MQMDE ve zprávách v přenosových frontách: Zprávy v přenosových frontách mají předponu struktury MQXQH, která obsahuje v něm version-1 MQMD. Je možné, že se nachází také MQMDE, umístěná mezi strukturou MQXQH a daty zprávy aplikace, ale obvykle se vyskytuje pouze tehdy, když jedno nebo více polí v MQMDE má nevýchozí hodnotu.

Další struktury záhlaví 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 pro MQMDE

Struktura MQMDE obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ , kteří jsou připojeni k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

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 zprávy” na stránce 383 .

Počáteční hodnota tohoto pole je MQENC_NATIVE.

Příznaky (MQLONG)

Lze zadat následující příznak:

MQMDEF_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQMDEF_NONE.

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 zprávy” na stránce 383 .

Počáteční hodnota tohoto pole je MQFMT_NONE.

GroupId (MQBYTE24)

Viz pole *GroupId* popsané v části “MQMD-deskriptor zprávy” na stránce 383. Počáteční hodnota tohoto pole je MQGI_NONE.

MsgFlags (MQLONG)

Viz pole *MsgFlags* popsané v části “MQMD-deskriptor zprávy” na stránce 383. Počáteční hodnota tohoto pole je MQMF_NONE.

Počet MsgSeqNumber (MQLONG)

Viz pole *MsgSeqNumber* popsané v části “MQMD-deskriptor zprávy” na stránce 383. Počáteční hodnota tohoto pole je 1.

Offset (MQLONG)

Viz pole *Offset* popsané v části “MQMD-deskriptor zprávy” na stránce 383. Počáteční hodnota tohoto pole je 0.

OriginalLength (MQLONG)

Viz pole *OriginalLength* popsané v části “MQMD-deskriptor zprávy” na stránce 383. Počáteční hodnota tohoto pole je MQOL_UNDEFINED.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQMDE

Tabulka 518. Počáteční hodnoty polí v MQMDE pro MQMDE

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQM_STRUCTION_ID	'MDE¬'
Version	MQMDE_VERSION_2	2
StrucLength	MQMDE_LENGTH_2	72
Encoding	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQMDEF_NONE	0
GroupId	MQGI_NONE	Hodnoty null

Tabulka 518. Počáteční hodnoty polí v MQMDE pro MQMDE (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
MsgSeqNumber	Není	1
Offset	Není	0
MsgFlags	MQMF_NONE	0
OriginalLength	MQOL_UNDEFINED	-1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQMDE_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQMDE MyMDE = {MQMDE_DEFAULT};
```

Deklarace C

```
typedef struct tagMQMDE MQMDE;
struct tagMQMDE {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    StrucLength;     /* Length of MQMDE structure */
  MQLONG    Encoding;        /* Numeric encoding of data that follows
                                MQMDE */
  MQLONG    CodedCharSetId;  /* Character-set identifier of data that
                                follows MQMDE */
  MQCHAR8   Format;          /* Format name of data that follows
                                MQMDE */
  MQLONG    Flags;           /* General flags */
  MQBYTE24  GroupId;         /* Group identifier */
  MQLONG    MsgSeqNumber;    /* Sequence number of logical message
                                within group */
  MQLONG    Offset;          /* Offset of data in physical message from
                                start of logical message */
  MQLONG    MsgFlags;        /* Message flags */
  MQLONG    OriginalLength;  /* Length of original message */
};
```

Deklarace COBOL

```
**   MQMDE structure
  10 MQMDE.
**    Structure identifier
   15 MQMDE-STRUCID        PIC X(4).
**    Structure version number
   15 MQMDE-VERSION        PIC S9(9) BINARY.
**    Length of MQMDE structure
   15 MQMDE-STRUCLENGTH    PIC S9(9) BINARY.
**    Numeric encoding of data that follows MQMDE
   15 MQMDE-ENCODING       PIC S9(9) BINARY.
**    Character-set identifier of data that follows MQMDE
   15 MQMDE-CODEDCHARSETID PIC S9(9) BINARY.
**    Format name of data that follows MQMDE
   15 MQMDE-FORMAT         PIC X(8).
**    General flags
   15 MQMDE-FLAGS          PIC S9(9) BINARY.
**    Group identifier
   15 MQMDE-GROUPID        PIC X(24).
**    Sequence number of logical message within group
   15 MQMDE-MSGSEQNUMBER   PIC S9(9) BINARY.
**    Offset of data in physical message from start of logical message
   15 MQMDE-OFFSET         PIC S9(9) BINARY.
**    Message flags
   15 MQMDE-MSGFLAGS       PIC S9(9) BINARY.
```

```
**    Length of original message
   15 MQMDE-ORIGINALLENGTH PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQMDE based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 StrucLength    fixed bin(31), /* Length of MQMDE structure */
  3 Encoding       fixed bin(31), /* Numeric encoding of data that
                                     follows MQMDE */
  3 CodedCharSetId fixed bin(31), /* Character-set identifier of data
                                     that follows MQMDE */
  3 Format         char(8),       /* Format name of data that follows
                                     MQMDE */
  3 Flags          fixed bin(31), /* General flags */
  3 GroupId        char(24),      /* Group identifier */
  3 MsgSeqNumber   fixed bin(31), /* Sequence number of logical message
                                     within group */
  3 Offset         fixed bin(31), /* Offset of data in physical message
                                     from start of logical message */
  3 MsgFlags       fixed bin(31), /* Message flags */
  3 OriginalLength fixed bin(31); /* Length of original message */
```

Deklarace High Level Assembler

```
MQMDE                 DSECT
MQMDE_STRUCID         DS   CL4   Structure identifier
MQMDE_VERSION         DS   F     Structure version number
MQMDE_STRUCLENGTH     DS   F     Length of MQMDE structure
MQMDE_ENCODING        DS   F     Numeric encoding of data that follows
*                                MQMDE
MQMDE_CODEDCHARSETID  DS   F     Character-set identifier of data that
*                                follows MQMDE
MQMDE_FORMAT          DS   CL8   Format name of data that follows MQMDE
MQMDE_FLAGS           DS   F     General flags
MQMDE_GROUPID         DS   XL24  Group identifier
MQMDE_MSGSEQNUMBER    DS   F     Sequence number of logical message
*                                within group
MQMDE_OFFSET          DS   F     Offset of data in physical message from
*                                start of logical message
MQMDE_MSGFLAGS        DS   F     Message flags
MQMDE_ORIGINALLENGTH  DS   F     Length of original message
*
MQMDE_LENGTH          EQU  *-MQMDE
                      ORG  MQMDE
MQMDE_AREA            DS   CL(MQMDE_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQMDE
  StrucId        As String*4 'Structure identifier'
  Version        As Long     'Structure version number'
  StrucLength    As Long     'Length of MQMDE structure'
  Encoding       As Long     'Numeric encoding of data that follows'
                             'MQMDE'
  CodedCharSetId As Long     'Character-set identifier of data that'
                             'follows MQMDE'
  Format         As String*8 'Format name of data that follows MQMDE'
  Flags          As Long     'General flags'
  GroupId        As MQBYTE24 'Group identifier'
  MsgSeqNumber   As Long     'Sequence number of logical message within'
                             'group'
  Offset         As Long     'Offset of data in physical message from'
                             'start of logical message'
  MsgFlags       As Long     'Message flags'
  OriginalLength As Long     'Length of original message'
End Type
```

MQMHBO-Popisovač zpráv pro volby vyrovnávací paměti

Následující tabulka shrnuje pole ve struktuře. MQMHBO struktura-zpracování zpráv obslužného programu pro vyrovnávací paměť

Tabulka 519. Pole v MQMHBO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby řízení akce MQMHBUF	Volby

Přehled pro MQMHBO

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ MQI.

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 (MQENC_NATIVE).

Pole pro MQMHBO

Parametry pro strukturu voleb vyrovnávací paměti-pole

Struktura MQMHBO obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

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 hodnotu. V případě potřeby mohou být použity následující hodnoty:

- Přidáno společně (nepřidávejte stejnou konstantu více než jednou), nebo
- zkombinovat pomocí bitové operace OR (jestliže 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQMHBO

Popisovač zprávy pro strukturu vyrovnávací paměti-počáteční hodnoty

Tabulka 520. Počáteční hodnoty polí v MQMHBO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQMHBO_STRUCTION_ID	'MHBO'
Version	MQMHBO_VERSION_1	1
Options	MQMHBO_PROPERTIES_IN_MQRFH2	

Notes:

1. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyce C-proměnná makraHodnota MQMHBO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQMHBO MyMHBO = {MQMHBO_DEFAULT};
```

Deklarace C
Struktura popisovače zpráv do struktury voleb vyrovnávací paměti-deklarace jazyka C

```
typedef struct tagMQMHBO MQMHBO;
struct tagMQMHBO {
  MQCHAR4  StrucId;        /* Structure identifier */
  MQLONG   Version;        /* Structure version number */
  MQLONG   Options;        /* Options that control the action of
                              MQMHBUF */
};
```

Deklarace COBOL
Popisovač zprávy pro strukturu voleb vyrovnávací paměti-deklarace jazyka COBOL

```
**   MQMHBO structure
  10 MQMHBO.
**    Structure identifier
   15 MQMHBO-STRUCID               PIC X(4).
**    Structure version number
   15 MQMHBO-VERSION               PIC S9(9) BINARY.
**    Options that control the action of MQMHBUF
   15 MQMHBO-OPTIONS               PIC S9(9) BINARY.
```

Deklarace PL/I
Struktura popisovače zpráv pro strukturu voleb vyrovnávací paměti-deklarace jazyka PL/I

```
Dcl
 1 MQMHBO based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
```

```
  3 Options        fixed bin(31), /* Options that control the action
                                     of MQMHBUF */
```

Deklarace High Level Assembler
Struktura popisovače zpráv pro strukturu vyrovnávací paměti-deklarace jazyka assembler

```
MQMHBO                  DSECT
MQMHBO_STRUCID          DS   CL4  Structure identifier
MQMHBO_VERSION          DS   F    Structure version number
MQMHBO_OPTIONS          DS   F    Options that control the
*                                 action of MQMHBUF
MQMHBO_LENGTH           EQU  *-MQMHBO
MQMHBO_AREA             DS   CL(MQMHBO_LENGTH)
```

MQOD-Deskriptor objektu

Následující tabulka shrnuje pole ve struktuře.

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
ObjectType	Typ objektu	ObjectType
ObjectName	Název objektu	ObjectName
ObjectQMgrName	Název správce front objektu	ObjectQMgrName
DynamicQName	Název dynamické fronty	DynamicQName
AlternateUserId	Alternativní identifikátor uživatele	AlternateUserid
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQOD_VERSION_2.		
RecsPresent	Počet přítomných záznamů objektů	RecsPresent
KnownDestCount	Počet úspěšně otevřených lokálních front	KnownDestCount
UnknownDestCount	Počet úspěšně otevřených vzdálených front	UnknownDestCount
InvalidDestCount	Počet front, které se nepodařilo otevřít	InvalidDestCount
ObjectRecOffset	Odsazení prvního záznamu objektu od začátku MQOD	PosunutíObjectRec
ResponseRecOffset	Odstup prvního záznamu odezvy od začátku MQOD	PosunutíResponseRec
ObjectRecPtr	Adresa prvního záznamu objektu	ObjectRecPtr
ResponseRecPtr	Adresa prvního záznamu odezvy	ResponseRecPtr
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQOD_VERSION_3.		
AlternateSecurityId	Alternativní identifikátor zabezpečení	AlternateSecurityId
ResolvedQName	Rozlišený název fronty	ResolvedQName
ResolvedQMgrName	Vyřešený název správce front	ResolvedQMgrNázev
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQOD_VERSION_4.		
ObjectString	Název dlouhého objektu	ObjectString
SelectionString	Řetězec výběru	SelectionString

Pole	Popis	Téma
ResObjectString	Vyřešený dlouhý název objektu	ResObjectString
ResolvedType	Typ vyřešeného objektu	ResolvedType

Přehled pro MQOD

Dostupnost: Všechny systémy WebSphere MQ spolu s klienty WebSphere MQ MQI připojenými k těmto systémům.

Účel: Struktura MQOD se používá k určení názvu objektu podle názvu. Platné jsou tyto typy objektů:

- Fronta nebo distribuční seznam
- Seznam názvů
- Definice procesu
- Správce front
- Téma

Struktura je vstupním/výstupním parametrem na voláních MQOPEN a MQPUT1 .

Verze: Aktuální verze MQOD je MQOD_VERSION_4. Aplikace, které chcete v rámci portu mezi několika prostředími, musí zajistit, aby požadovaná verze MQOD byla podporována ve všech dotčených prostředích. Pole, která existují pouze v posledních verzích struktury, jsou identifikována jako taková v popisech, které následují.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejaktuálnější verzi MQOD, která je podporována prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQOD_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí 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 aplikaci MQOD musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQOD

Struktura MQOD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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) systému 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 systému 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.

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 otevřenému 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.

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.

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 přítomno, je nastaveno *pouze* , pokud je parametr *CompCode* na volání MQOPEN nebo MQPUT1 MQCC_OK nebo MQCC_WARNING; *není* nastaven, pokud je parametr *CompCode* MQCC_FAILED.

Toto je výstupní pole. Počáteční hodnota tohoto pole je 0. Toto pole je ignorováno, pokud *Version* je menší než MQOD_VERSION_1.

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.

ObjectName (MQCHAR48)

Jedná se o lokální název objektu, jak je definován ve správci front identifikovaném příkazem *ObjectQMgrName*. 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. Použijte znak null pro označení konce významných dat v názvu; hodnoty null a libovolné znaky následující za ním jsou považovány za mezery. V označeném prostředí platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OS:
  - Vyhněte se názvům, které začínají nebo končí podtržítkem; nemohou být zpracovány operacemi a řídicími panely.
  - Znak procentní části má speciální význam pro RACF. Je-li jako externí správce zabezpečení použit modul RACF , nesmí názvy obsahovat žádné procento. Pokud ano, tyto názvy nejsou při použití generických profilů RACF zahrnuty do žádných kontrol zabezpečení.

- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procento, pokud jsou zadány v příkazech, uzavřeny do uvozovek. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry na voláních.

Úplný název tématu může být sestaven ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o způsobu použití těchto dvou polí naleznete v tématu "Použití řetězců témat" na stránce 538.

Pro typy označených objektů platí následující body:

- Je-li *ObjectName* názvem modelové fronty, správce front vytvoří dynamickou frontu s atributy modelové fronty a vrátí se do pole *ObjectName* název vytvořené fronty. Modelovou frontu lze zadat pouze v rámci volání MQOPEN. Modelová fronta není na volání MQPUT1 platná.
- Je-li *ObjectName* jméno alias fronty s TARGTYPE (TOPIC), provede se kontrola zabezpečení na pojmenované frontě alias; to je normální, když se používají alias fronty. Po úspěšném dokončení kontroly zabezpečení bude volání MQOPEN pokračovat a bude se chovat jako volání MQOPEN v objektu MQOT_TOPIC; to zahrnuje provedení kontroly zabezpečení proti objektu administrativního tématu.
- Pokud produkt *ObjectName* a *ObjectQMgrName* identifikují sdílenou frontu vlastněnou skupinou sdílení front, do níž patří lokální správce front, nesmí být v lokálním správci front také definice fronty se stejným názvem. Existuje-li taková definice (lokální fronta, alias fronta, vzdálená fronta nebo modelová fronta), volání selže s kódem příčiny MQRC_OBJECT_NOT_UNIQUE.
- Je-li otevíraný objekt distribuční seznam (tedy *RecsPresent* je přítomný a větší než nula), *ObjectName* musí být prázdný nebo řetězec s hodnotou null. Není-li tato podmínka splněna, volání selže s kódem příčiny MQRC_OBJECT_NAME_ERROR.
- Je-li *ObjectType* MQOT_Q_MGR, platí speciální pravidla; v tomto případě musí být název zcela prázdný až na první znak null nebo na konec pole.

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_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v 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 443). 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, kteráje vlastněn 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 *nepatří*), bude *ObjectQMgrName* zmutované jméno skupiny sdílení front. Můžete použít název správce front, který patří do této skupiny, ale to může zpozdit zprávu v případě, že daný 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.

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).

Je-li vybrána jakákoli technika, použijte jeden z produktů *ObjectRecOffset* a *ObjectRecPtr* ; volání selže s kódem příčiny MQRC_OBJECT_RECORS_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.

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.

Pro uvedení záznamů objektů můžete použít buď *ObjectRecPtr* nebo *ObjectRecOffset* , ale ne obojí; podrobnosti najdete v popisu pole *ObjectRecOffset* . 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.

ObjectString (MQCHARV)
Pole ObjectString určuje dlouhý název objektu.

Uvádí dlouhý název objektu, který se má použít. Toto pole je odkazováno pouze na určité hodnoty *ObjectType*a je ignorován pro všechny ostatní hodnoty. Podrobnosti o tom, které hodnoty označují, že toto pole je použito, viz popis *ObjectType* .

Pokud je parametr *ObjectString* zadán nesprávně, v souladu s popisem 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 počáteční hodnoty v rámci struktury MQCHARV.

Úplný název tématu může být sestaven ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o způsobu použití těchto dvou polí naleznete v tématu “Použití řetězců témat” na stránce 538.

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* .

MQO_NAMELIST
Seznam jmen. Název objektu se nachází v poli *ObjectName* .

PROCES 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 může být sestaven ze dvou různých polí: *ObjectName* a *ObjectString*.

Podrobnosti o způsobu použití těchto dvou polí naleznete v tématu “Použití řetězců témat” na stránce 538.

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

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.

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.

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. Je-li fronta vlastněna jinou skupinou sdílení front, může být produktem *ResolvedQName* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (vrácená hodnota vrácená hodnotou je určena 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.

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.

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.

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 odpovědí lze poskytovat stejným způsobem jako záznamy objektů, a to buď zadáním posunu v *ResponseRecOffset*, nebo zadáním adresy v *ResponseRecPtr*; podrobnosti o tom, jak to provést, viz popis *ObjectRecOffset* výše. 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.

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.

Pro uvedení záznamů odpovědí použijte buď *ResponseRecPtr* nebo *ResponseRecOffset* , ale ne oba; podrobnosti naleznete v popisu pole *ResponseRecOffset* . 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.

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 proměnné délky" na stránce 271 , 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.

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.

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.

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 WebSphere 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.

Počáteční hodnoty a deklarace jazyka pro MQOD

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQOD_STRUCTION_ID	'OD¬¬'
Version	MQOD_VERSION_1	1
ObjectType	MQOT_Q	1
ObjectName	Není	Nulový řetězec nebo prázdné znaky
ObjectQMgrName	Není	Nulový řetězec nebo prázdné znaky
DynamicQName	Není	'CSQ.*' v systému z/OS; 'AMQ.*' jinak
AlternateUserId	Není	Nulový řetězec nebo prázdné znaky
RecsPresent	Není	0
KnownDestCount	Není	0
UnknownDestCount	Není	0
InvalidDestCount	Není	0
ObjectRecOffset	Není	0
ResponseRecOffset	Není	0
ObjectRecPtr	Není	Nulový ukazatel nebo bajty null
ResponseRecPtr	Není	Nulový ukazatel nebo bajty null
AlternateSecurityId	MQSID_NONE	Hodnoty null
ResolvedQName	Není	Nulový řetězec nebo prázdné znaky
ResolvedQMgrName	Není	Nulový řetězec nebo prázdné znaky
ObjectString	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	Jak je definováno pro MQCHARV
SelectionString	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	Jak je definováno pro MQCHARV
ResObjectString	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	Jak je definováno pro MQCHARV
ResolvedType	MQOT_NONE	0

Název pole	Název konstanty	Hodnota konstanty

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.
3. V programovacím jazyce C-proměnná makraHodnota MQOD_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQOD MyOD = {MQOD_DEFAULT};
```

Deklarace C

```
typedef struct tagMQOD MQOD;
struct tagMQOD {
  MQCHAR4    StrucId;              /* Structure identifier */
  MQLONG     Version;              /* Structure version number */
  MQLONG     ObjectType;           /* Object type */
  MQCHAR48   ObjectName;           /* Object name */
  MQCHAR48   ObjectQMgrName;       /* Object queue manager name */
  MQCHAR48   DynamicQName;         /* Dynamic queue name */
  MQCHAR12   AlternateUserId;      /* Alternate user identifier */
  /* Ver:1 */
  MQLONG     RecsPresent;          /* Number of object records present */
  MQLONG     KnownDestCount;       /* Number of local queues opened
                                      successfully */
  MQLONG     UnknownDestCount;     /* Number of remote queues opened
                                      successfully */
  MQLONG     InvalidDestCount;     /* Number of queues that failed to
                                      open */
  MQLONG     ObjectRecOffset;      /* Offset of first object record from
                                      start of MQOD */
  MQLONG     ResponseRecOffset;    /* Offset of first response record
                                      from start of MQOD */
  MQPTR      ObjectRecPtr;         /* Address of first object record */
  MQPTR      ResponseRecPtr;       /* Address of first response record */
  /* Ver:2 */
  MQBYTE40   AlternateSecurityId;  /* Alternate security identifier */
  MQCHAR48   ResolvedQName;        /* Resolved queue name */
  MQCHAR48   ResolvedQMgrName;     /* Resolved queue manager name */
  /* Ver:3 */
  MQCHARV    ObjectString;         /* Object Long name */
  MQCHARV    SelectionString;      /* Message Selector */
  MQCHARV    ResObjectString;      /* Resolved Long object name*/
  MQLONG     ResolvedType          /* Alias queue resolved
                                      oject type */
  /* Ver:4 */
};
```

Deklarace COBOL

```
** MQOD structure
 10  MQOD.
** Structure identifier
 15  MQOD-STRUCID                    PIC X(4).
** Structure version number
 15  MQOD-VERSION                    PIC S9(9) BINARY.
** Object type
 15  MQOD-OBJECTTYPE                 PIC S9(9) BINARY.
** Object name
 15  MQOD-OBJECTNAME                 PIC X(48).
** Object queue manager name
 15  MQOD-OBJECTQMGRNAME             PIC X(48).
** Dynamic queue name
 15  MQOD-DYNAMICQNAME               PIC X(48).
** Alternate user identifier
 15  MQOD-ALTERNATEUSERID            PIC X(12).
** Number of object records present
 15  MQOD-RECSPRESENT                PIC S9(9) BINARY.
** Number of local queues opened successfully
```

```
 15  MQOD-KNOWNDESTCOUNT              PIC S9(9) BINARY.
** Number of remote queues opened successfully
 15  MQOD-UNKNOWNDESTCOUNT            PIC S9(9) BINARY.
** Number of queues that failed to open
 15  MQOD-INVALIDDESTCOUNT            PIC S9(9) BINARY.
** Offset of first object record from start of MQOD
 15  MQOD-OBJECTRECOFFSET             PIC S9(9) BINARY.
** Offset of first response record from start of MQOD
 15  MQOD-RESPONSERECOFFSET           PIC S9(9) BINARY.
** Address of first object record
 15  MQOD-OBJECTRECPTR                POINTER.
** Address of first response record
 15  MQOD-RESPONSERECPTR              POINTER.
** Alternate security identifier
 15  MQOD-ALTERNATESECURITYID         PIC X(40).
** Resolved queue name
 15  MQOD-RESOLVEDQNAME               PIC X(48).
** Resolved queue manager name
 15  MQOD-RESOLVEDQMGRNAME            PIC X(48).
** Object Long name
 15 MQOD-OBJECTSTRING.
** Address of variable length string
    20  MQOD-OBJECTSTRING-VSPTR        POINTER.
** Offset of variable length string
    20  MQOD-OBJECTSTRING-VSOFFSET     PIC S9(9) BINARY.
** size of buffer
    20  MQOD-OBJECTSTRING-VSBUFSIZE    PIC S9(9) BINARY.
** Length of variable length string
    20  MQOD-OBJECTSTRING-VSLENGTH     PIC S9(9) BINARY.
** CCSID of variable length string
    20  MQOD-OBJECTSTRING-VSCCSID      PIC S9(9) BINARY.
** Message Selector
 15 MQOD-SELECTIONSTRING.
** Address of variable length string
    20  MQOD-SELECTIONSTRING-VSPTR     POINTER.
** Offset of variable length string
    20  MQOD-SELECTIONSTRING-VSOFFSET  PIC S9(9) BINARY.
** size of buffer
    20  MQOD-SELECTIONSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
    20  MQOD-SELECTIONSTRING-VSLENGTH  PIC S9(9) BINARY.
** CCSID of variable length string
    20  MQOD-SELECTIONSTRING-VSCCSID   PIC S9(9) BINARY.
** Resolved Long object name
 15 MQOD-RESOBJECTSTRING.
** Address of variable length string
    20  MQOD-RESOBJECTSTRING-VSPTR     POINTER.
** Offset of variable length string
    20  MQOD-RESOBJECTSTRING-VSOFFSET  PIC S9(9) BINARY.
** size of buffer
    20  MQOD-RESOBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
** Length of variable length string
    20  MQOD-RESOBJECTSTRING-VSLENGTH  PIC S9(9) BINARY.
** CCSID of variable length string
    20  MQOD-RESOBJECTSTRING-VSCCSID   PIC S9(9) BINARY.
** Alias queue resolved object type
    15  MQOD-RESOLVEDTYPE              PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQOD based,
  3 StrucId             char(4),       /* Structure identifier */
  3 Version             fixed bin(31), /* Structure version number */
  3 ObjectType          fixed bin(31), /* Object type */
  3 ObjectName          char(48),      /* Object name */
  3 ObjectQMgrName      char(48),      /* Object queue manager name */
  3 DynamicQName        char(48),      /* Dynamic queue name */
  3 AlternateUserId     char(12),      /* Alternate user identifier */
  3 RecsPresent         fixed bin(31), /* Number of object records
                                          present */
  3 KnownDestCount      fixed bin(31), /* Number of local queues opened
                                          successfully */
  3 UnknownDestCount    fixed bin(31), /* Number of remote queues opened
                                          successfully */
  3 InvalidDestCount    fixed bin(31), /* Number of queues that failed to
                                          open */
  3 ObjectRecOffset     fixed bin(31), /* Offset of first object record
                                          from start of MQOD */
  3 ResponseRecOffset   fixed bin(31), /* Offset of first response record
```

```
                                      from start of MQOD */
  3 ObjectRecPtr        pointer,       /* Address of first object record */
  3 ResponseRecPtr      pointer,       /* Address of first response
                                          record */
  3 AlternateSecurityId char(40),      /* Alternate security identifier */
  3 ResolvedQName       char(48),      /* Resolved queue name */
  3 ResolvedQMgrName    char(48),      /* Resolved queue manager name */
  3 ObjectString,                      /* Object Long name */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31), /* CCSID of variable length string */
  3 SelectionString,                   /* Message Selection */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31), /* CCSID of variable length string */
  3 ResObjectString,                   /* Resolved Long object name */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31), /* CCSID of variable length string */
    3 ResolvedType        fixed bin(31); /* Alias queue resolved object type */
```

Deklarace High Level Assembler

```
MQOD                           DSECT
MQOD_STRUCID                   DS   CL4   Structure identifier
MQOD_VERSION                   DS   F     Structure version number
MQOD_OBJECTTYPE                DS   F     Object type
MQOD_OBJECTNAME                DS   CL48  Object name
MQOD_OBJECTQMGRNAME            DS   CL48  Object queue manager name
MQOD_DYNAMICQNAME              DS   CL48  Dynamic queue name
MQOD_ALTERNATEUSERID           DS   CL12  Alternate user identifier
MQOD_RECSPRESENT               DS   F     Number of object records present
MQOD_KNOWNDESTCOUNT            DS   F     Number of local queues opened
*                                         successfully
MQOD_UNKNOWNDESTCOUNT          DS   F     Number of remote queues opened
*                                         successfully
MQOD_INVALIDDESTCOUNT          DS   F     Number of queues that failed to
*                                         open
MQOD_OBJECTRECOFFSET           DS   F     Offset of first object record from
*                                         start of MQOD
MQOD_RESPONSERECOFFSET         DS   F     Offset of first response record
*                                         from start of MQOD
MQOD_OBJECTRECPTR              DS   F     Address of first object record
MQOD_RESPONSERECPTR            DS   F     Address of first response record
MQOD_ALTERNATESECURITYID       DS   XL40  Alternate security identifier
MQOD_RESOLVEDQNAME             DS   CL48  Resolved queue name
MQOD_RESOLVEDQMGRNAME          DS   CL48  Resolved queue manager name
MQOD_OBJECTSTRING              DS   F     Object Long name
MQOD_OBJECTSTRING_VSPTR        DS   F     Address of variable length string
MQOD_OBJECTSTRING_VSOFFSET     DS   F     Offset of variable length string
MQOD_OBJECTSTRING_VSBUFSIZE    DS   F     size of buffer
MQOD_OBJECTSTRING_VSLENGTH     DS   F     Length of variable length string
MQOD_OBJECTSTRING_VSCCSID      DS   F     CCSID of variable length string
MQOD_OBJECTSTRING_LENGTH       EQU  *- MQOD_OBJECTSTRING
                               ORG  MQOD_OBJECTSTRING
MQOD_OBJECTSTRING_AREA         DS   CL(MQOD_OBJECTSTRING_LENGTH)
*
MQOD_SELECTIONSTRING           DS   F     Message Selector
MQOD_SELECTIONSTRING_VSPTR     DS   F     Address of variable length string
MQOD_SELECTIONSTRING_VSOFFSET  DS   F     Offset of variable length string
MQOD_SELECTIONSTRING_VSBUFSIZE DS   F     size of buffer
MQOD_SELECTIONSTRING_VSLENGTH  DS   F     Length of variable length string
MQOD_SELECTIONSTRING_VSCCSID   DS   F     CCSID of variable length string
MQOD_SELECTIONSTRING_LENGTH    EQU  *- MQOD_SELECTIONSTRING
                               ORG  MQOD_SELECTIONSTRING
MQOD_SELECTIONSTRING_AREA      DS   CL(MQOD_SELECTIONSTRING_LENGTH)
*
MQOD_RESOBJECTSTRING           DS   F     Resolved Long object name
MQOD_RESOBJECTSTRING_VSPTR     DS   F     Address of variable length string
MQOD_RESOBJECTSTRING_VSOFFSET  DS   F     Offset of variable length string
MQOD_RESOBJECTSTRING_VSBUFSIZE DS   F     size of buffer
MQOD_RESOBJECTSTRING_VSLENGTH  DS   F     Length of variable length string
MQOD_RESOBJECTSTRING_VSCCSID   DS   F     CCSID of variable length string
MQOD_RESOBJECTSTRING_LENGTH    EQU  *- MQOD_RESOBJECTSTRING
```

```
                                ORG  MQOD_RESOBJECTSTRING
MQOD_RESOBJECTSTRING_AREA       DS   CL(MQOD_RESOBJECTSTRING_LENGTH)
MQOD_RESOLVEDTYPE               DS   F     Alias queue object resolved type
*
MQOD_LENGTH                     EQU  *-MQOD
                                ORG  MQOD
MQOD_AREA                       DS   CL(MQOD_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQOD
  StrucId             As String*4  'Structure identifier'
  Version             As Long      'Structure version number'
  ObjectType          As Long      'Object type'
  ObjectName          As String*48 'Object name'
  ObjectQMgrName      As String*48 'Object queue manager name'
  DynamicQName        As String*48 'Dynamic queue name'
  AlternateUserId     As String*12 'Alternate user identifier'
  RecsPresent         As Long      'Number of object records present'
  KnownDestCount      As Long      'Number of local queues opened'
                                   'successfully'
  UnknownDestCount    As Long      'Number of remote queues opened'
                                   'successfully'
  InvalidDestCount    As Long      'Number of queues that failed to'
                                   'open'
  ObjectRecOffset     As Long      'Offset of first object record from'
                                   'start of MQOD'
  ResponseRecOffset   As Long      'Offset of first response record'
                                   'from start of MQOD'
  ObjectRecPtr        As MQPTR     'Address of first object record'
  ResponseRecPtr      As MQPTR     'Address of first response record'
  AlternateSecurityId As MQBYTE40  'Alternate security identifier'
  ResolvedQName       As String*48 'Resolved queue name'
  ResolvedQMgrName    As String*48 'Resolved queue manager name'
End Type
```

MQOR-Záznam objektu

Následující tabulka shrnuje pole ve struktuře.

Tabulka 521. Pole v MQOR		
Pole	**Popis**	**Téma**
ObjectName	Název objektu	ObjectName
ObjectQMgrName	Název správce front objektu	ObjectQMgrName

Přehled pro MQOR

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windowsa klienti WebSphere MQ MQI, kteří jsou připojeni k těmto systémům.

Účel: Použijte strukturu MQOR k určení názvu fronty a názvu správce front jedné cílové fronty. MQOR je vstupní struktura pro volání MQOPEN a MQPUT1 .

Znaková sada a kódování: Data v MQOR musí být ve znakové sadě, která je dána atributem správce front *CodedCharSetId* a kódováním lokálního správce front uvedeného MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur v rámci volání MQOPEN můžete otevřít seznam front; tento seznam se nazývá *distribuční seznam*. Každá zpráva pomocí manipulátoru fronty vráceného tímto voláním MQOPEN je umístěna do každé z front v seznamu za předpokladu, že byla fronta úspěšně otevřena.

Pole pro MQOR

Struktura MQOR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

Počáteční hodnoty a deklarace jazyka pro MQOR

Tabulka 522. Počáteční hodnoty polí v MQOR pro MQOR

Název pole	Název konstanty	Hodnota konstanty
ObjectName	Není	Nulový řetězec nebo prázdné znaky
ObjectQMgrName	Není	Nulový řetězec nebo prázdné znaky

Notes:

1. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyce C-proměnná makraObjekt MQOR_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQOR MyOR = {MQOR_DEFAULT};
```

Deklarace C

```
typedef struct tagMQOR MQOR;
struct tagMQOR {
  MQCHAR48  ObjectName;      /* Object name */
  MQCHAR48  ObjectQMgrName;  /* Object queue manager name */
};
```

Deklarace COBOL

```
**   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

```
dcl
 1 MQOR based,
  3 ObjectName     char(48), /* Object name */
  3 ObjectQMgrName char(48); /* Object queue manager name */
```

Deklarace jazyka Visual Basic

```
Type MQOR
  ObjectName     As String*48 'Object name'
  ObjectQMgrName As String*48 'Object queue manager name'
End Type
```

MQPD-Deskriptor vlastnosti

Následující tabulka shrnuje pole ve struktuře.

Tabulka 523. Pole v MQPD

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby
Support	Požadovaná podpora pro vlastnost zprávy	Podpora
Context	Kontext zprávy, do kterého patří vlastnost	Kontext
CopyOptions	Kopírovat volby do které vlastnosti patří	CopyOptions

Přehled pro MQPD

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS a klienti WebSphere MQ MQI.

Účel: Produkt **MQPD** se používá k definování atributů vlastnosti. Struktura je vstupním/výstupním parametrem na volání MQSETMP a výstupním parametrem volání MQINQMP.

Znaková sada a kódování: Data v souboru **MQPD** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole pro MQPD

Struktura MQPD obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

Kontext (MQLONG)

Tato vlastnost popisuje kontext zprávy, do níž daná vlastnost patří.

When a queue manager receives a message containing a WebSphere 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 WebSphere MQ verze 7.0 je vlastnost přidružená k uživatelskému kontextu 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 pouze výstupní pole pro rozpoznané vlastnosti produktu WebSphere MQ ; produkt WebSphere MQ nastavuje příslušnou hodnotu.

When a queue manager receives a message containing a WebSphere 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, a pokud potřebujete více než jednu, mohou být tyto hodnoty:

- sečíst (žádnou konstantu nepřičítejte vícekrát než jednou) nebo
- zkombinovat pomocí bitové operace OR (jestliže 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 není požadována žádná z výše uvedených voleb, 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.

Volby (MQLONG)

Hodnota musí být:

MQPD_NONE
Nejsou zadány žádné volby

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

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**.

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 produktem WebSphere MQ; podpora pro všechny ostatní vlastnosti je volitelná.

Pole je automaticky nastaveno na správnou hodnotu, je-li správce front znám vlastnost definované produktem WebSphere MQ. Není-li vlastnost rozpoznána, je objekt MQPD_SUPPORT_OPTIONAL přiřazen. When a queue manager receives a message containing a WebSphere MQ-defined property that the queue manager recognizes as being incorrect, the queue manager corrects the value of the *Support* field.

Při nastavení vlastnosti definované produktem WebSphere MQpomocí volání MQSETMP na obslužné rutiny zprávy, kde byla nastavena volba MQCMHO_NO_VALIDATION, se jako vstupní pole stane *Support* . To umožňuje aplikaci umístit vlastnost definovaná uživatelem produktu WebSphere MQse 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 přidružena k vlastnostem, které nejsou definovanými vlastnostmi produktu WebSphere MQ.

Pokud správce front produktu WebSphere MQ verze 7.0 , který podporuje vlastnosti zpráv, 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 k vlastnostem, které nejsou definované produktem WebSphere MQ.

POŽADOVÁNA PODPORA MQPD_SUPPORT_REQUIRED
Podpora pro vlastnost je povinná. Zpráva je odmítnuta správcem front, který nepodporuje vlastnost definovaná uživatelem produktu WebSphere 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 definované produktem WebSphere 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.

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**.

Počáteční hodnoty a deklarace jazyka pro MQPD

Tabulka 524. Počáteční hodnoty polí v MQPD

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY OBJEKTU MQPD_BEAN	'PD'
Version	MQPD_VERSION_1	1
Options	MQPD_NONE	0
Support	PODPORA MQPD_SUPPORT_OPTIONAL	0
Context	MQPD_NO_CONTEXT	0
CopyOptions	MQCOPY_DEFAULT	0

Notes:

1. V programovacím jazyce C obsahuje proměnná makra MQPD_DEFAULT hodnoty uvedené výše. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQPD MyPD = {MQPD_DEFAULT};
```

Deklarace C

```
typedef struct tagMQPD MQPD;
struct tagMQPD {
  MQCHAR4  StrucId;     /* Structure identifier */
  MQLONG   Version;     /* Structure version number */
  MQLONG   Options;     /* Options that control the action of
                           MQSETMP and MQINQMP */
  MQLONG   Support;     /* Property support option */
  MQLONG   Context;     /* Property context */
  MQLONG   CopyOptions; /* Property copy options */
};
```

Deklarace COBOL

```
**   MQPD structure
  10 MQPD.
**    Structure identifier
   15 MQPD-STRUCID PIC X(4).
**    Structure version number
   15 MQPD-VERSION PIC S9(9) BINARY.
**    Options that control the action of MQSETMP and
**    MQINQMP
   15 MQPD-OPTIONS PIC S9(9) BINARY.
**    Property support option
   15 MQPD-SUPPORT PIC S9(9) BINARY.
**    Property context
   15 MQPD-CONTEXT PIC S9(9) BINARY.
**    Property copy options
   15 MQPD-COPYOPTIONS PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQPD based,
   3 StrucId     char(4),       /* Structure identifier */
   3 Version     fixed bin(31), /* Structure version number */
   3 Options     fixed bin(31), /* Options that control the action
                                   of MQSETMP and MQINQMP */
   3 Support     fixed bin(31), /* Property support option */
```

```
  3 Context     fixed bin(31), /* Property context */
  3 CopyOptions fixed bin(31); /* Property copy options */
```

Deklarace High Level Assembler

```
MQPD             DSECT
MQPD_STRUCID     DS   CL4    Structure identifier
MQPD_VERSION     DS   F      Structure version number
MQPD_OPTIONS     DS   F      Options that control the
*                            action of MQSETMP and MQINQMP
MQPD_SUPPORT     DS   F      Property support option
MQPD_CONTEXT     DS   F      Property context
MQPD_COPYOPTIONS DS   F      Property copy options
MQPD_LENGTH      EQU  *-MQPD
MQPD_AREA        DS   CL(MQPD_LENGTH)
```

MQPMO-Volby vložení zprávy

Následující tabulka shrnuje pole ve struktuře.

Tabulka 525. Struktura MQPMO

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby, které řídí akce MQPUT a MQPUT1	Volby
Timeout	Vyhrazené	žurnálů
Context	Popisovač objektu vstupní fronty	Kontext
KnownDestCount	Počet zpráv odeslaných úspěšně do lokálních front	KnownDestCount
UnknownDestCount	Počet zpráv odeslaných úspěšně do vzdálených front	UnknownDestCount
InvalidDestCount	Počet zpráv, které nebylo možné odeslat	InvalidDestCount
ResolvedQName	Vyřešený název cílové fronty	ResolvedQName
ResolvedQMgrName	Vyřešený název správce cílové fronty	ResolvedQMgrNázev
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQPMO_VERSION_2.		
RecsPresent	Počet vložených záznamů zpráv nebo záznamů odpovědí	RecsPresent
PutMsgRecFields	Příznaky určující, která pole MQPMR jsou přítomna	PutMsgRecFields
PutMsgRecOffset	Odstup prvního záznamu vložení-zprávy od začátku MQPMO	PutMsgRecOffset
ResponseRecOffset	Odstup prvního záznamu odezvy od začátku MQPMO	PosunutíResponseRec
PutMsgRecPtr	Adresa záznamu prvního vložení zprávy	PutMsgRecPtr
ResponseRecPtr	Adresa prvního záznamu odezvy	ResponseRecPtr
Poznámka: Zbývající pole se ignorují, pokud *Version* je menší než MQPMO_VERSION_3.		
OriginalMsgHandle	Popisovač původní zprávy	OriginalMsg

Tabulka 525. Struktura MQPMO (pokračování)		
Pole	**Popis**	**Téma**
NewMsgHandle	Popisovač nové zprávy	NewMsgPopisovač
Action	Typ prováděné operace a vztah mezi původní zprávou určenou polem *OriginalMsgHandle* a novou zprávou zadanou pomocí pole *NewMsgHandle*	Akce
PubLevel	Úroveň odběru, na kterou je publikace zaměřena	PubLevel

Přehled pro MQPMO

Dostupnost: Všechny systémy WebSphere MQ a klienti produktu WebSphere MQ , kteří jsou připojeni k těmto systémům.

Účel: Struktura MQPMO umožňuje aplikaci určit volby, které řídí způsob vkládání zpráv do front nebo publikování na témata. Struktura je vstupním/výstupním parametrem na volání MQPUT a MQPUT1 .

Verze: Aktuální verze MQPMO je MQPMO_VERSION_3. Některá pole jsou k dispozici pouze v určitých verzích MQPMO. Pokud potřebujete portovat aplikace mezi několika prostředími, musíte se ujistit, že je verze MQPMO konzistentní ve všech prostředích. Pole, která existují pouze v určitých verzích struktury, jsou identifikována jako "MQPMO-Volby vložení zprávy" na stránce 460 a v popisech polí.

Soubory záhlaví, COPY a INCLUDE poskytnuté pro podporované programovací jazyky obsahují nejnověji verzi MQPMO, která je podporována prostředím, ale s počáteční hodnotou pole *Version* nastavenou na MQPMO_VERSION_1. Chcete-li použít pole, která nejsou přítomna ve struktuře version-1 , aplikace musí nastavit pole *Version* 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 daného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQPMO

Struktura MQPMO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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 a pak počáteční pole deskriptoru zpráv jsou vybrána takto:

Tabulka 526. 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
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 je 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 527. 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

Tabulka 527. Transformace popisovače zprávy sestavy (pokračování)	
Pole v MQMD	**Použitá hodnota**
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 je 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 vybrat uživatelská data zprávy sestavy, produkt WebSphere MQ konzultuje pole Sestava a Zpětná vazba ve výsledném deskriptoru MQMD a parametry Vyrovnávací paměť a BufferLength volání MQPUT nebo MQPUT1 .

- Je-li zpětná vazba MQFB_COA, MQFB_COD nebo MQFB_EXPIRATION, 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 se zobrazí v následující tabulce:

Tabulka 528. 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_DA TA	Není	Vyrovnávací paměť (prvních 100 bajtů)	Není
MQRO_COA_WITH_DA TA	Vyrovnávací paměť (prvních 100 bajtů)	Není	Není

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 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.

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.

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**.

Volby MQPMO (MQLONG)
Pole Volby řídí činnost volání **MQPUT** a **MQPUT1** .

Volba rozsahu. Můžete zadat libovolné volby MQPMO nebo žádnou z nich. Je-li požadována více než jedna volba, hodnoty, které uvedete pro volby, lze použít následujícími způsoby:

- Hodnoty lze přidat. Nepřidávejte stejnou konstantu více než jednou.
- Hodnoty lze kombinovat s použitím 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 viz popis volání “MQSUBRQ-Požadavek na odběr” na stránce 750 .

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 nežz/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 WebSphere MQ automaticky zvýší pořadové číslo zprávy.
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:

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.
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 aplikace klienta připojena ke správci front na úrovni dřívější než verze 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.

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**.

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 .

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 zadáním adresy v *PutMsgRecPtr*; podrobnosti o tom, jak to provést, najdete v poli *ObjectRecOffset* , které je popsáno v “MQOD-Deskriptor objektu” na stránce 440.

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.

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* pro uvedení záznamů vložení zpráv, ale ne obojí; podrobnosti najdete v popisu pole *PutMsgRecOffset* . 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.

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.

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. Je-li fronta vlastněna jinou skupinou sdílení front, může být produktem *ResolvedQName* název skupiny sdílení front nebo název správce front, který je členem skupiny sdílení front (vrácená hodnota vrácená hodnotou je určena 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.

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.

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 zadáním adresy v *ResponseRecPtr*; podrobnosti o tom, jak to provést, najdete v poli *ObjectRecOffset* , které je popsáno v “MQOD-Deskriptor objektu” na stránce 440. Nicméně, použijte ne více než jeden z *ResponseRecOffset* a *ResponseRecPtr*; volání selže s kódem příčiny MQRC_RESPONSE_RECORS_ERROR, pokud jsou oba 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.

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.

Pro uvedení záznamů odpovědí použijte buď *ResponseRecPtr* nebo *ResponseRecOffset* , ale ne oba; podrobnosti naleznete v popisu pole *ResponseRecOffset* . 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.

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.

Časový limit (MQLONG)

Jedná se o vyhrazené pole; jeho hodnota není významná. Počáteční hodnota tohoto pole je -1.

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.

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, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ MQI 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.

Počáteční hodnoty a deklarace jazyka pro MQPMO

Tabulka 529. Počáteční hodnoty polí v MQPMO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQPMO_STRUC_ID	'PMO¬'
Version	MQPMO_VERSION_1	1
Options	MQPMO_NONE	0
Timeout	Není	-1
Context	Není	0
KnownDestCount	Není	0
UnknownDestCount	Není	0
InvalidDestCount	Není	0
ResolvedQName	Není	Nulový řetězec nebo prázdné znaky
ResolvedQMgrName	Není	Nulový řetězec nebo prázdné znaky
RecsPresent	Není	0

Tabulka 529. Počáteční hodnoty polí v MQPMO (pokračování)

Název pole	Název konstanty	Hodnota konstanty
PutMsgRecFields	MQPMRF_NONE	0
PutMsgRecOffset	Není	0
ResponseRecOffset	Není	0
PutMsgRecPtr	Není	Nulový ukazatel nebo bajty null
ResponseRecPtr	Není	Nulový ukazatel nebo bajty null
OriginalMsgHandle	MQM_NONE	0
NewMsgHandle	MQM_NONE	0
Action	NOVÁ HODNOTA MQACTP_NEW	0
PubLevel	Není	9

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.
3. V programovacím jazyce C-proměnná makraHodnota MQPMO_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQPMO MyPMO = {MQPMO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQPMO MQPMO;
struct tagMQPMO {
  MQCHAR4   StrucId;            /* Structure identifier */
  MQLONG    Version;            /* Structure version number */
  MQLONG    Options;            /* Options that control the action of
                                   MQPUT and MQPUT1 */
  MQLONG    Timeout;            /* Reserved */
  MQHOBJ    Context;            /* Object handle of input queue */
  MQLONG    KnownDestCount;     /* Number of messages sent
                                   successfully to local queues */
  MQLONG    UnknownDestCount;   /* Number of messages sent
                                   successfully to remote queues */
  MQLONG    InvalidDestCount;   /* Number of messages that could not
                                   be sent */
  MQCHAR48  ResolvedQName;      /* Resolved name of destination
                                   queue */
  MQCHAR48  ResolvedQMgrName;   /* Resolved name of destination queue
                                   manager */
  /* Ver:1 */
  MQLONG    RecsPresent;        /* Number of put message records or
                                   response records present */
  MQLONG    PutMsgRecFields;    /* Flags indicating which MQPMR fields
                                   are present */
  MQLONG    PutMsgRecOffset;    /* Offset of first put message record
                                   from start of MQPMO */
  MQLONG    ResponseRecOffset;  /* Offset of first response record
                                   from start of MQPMO */
  MQPTR     PutMsgRecPtr;       /* Address of first put message
                                   record */
  MQPTR     ResponseRecPtr;     /* Address of first response record */
  /* Ver:2 */
  MQHMSG    OriginalMsgHandle;  /* Original message handle */
  MQHMSG    NewMsgHandle;       /* New message handle */
  MQLONG    Action;             /* The action being performed */
  MQLONG    PubLevel;           /* Subscription level */
  /* Ver:3 */
};
```

Deklarace COBOL

```
**   MQPMO structure
  10 MQPMO.
**    Structure identifier
   15 MQPMO-STRUCID           PIC X(4).
**    Structure version number
   15 MQPMO-VERSION           PIC S9(9) BINARY.
**    Options that control the action of MQPUT and MQPUT1
   15 MQPMO-OPTIONS           PIC S9(9) BINARY.
**    Reserved
   15 MQPMO-TIMEOUT           PIC S9(9) BINARY.
**    Object handle of input queue
   15 MQPMO-CONTEXT           PIC S9(9) BINARY.
**    Number of messages sent successfully to local queues
   15 MQPMO-KNOWNDESTCOUNT    PIC S9(9) BINARY.
**    Number of messages sent successfully to remote queues
   15 MQPMO-UNKNOWNDESTCOUNT  PIC S9(9) BINARY.
**    Number of messages that could not be sent
   15 MQPMO-INVALIDDESTCOUNT  PIC S9(9) BINARY.
**    Resolved name of destination queue
   15 MQPMO-RESOLVEDQNAME     PIC X(48).
**    Resolved name of destination queue manager
   15 MQPMO-RESOLVEDQMGRNAME  PIC X(48).
**    Number of put message records or response records present
   15 MQPMO-RECSPRESENT       PIC S9(9) BINARY.
**    Flags indicating which MQPMR fields are present
   15 MQPMO-PUTMSGRECFIELDS   PIC S9(9) BINARY.
**    Offset of first put message record from start of MQPMO
   15 MQPMO-PUTMSGRECOFFSET   PIC S9(9) BINARY.
**    Offset of first response record from start of MQPMO
   15 MQPMO-RESPONSERECOFFSET PIC S9(9) BINARY.
**    Address of first put message record
   15 MQPMO-PUTMSGRECPTR      POINTER.
**    Address of first response record
   15 MQPMO-RESPONSERECPTR    POINTER.
**    Original message handle
   15 MQPMO-ORIGINALMSGHANDLE PIC S9(18) BINARY.
**    New message handle
   15 MQPMO-NEWMSGHANDLE      PIC S9(18) BINARY.
**    The action being performed
   15 MQPMO-ACTION            PIC S9(9) BINARY.
**    Publish level
   15 MQPMO-PUBLEVEL          PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQPMO based,
  3 StrucId           char(4),       /* Structure identifier */
  3 Version           fixed bin(31), /* Structure version number */
  3 Options           fixed bin(31), /* Options that control the action
                                        of MQPUT and MQPUT1 */
  3 Timeout           fixed bin(31), /* Reserved */
  3 Context           fixed bin(31), /* Object handle of input queue */
  3 KnownDestCount    fixed bin(31), /* Number of messages sent
                                        successfully to local queues */
  3 UnknownDestCount  fixed bin(31), /* Number of messages sent
                                        successfully to remote queues */
  3 InvalidDestCount  fixed bin(31), /* Number of messages that could
                                        not be sent */
  3 ResolvedQName     char(48),      /* Resolved name of destination
                                        queue */
  3 ResolvedQMgrName  char(48),      /* Resolved name of destination
                                        queue manager */
  3 RecsPresent       fixed bin(31), /* Number of put message records or
                                        response records present */
  3 PutMsgRecFields   fixed bin(31), /* Flags indicating which MQPMR
                                        fields are present */
  3 PutMsgRecOffset   fixed bin(31), /* Offset of first put message
                                        record from start of MQPMO */
  3 ResponseRecOffset fixed bin(31), /* Offset of first response record
                                        from start of MQPMO */
  3 PutMsgRecPtr      pointer,       /* Address of first put message
                                        record */
  3 ResponseRecPtr    pointer,       /* Address of first response
                                        record */
```

```
  3 OriginalMsgHandle fixed bin(63), /* Original message handle */
  3 NewMsgHandle      fixed bin(63); /* New message handle */
  3 Action            fixed bin(31); /* The action being performed */
  3 PubLevel          fixed bin(31); /* Publish level */
```

Deklarace High Level Assembler

```
MQPMO                   DSECT
MQPMO_STRUCID           DS   CL4   Structure identifier
MQPMO_VERSION           DS   F     Structure version number
MQPMO_OPTIONS           DS   F     Options that control the action of
*                                  MQPUT and MQPUT1
MQPMO_TIMEOUT           DS   F     Reserved
MQPMO_CONTEXT           DS   F     Object handle of input queue
MQPMO_KNOWNDESTCOUNT    DS   F     Number of messages sent successfully
*                                  to local queues
MQPMO_UNKNOWNDESTCOUNT  DS   F     Number of messages sent successfully
*                                  to remote queues
MQPMO_INVALIDDESTCOUNT  DS   F     Number of messages that could not be
*                                  sent
MQPMO_RESOLVEDQNAME     DS   CL48  Resolved name of destination queue
MQPMO_RESOLVEDQMGRNAME  DS   CL48  Resolved name of destination queue
*                                  manager
MQPMO_RECSPRESENT       DS   F     Number of put message records or
*                                  response records present
MQPMO_PUTMSGRECFIELDS   DS   F     Flags indicating which MQPMR
*                                  fields are present
MQPMO_PUTMSGRECOFFSET   DS   F     Offset of first put message record
*                                  from start of MQPMO
MQPMO_RESPONSERECOFFSET DS   F     Offset of first response record
*                                  from start of MQPMO
MQPMO_PUTMSGRECPTR      DS   F     Address of first put message
*                                  record
MQPMO_RESPONSERECPTR    DS   F     Address of first response record
MQPMO_ORIGINALMSGHANDLE DS   D     Original message handle
MQPMO_NEWMSGHANDLE      DS   D     New message handle
MQPMO_ACTION            DS   F     The action being performed
MQPMO_PUBLEVEL          DS   F     Publish level
*
MQPMO_LENGTH            EQU  *-MQPMO
                        ORG  MQPMO
MQPMO_AREA              DS   CL(MQPMO_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQPMO
  StrucId           As String*4  'Structure identifier'
  Version           As Long      'Structure version number'
  Options           As Long      'Options that control the action of'
                                 'MQPUT and MQPUT1'
  Timeout           As Long      'Reserved'
  Context           As Long      'Object handle of input queue'
  KnownDestCount    As Long      'Number of messages sent successfully'
                                 'to local queues'
  UnknownDestCount  As Long      'Number of messages sent successfully'
                                 'to remote queues'
  InvalidDestCount  As Long      'Number of messages that could not be'
                                 'sent'
  ResolvedQName     As String*48 'Resolved name of destination queue'
  ResolvedQMgrName  As String*48 'Resolved name of destination queue'
                                 'manager'
  RecsPresent       As Long      'Number of put message records or'
                                 'response records present'
  PutMsgRecFields   As Long      'Flags indicating which MQPMR fields'
                                 'are present'
  PutMsgRecOffset   As Long      'Offset of first put message record'
                                 'from start of MQPMO'
  ResponseRecOffset As Long      'Offset of first response record from'
                                 'start of MQPMO'
  PutMsgRecPtr      As MQPTR     'Address of first put message record'
  ResponseRecPtr    As MQPTR     'Address of first response record'
End Type
```

záznam MQPMR-Put-message

Následující tabulka shrnuje pole ve struktuře.

Tabulka 530. Pole v MQPMR		
Pole	**Popis**	**Téma**
MsgId	Identifikátor zprávy	MsgId
CorrelId	Identifikátor korelace	CorrelId
GroupId	Identifikátor skupiny	GroupId
Feedback	Zpětná vazba nebo kód příčiny	Zpětná vazba
AccountingToken	Token evidence	AccountingToken

Přehled pro MQPMR

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Okna, plus klienti WebSphere MQ připojené k těmto systémům.

Účel: Při vkládání zprávy do distribučního seznamu použijte strukturu MQPMR k uvedení různých vlastností zprávy pro jedno místo určení. MQPMR je struktura vstupu/výstupu pro volání MQPUT a MQPUT1 .

Znaková sada a kódování: Data ve struktuře MQPMR musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front poskytnutého funkcí MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ , musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur na volání MQPUT nebo MQPUT1 můžete 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 *PutMsgRecFields* v MQPMO. Pole, která jsou přítomna ***se musí vyskytnout v následujícím pořadí***:

- *MsgId*
- *CorrelId*
- *GroupId*
- *Feedback*
- *AccountingToken*

Pole, která nejsou přítomna, nezabírají žádný prostor v záznamu.

Vzhledem k tomu, že MQPMR nemá pevné rozvržení, není v záhlaví, COPY a INCLUDE pro podporované programovací jazyky poskytnuta žádná definice. Programátor aplikace musí vytvořit deklaraci obsahující pole, která jsou vyžadována aplikací, a nastavit příznaky v produktu *PutMsgRecFields* tak, aby určovaly pole, která jsou přítomná.

Pole pro MQPMR

Struktura MQPMR obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

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. Informace o obsahu tohoto pole naleznete v popisu *AccountingToken* v části “MQMD-deskriptor zprávy” na stránce 383 .

Není-li toto pole k dispozici, bude použita hodnota v produktu MQMD.

Toto je vstupní 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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQPMR

Pro tuto strukturu nejsou definovány žádné počáteční hodnoty, protože v záhlaví, COPY a INCLUDE nejsou pro podporované programovací jazyky poskytnuty žádné deklarace struktury. Ukázková deklarace zobrazují, jak deklarovat strukturu, pokud jsou vyžadována všechna pole.

Deklarace C

```
typedef struct tagMQPMR MQPMR;
struct tagMQPMR {
  MQBYTE24  MsgId;            /* Message identifier */
  MQBYTE24  CorrelId;         /* Correlation identifier */
  MQBYTE24  GroupId;          /* Group identifier */
  MQLONG    Feedback;         /* Feedback or reason code */
  MQBYTE32  AccountingToken;  /* Accounting token */
};
```

Deklarace COBOL

```
**   MQPMR structure
  10 MQPMR.
**    Message identifier
   15 MQPMR-MSGID           PIC X(24).
**    Correlation identifier
   15 MQPMR-CORRELID        PIC X(24).
**    Group identifier
   15 MQPMR-GROUPID         PIC X(24).
**    Feedback or reason code
   15 MQPMR-FEEDBACK        PIC S9(9) BINARY.
**    Accounting token
   15 MQPMR-ACCOUNTINGTOKEN PIC X(32).
```

Deklarace PL/I

```
dcl
 1 MQPMR based,
  3 MsgId           char(24),      /* Message identifier */
  3 CorrelId        char(24),      /* Correlation identifier */
  3 GroupId         char(24),      /* Group identifier */
  3 Feedback        fixed bin(31), /* Feedback or reason code */
  3 AccountingToken char(32);      /* Accounting token */
```

Deklarace jazyka Visual Basic

```
Type MQPMR
  MsgId           As MQBYTE24 'Message identifier'
  CorrelId        As MQBYTE24 'Correlation identifier'
  GroupId         As MQBYTE24 'Group identifier'
  Feedback        As Long     'Feedback or reason code'
  AccountingToken As MQBYTE32 'Accounting token'
End Type
```

MQRFH-Pravidla a záhlaví formátování

Tento oddíl popisuje pravidla a formátovací záhlaví, jaká pole obsahuje, a počáteční hodnoty těchto polí.

Přehled pro MQRFH

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ MQI připojené k těmto systémům.

Účel: Struktura MQRFH definuje rozvržení pravidel a záhlaví formátování. Toto záhlaví použijte k odeslání řetězcových dat ve formě dvojic název/hodnota.

Název formátu: MQFMT_RF_HEADER.

Znaková sada a kódování: Pole ve struktuře MQRFH (včetně produktu *NameValueString*) jsou ve znakové sadě a kódování zadané v polích *CodedCharSetId* a *Encoding* 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 pro MQRFH

Struktura MQRFH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

CodedCharSetId (MQLONG)

Tato hodnota určuje identifikátor znakové sady dat, která následuje *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.

Kódování (MQLONG)

Určuje číselné kódování dat, která jsou následující: *NameValueString*; nevztahuje se na číselná data v samotné struktuře MQRFH.

Na základě volání MQPUT nebo MQPUT1 musí aplikace nastavit toto pole na hodnotu odpovídající datům.

Počáteční hodnota tohoto pole je MQENC_NATIVE.

Příznaky (MQLONG)

Je možné zadat následující:

MQRFH_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQRFH_NONE.

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.

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_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 *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.

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.

StrucLength (MQLONG)

Jedná se o délku struktury MQRFH v bajtech, včetně pole *NameValueString* na konci struktury. Délka *neobsahuje* 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQRFH

Tabulka 531. Počáteční hodnoty polí v MQRFH pro MQRFH

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQRFH_STRUCT	'RFH¬'
Version	MQRFH_VERSION_1	1
StrucLength	PEVNÉ PRODLOUŽENÍ MQRFH_STRUCLOTH_FIXED	32
Encoding	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQRFH_NONE	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQRFH_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRFH MyRFH = {MQRFH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQRFH MQRFH;
struct tagMQRFH {
  MQCHAR4  StrucId;         /* Structure identifier */
  MQLONG   Version;         /* Structure version number */
  MQLONG   StrucLength;     /* Total length of MQRFH including
                               NameValueString */
  MQLONG   Encoding;        /* Numeric encoding of data that follows
                               NameValueString */
  MQLONG   CodedCharSetId;  /* Character set identifier of data that
                               follows NameValueString */
  MQCHAR8  Format;          /* Format name of data that follows
                               NameValueString */
  MQLONG   Flags;           /* Flags */
};
```

Deklarace COBOL

```
**   MQRFH structure
  10 MQRFH.
**    Structure identifier
   15 MQRFH-STRUCID        PIC X(4).
**    Structure version number
   15 MQRFH-VERSION        PIC S9(9) BINARY.
**    Total length of MQRFH including NAMEVALUESTRING
   15 MQRFH-STRUCLENGTH    PIC S9(9) BINARY.
**    Numeric encoding of data that follows NAMEVALUESTRING
   15 MQRFH-ENCODING       PIC S9(9) BINARY.
**    Character set identifier of data that follows NAMEVALUESTRING
   15 MQRFH-CODEDCHARSETID PIC S9(9) BINARY.
**    Format name of data that follows NAMEVALUESTRING
   15 MQRFH-FORMAT         PIC X(8).
**    Flags
   15 MQRFH-FLAGS          PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQRFH based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 StrucLength    fixed bin(31), /* Total length of MQRFH including
                                     NameValueString */
  3 Encoding       fixed bin(31), /* Numeric encoding of data that
                                     follows NameValueString */
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                     that follows NameValueString */
  3 Format         char(8),       /* Format name of data that follows
                                     NameValueString */
  3 Flags          fixed bin(31); /* Flags */
```

Deklarace High Level Assembler

```
MQRFH                 DSECT
MQRFH_STRUCID         DS   CL4  Structure identifier
MQRFH_VERSION         DS   F    Structure version number
MQRFH_STRUCLENGTH     DS   F    Total length of MQRFH including
*                               NAMEVALUESTRING
MQRFH_ENCODING        DS   F    Numeric encoding of data that follows
*                               NAMEVALUESTRING
MQRFH_CODEDCHARSETID  DS   F    Character set identifier of data that
*                               follows NAMEVALUESTRING
MQRFH_FORMAT          DS   CL8  Format name of data that follows
*                               NAMEVALUESTRING
MQRFH_FLAGS           DS   F    Flags
*
MQRFH_LENGTH          EQU  *-MQRFH
                      ORG  MQRFH
MQRFH_AREA            DS   CL(MQRFH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQRFH
  StrucId        As String*4 'Structure identifier'
  Version        As Long     'Structure version number'
  StrucLength    As Long     'Total length of MQRFH including'
                             'NameValueString'
  Encoding       As Long     'Numeric encoding of data that follows'
                             'NameValueString'
  CodedCharSetId As Long     'Character set identifier of data that'
                             'follows NameValueString'
  Format         As String*8 'Format name of data that follows'
                             'NameValueString'
  Flags          As Long     'Flags'
End Type
```

MQRFH2 -Pravidla a formátovací záhlaví 2

Tato sekce popisuje pravidla a formátování záhlaví 2, jaká pole obsahuje, a počáteční hodnoty těchto polí.

Přehled pro MQRFH2

Dostupnost

Všechny systémy WebSphere MQ spolu s klienty WebSphere MQ MQI připojenými k těmto systémům.

Účel

Hlavička MQRFH2 je založena na záhlaví MQRFH , ale umožňuje přenos řetězců Unicode bez překladu a může přenášet číselné datové typy.

Struktura MQRFH2 definuje formát pravidel a záhlaví formátování version-2 . Toto záhlaví použijte k odeslání dat, která byla zakódována pomocí syntaxe podobné XML. Zpráva může obsahovat dvě nebo

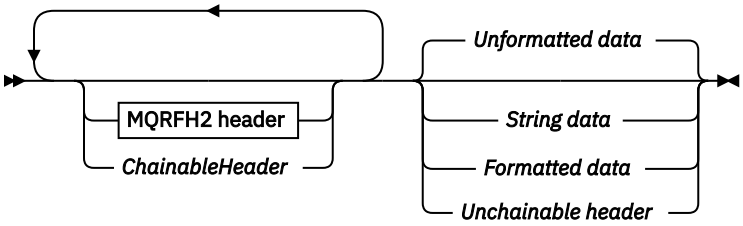
více struktur MQRFH2 v řadě, s uživatelskými daty volitelně následujících po poslední struktuře MQRFH2 v řadě.

Název formátu

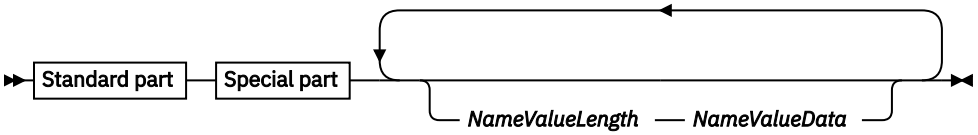
MQFMT_RF_HEADER_2

Syntax

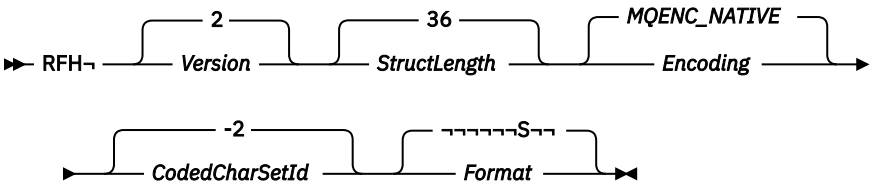
WebSphere MQ Message



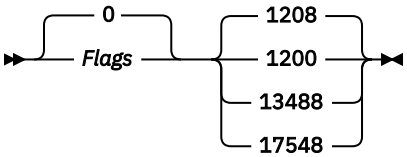
MQRFH2 header



Standard part



Special part



Znaková sada a kódování

Speciální pravidla platí pro znakovou sadu a kódování použité pro strukturu MQRFH2 :

- Pole jiná než *NameValueData* jsou ve znakové sadě a kódování dána poli *CodedCharSetId* a *Encoding* ve struktuře záhlaví, která předchází MQRFH2, nebo podle těchto polí ve struktuře MQMD , pokud je 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 MQGMO_CONVERT zadán ve volání MQGET , převede správce front na požadovanou znakovou sadu a kódování pole MQRFH2 (jiná než *NameValueData*).
- Parametr *NameValueData* se nachází ve znakové sadě zadané v poli *NameValueCCSID* . Pro *NameValueCCSID* jsou platné pouze uvedené znakové sady Unicode; Podrobnosti naleznete v popisu *NameValueCCSID* .

Některé znakové sady mají reprezentaci, která závisí na daném kódování. Je-li *NameValueCCSID* jednou z těchto znakových sad, musí být *NameValueData* ve stejném kódování jako ostatní pole v MQRFH2.

Je-li ve volání MQGET zadán parametr MQGMO_CONVERT , správce front převede *NameValueData* na požadované kódování, ale nezmění jeho znakovou sadu.

Pole pro MQRFH2

Struktura MQRFH2 obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

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.

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.

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	open-ended UCS-2
13488	Podmnožina UCS-2 2.0
17584	UCS-2 2.1 dílčí sada (včetně symbolu Euro)
1208	UTF-8

Pro znakové sady UCS-2 musí být kódování (pořadí bajtů) produktu *NameValueData* stejné jako kódování ostatních polí ve struktuře MQRFH2 . Náhradní znaky (X'D800'až X'DFFF') 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.

Data NameValueData (MQCHARn)
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* , které předchází poli *NameValueData* . Délka musí být násobkem čtyř.

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:

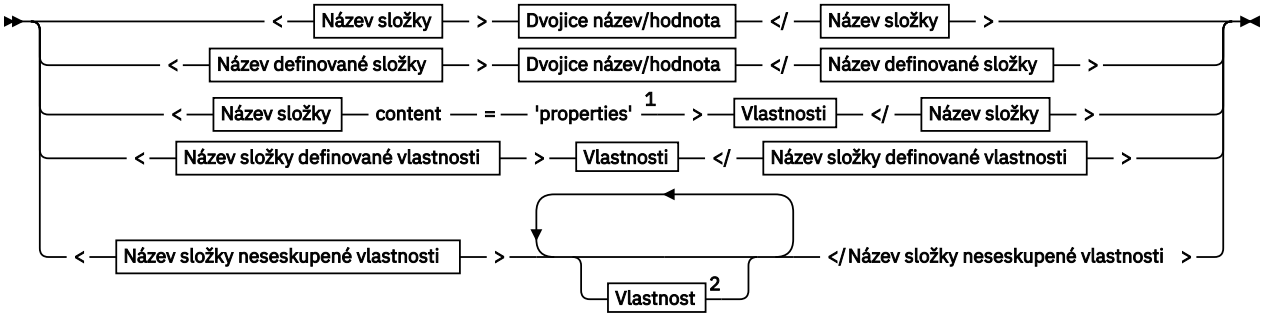
```
length1 data1 length2 data2 length3 data3
```

NameValueData se *nepřevádí* na znakovou sadu zadanou ve volání MQGET . I v případě, že je zpráva načtena s volbou MQGMO_CONVERT , zůstane *NameValueData* ve své původní znakové sadě. Avšak *NameValueData je* převedeno na kódování zadané ve volání MQGET .

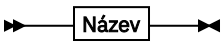
Poznámka: Protože tato pole jsou volitelná, jsou vynechána z deklarací struktury, které jsou poskytovány pro různé podporované programovací jazyky.

Poznámka: Termíny "definované" a "vyhrazené" se používají v diagramu syntaxe. Hodnota "Definováno" znamená, že název je používán produktem IBM WebSphere MQ. "Vyhrazeno" znamená, že název je vyhrazen pro budoucí použití produktem WebSphere MQ.

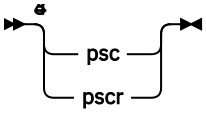
NameValueData syntaxe



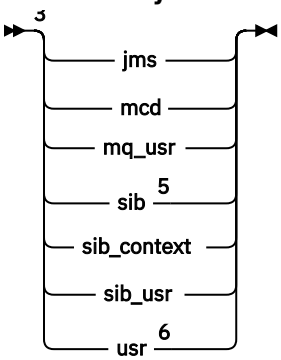
Název složky



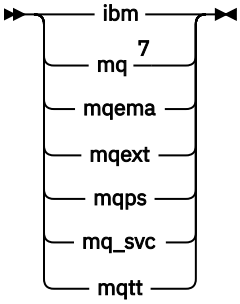
Název definované složky



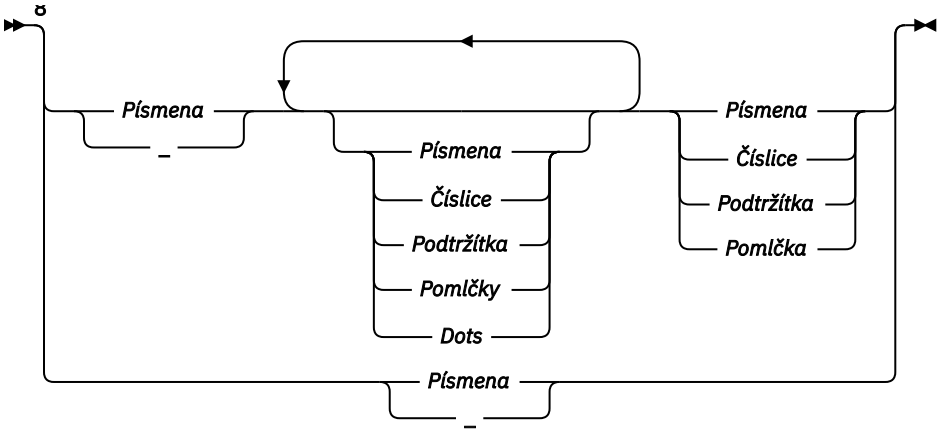
Název složky definované vlastnosti



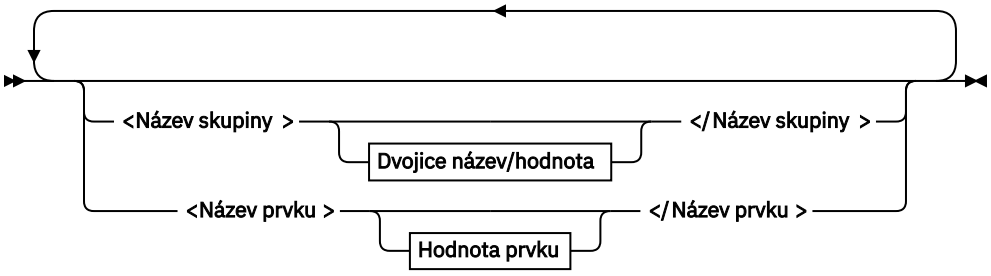
Název složky neseskupené vlastnosti



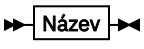
Název



Dvojice název/hodnota

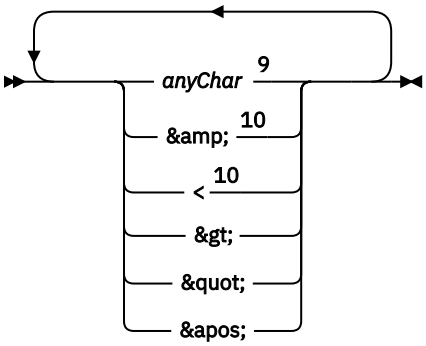


Název skupiny

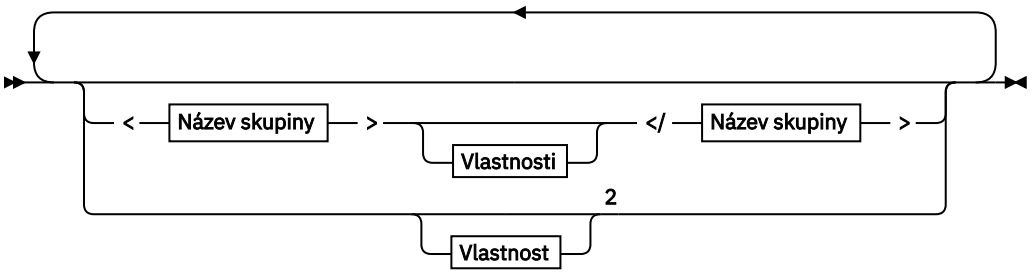


Název prvku

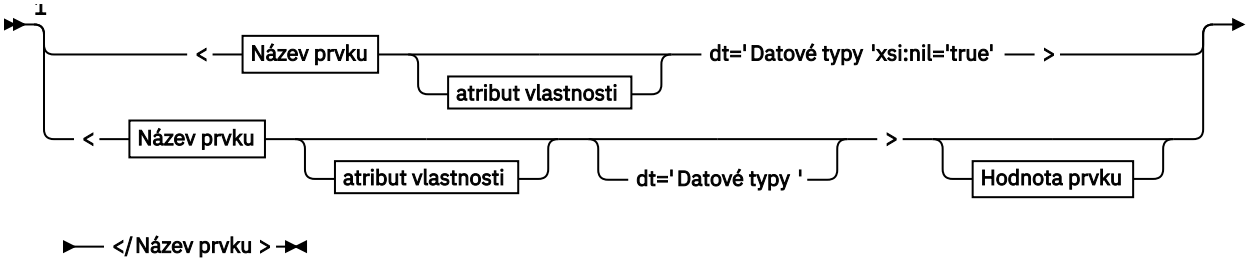
Hodnota prvku



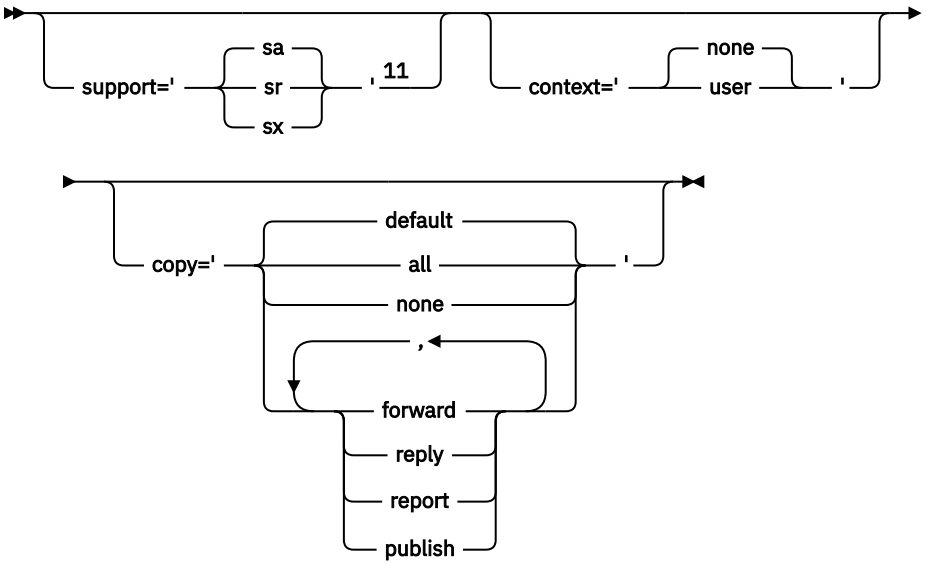
Vlastnosti



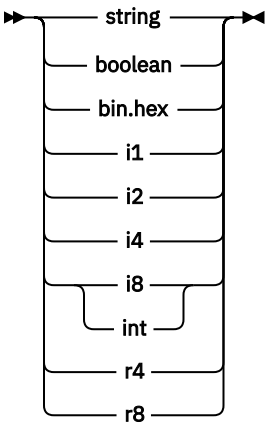
Vlastnost



atribut vlastnosti



Datové typy



Poznámky:

[1] Dvojité uvozovky nebo jednoduché uvozovky jsou platné.

[2] Nepoužijte neplatný název vlastnosti, viz “Neplatný název vlastnosti” na stránce 504. Použijte vyhrazený název vlastnosti pouze pro její definovaný účel; viz “Definované názvy vlastností” na stránce 503.

[3] Název musí být uveden malými písmeny.

[4] Podporována je pouze jedna složka `psc` a `pscr` .

[5] Významné jsou pouze vlastnosti v prvním záhlaví `MQRFH2` . Produkt WebSphere Application Server Service Integration Bus ignoruje složky sib, sib_contexta sib_usr v následujících záhlavích správce `MQRFH2` .

[6] Více než jedna složka `usr` musí být přítomna v `MQRFH2`. 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 jsou v podstatě tvořeny kategoriemi Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` a `Nd`.

[9] Všechny znaky jsou významné. Počáteční a koncové mezery jsou součástí hodnoty prvku.

[10] Nepoužijte neplatný znak; viz “Neplatné znaky” na stránce 503. Použijte řídicí posloupnost, spíše než tyto neplatné znaky.

[11] Atribut vlastnosti podpory je platný pouze ve složce `mq` .

Název složky

`NameValueData` obsahuje jednu složku. Chcete-li vytvořit více složek, vytvořte více polí *`NameValueData`* . V jednom záhlaví `MQRFH2` v rámci zprávy můžete vytvořit více polí *`NameValueData`* . 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. `MQSETMP`a 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 smíšeny na stejné úrovni; viz Obrázek 1 na stránce 494. Nekombinujte název skupiny a název prvku, viz Obrázek 2 na stránce 494

```
<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 503 a “Vyhrazená složka nebo název složky vlastností” na stránce 503. Použít definovaný název složky pouze pro její definovaný účel; viz “Název definované složky” na stránce 495.

Přidáte-li atribut `'content=properties'` do značky názvu složky, stane se tato složka složkou vlastností. Další informace naleznete v tématu Obrázek 3 na stránce 494.

```
<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. `Folder1` v Obrázek 4 na stránce 494 musí být jiný název než `Folder2` v Obrázek 5 na stránce 494.

```
<Folder1><NVP1>value</NVP1></Folder1>
```

Obrázek 4. `Folder1` prostor jmen

```
<Folder2 content='properties'><Property1>value</Property1></Folder2>
```

Obrázek 5. `Folder2` prostor jmen

Skupiny, vlastnosti a dvojice název/hodnota v různých složkách mají různé obory názvů. `Property1` v produktu Obrázek 5 na stránce 494 je jiná vlastnost produktu `Property1` v produktu Obrázek 6 na stránce 494.

```
<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. K přístupu k vlastnostem zpráv použijte definovaná rozhraní, jako jsou vlastnosti zprávy MQI nebo zprávy JMS. 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 WebSphere MQnebo jiným produktem. Nevytvářejte složku se stejným názvem a do složek nepřidávejte své vlastní dvojice název/hodnota. 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 WebSphere MQnebo jiný produkt. Názvy složek a jejich obsah naleznete v tématu Složky vlastností. Definované názvy složek vlastností jsou podmnožinou všech názvů složek vyhrazených produktem WebSphere MQ; viz “Vyhrazená složka nebo název složky vlastností” na stránce 503.

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, například mq a sib, WebSphere MQ ignoruje nebo odkáže vlastnosti, které nerozezná.

Popis každé definované složky vlastností obsahuje seznam vlastností, které IBM WebSphere 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 WebSphere 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. Nevytvářejte však vlastnosti s neplatnými názvy vlastností; viz “Neplatný název vlastnosti” na stránce 504.

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 rozhraní JMS MQRFH2.

Tabulka 532. jms - název vlastnosti, 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 532. jms - název vlastnosti, 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 nachází ve zprávě obsahující produkt MQRFH2 odeslaný z produktu WebSphere Message Broker. Popisuje doménu, formát, typ a sadu zpráv příslušné zprávy.

Tabulka 533. 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 JMS. Vlastnosti, které nesplňují požadavky 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 (WAS/SIB) produktu WebSphere Application Server. Vlastnosti produktu `sib` nejsou vystaveny jako vlastnosti JMS aplikacím produktu IBM WebSphere MQ JMS, protože tyto aplikace nejsou podporovány. Některé vlastnosti produktu `sib` například nemohou být vystaveny jako vlastnosti 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 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 JMS, protože nejsou podporované typy. Produkt `sib_usr` je vystaven aplikacím WAS/SIB v rozhraní produktu `SIMessage` , viz téma Vývoj integrace služeb.

Typ vlastnosti `sib_usr` musí být `bin.hexa` hodnota musí být ve správném formátu. Pokud aplikace IBM WebSphere 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 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 534. `usr` - název vlastnosti, synonymum, datový typ a složka.

Hodnoty vlastností webových služeb jsou popsány v části MQRFH2 Nastavení protokolu SOAP .

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 WebSphere MQ.

Tabulka 535. 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 WebSphere 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 serverem WebSphere Application Server. Složka byla nahrazena mqext.

Nepřidávejte své vlastní vlastnosti do složky mqema.

mqext

mqext obsahuje vlastnosti, které jsou používány pouze serverem WebSphere Application Server. Složka je k dispozici pouze v případě, že aplikace byla nastavena alespoň jednou z definovaných vlastností IBM .

Tabulka 536. mqext - název vlastnosti, synonymum, datový typ a složka

Synonymum vlastnosti	Název vlastnosti	Datový typ	Složka
JMSArmCorrelator	mqext.Arm	string	<mqext><Arm>*armCorrelator*</Arm></mqext>
JMSRMCorrelator	mqext.Wrm	string	<mqext><Wrm>*wrmCorrelator*</Wrm></mqext>

Nepřidávejte své vlastní vlastnosti do složky mqext.

mqps

Produkt mqps obsahuje vlastnosti, které jsou používány pouze v rámci publikování/odběru produktu IBM WebSphere MQ. Tato složka je přítomna pouze, když má aplikace nastavenu minimálně jednu z integrovaných vlastností publikování/odběru.

Tabulka 537. 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 IBM WebSphere MQ Telemetry

Tabulka 538. 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 494.

Vlastnosti

V syntaktických diagramech popisuje "Vlastnosti" obsah složky vlastností. Složka vlastností obsahuje skupiny a vlastnosti. Vlastnost je dvojice název/hodnota s volitelným atributem datového typu. Skupina obsahuje vlastnosti a další skupiny.

Pokud jde o stromy, vlastnosti jsou 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 494.

Vlastnost

Vlastnost zprávy je dvojice název/hodnota ve složce vlastností. Volitelně může obsahovat atribut datového typu a atribut vlastnosti. Příklad viz Obrázek 7 na stránce 500. Je-li atribut datového typu vynechán, typ vlastnosti je string.

```
<pf><p1 dt='i8' >value</p1></pf>
```

Obrázek 7. Atribut datového typu

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* v Obrázek 8 na stránce 500. Řetězec je formátován pro snadnější čtení.

```
<myPropertyFolder1>
    <myGroup1>
        <myGroup2>
            <myProperty1>value</myProperty1>
        </myGroup2>
    </myGroup1>
</myPropertyFolder1>
```

Obrázek 8. Mapování názvu jedné vlastnosti

Složka vlastností může obsahovat více vlastností. Například vlastnosti v produktu Obrázek 9 na stránce 500 jsou mapovány na složku vlastností v produktu Obrázek 10 na stránce 501 .

```
myPropertyFolder1.myProperty4
myPropertyFolder1.myGroup1.myGroup2.myProperty1
myPropertyFolder1.myGroup1.myGroup2.myProperty2
myPropertyFolder1.myGroup1.myProperty3
```

Obrázek 9. 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 10. Mapování více názvů vlastností

Název

Název musí začínat písmenem *Letter* nebo *Underscore*. Nesmí obsahovat *dvojtečku*, ne konec za *Období* a obsahovat pouze *Písmena*, *Číslice*, *Podtržítka*, *Pomlčka*a *Dots*. Platné znaky jsou definovány ve specifikaci XML W3C a jsou v podstatě tvořeny kategoriemi Unicode `Ll, Lu, Lo, Lt, Nl, Mc, Mn, Lm,` a `Nd`.

Úplná cesta k vlastnosti nebo dvojici název/hodnota nesmí porušit pravidlo popsané v "Neplatný název cesty" na stránce 503. 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. Dvojice vlastností a názvu/hodnoty 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 všechny bílé znaky mezi značkou <*Element name*> a </*Element name*>. Nepoužívejte dva znaky < a & v hodnotě. Poté nahraďte produkty < a `&`.

atribut vlastnosti

Pole deskriptoru vlastností mapují pole deskriptoru vlastností: Mapování jsou následující:

Podpora

sa
MQPD_SUPPORT_OPTIONAL

sr
MQPD_SUPPORT_REQUIRED

sx
MQPD_SUPPORT_REQUIRED_IF_LOCAL

Kontext

none
MQPD_NO_CONTEXT

user
MQPD_USER _CONTEXT

CopyOptions

forward
MQPD_COPY_FORWARD

reply
MQPD_COPY_REPLY

report
MQPD_COPY_REPORT

publish
MQPD_COPY_PUBLISH

all
MQPD_COPY_ALL

Nepoužívejte `all` v kombinaci s dalšími volbami.

default
MQPD_COPY_DEFAULT

Nepoužívejte `default` v kombinaci s dalšími volbami. `default` je stejný jako `forward` + `report` + `publish`

none
MQPD_COPY_NONE

Nepoužívejte none v kombinaci s dalšími volbami.

Atributy vlastností Podpora se vztahují pouze na vlastnosti ve složce `mq` .

Atributy vlastností Kontext a CopyOptions jsou použitelné pro všechny složky vlastností.

Datový typ

Datové typy produktu MQRFH2 jsou mapovány na typy vlastností zpráv takto:

Tabulka 539. 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:

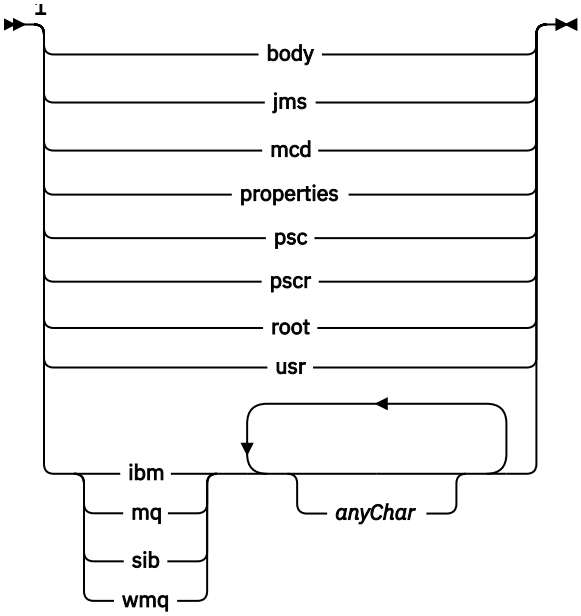
```
<NullProperty
xsi:nil='true'></NullProperty>
```

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 rezervovány pro názvy složek nebo vlastností vytvořených společností IBM.

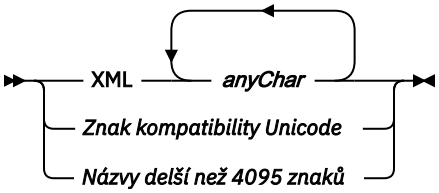


Poznámky:

[1] Vyhrazená složka nebo název vlastnosti obsahuje libovolnou kombinaci malých a velkých písmen.

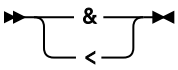
Neplatný název cesty

Omezte úplnou cestu páru název/hodnota nebo vlastnost nezahrnujte některý z následujících řetězců.



Neplatné znaky

Vždy používejte esc sekvence & a < místo literálů "&" a "<".



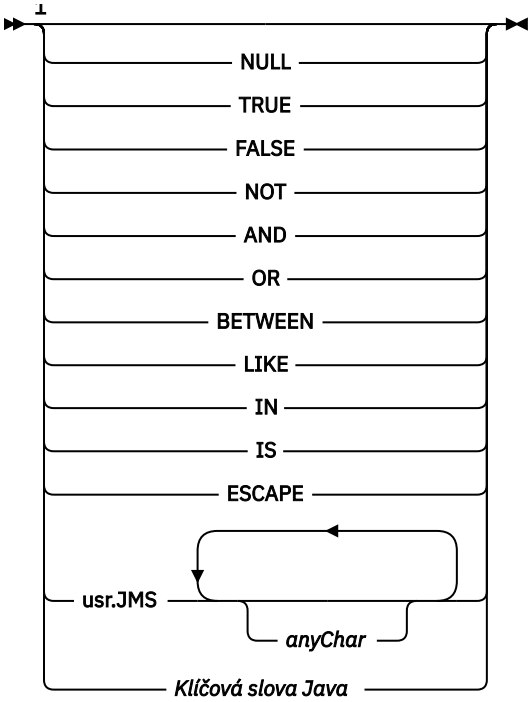
Definované názvy vlastností

Definované názvy vlastností jsou názvy vlastností, které jsou definovány produktem WebSphere MQnebo jinými produkty a používají je produkt IBM WebSphere 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.

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.

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.

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 neobsahuje 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQRFH2

Tabulka 540. Počáteční hodnoty polí v MQRFH2 pro MQRFH2

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQRFH_STRUCT	'RFH¬'
Version	MQRFH_VERSION_2	2
StrucLength	MQRFH_STRUC_LENGTH_FIXED_2	36
Encoding	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	MQCSI_INHERIT	-2
Format	MQFMT_NONE	Mezery
Flags	MQRFH_NONE	0
NameValueCCSID	Není	1208

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQRFH2_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

   ```
   MQRFH2 MyRFH2 = {MQRFH2_DEFAULT};
   ```

Deklarace C

```
typedef struct tagMQRFH2 MQRFH2;
struct tagMQRFH2 {
  MQCHAR4  StrucId;         /* Structure identifier */
```

```
  MQLONG   Version;         /* Structure version number */
  MQLONG   StrucLength;     /* Total length of MQRFH2 including all
                               NameValueLength and NameValueData
                               fields */
  MQLONG   Encoding;        /* Numeric encoding of data that follows
                               last NameValueData field */
  MQLONG   CodedCharSetId;  /* Character set identifier of data that
                               follows last NameValueData field */
  MQCHAR8  Format;          /* Format name of data that follows last
                               NameValueData field */
  MQLONG   Flags;           /* Flags */
  MQLONG   NameValueCCSID;  /* Character set identifier of
                               NameValueData */
};
```

Deklarace COBOL

```
**   MQRFH2 structure
  10 MQRFH2.
**    Structure identifier
   15 MQRFH2-STRUCID        PIC X(4).
**    Structure version number
   15 MQRFH2-VERSION        PIC S9(9) BINARY.
**    Total length of MQRFH2 including all NAMEVALUELENGTH and
**    NAMEVALUEDATA fields
   15 MQRFH2-STRUCLENGTH    PIC S9(9) BINARY.
**    Numeric encoding of data that follows last NAMEVALUEDATA field
   15 MQRFH2-ENCODING       PIC S9(9) BINARY.
**    Character set identifier of data that follows last NAMEVALUEDATA
**    field
   15 MQRFH2-CODEDCHARSETID PIC S9(9) BINARY.
**    Format name of data that follows last NAMEVALUEDATA field
   15 MQRFH2-FORMAT         PIC X(8).
**    Flags
   15 MQRFH2-FLAGS          PIC S9(9) BINARY.
**    Character set identifier of NAMEVALUEDATA
   15 MQRFH2-NAMEVALUECCSID PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQRFH2 based,
  3 StrucId        char(4),       /* Structure identifier */
  3 Version        fixed bin(31), /* Structure version number */
  3 StrucLength    fixed bin(31), /* Total length of MQRFH2 including
                                     all NameValueLength and
                                     NameValueData fields */
  3 Encoding       fixed bin(31), /* Numeric encoding of data that
                                     follows last NameValueData field */
  3 CodedCharSetId fixed bin(31), /* Character set identifier of data
                                     that follows last NameValueData
                                     field */
  3 Format         char(8),       /* Format name of data that follows
                                     last NameValueData field */
  3 Flags          fixed bin(31), /* Flags */
  3 NameValueCCSID fixed bin(31); /* Character set identifier of
                                     NameValueData */
```

Deklarace High Level Assembler

```
MQRFH                DSECT
MQRFH_STRUCID        DS   CL4  Structure identifier
MQRFH_VERSION        DS   F    Structure version number
MQRFH_STRUCLENGTH    DS   F    Total length of MQRFH2 including all
*                              NAMEVALUELENGTH and NAMEVALUEDATA fields
MQRFH_ENCODING       DS   F    Numeric encoding of data that follows
*                              last NAMEVALUEDATA field
MQRFH_CODEDCHARSETID DS   F    Character set identifier of data that
*                              follows last NAMEVALUEDATA field
MQRFH_FORMAT         DS   CL8  Format name of data that follows last
*                              NAMEVALUEDATA field
MQRFH_FLAGS          DS   F    Flags
MQRFH_NAMEVALUECCSID DS   F    Character set identifier of
```

```
*                               NAMEVALUEDATA
*
MQRFH_LENGTH          EQU  *-MQRFH
                      ORG  MQRFH
MQRFH_AREA            DS   CL(MQRFH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQRFH2
  StrucId        As String*4 'Structure identifier'
  Version        As Long     'Structure version number'
  StrucLength    As Long     'Total length of MQRFH2 including all'
                             'NameValueLength and NameValueData fields'
  Encoding       As Long     'Numeric encoding of data that follows'
                             'last NameValueData field'
  CodedCharSetId As Long     'Character set identifier of data that'
                             'follows last NameValueData field'
  Format         As String*8 'Format name of data that follows last'
                             'NameValueData field'
  Flags          As Long     'Flags'
  NameValueCCSID As Long     'Character set identifier of NameValueData'
End Type
```

MQRMH-záhlaví zprávy odkazu

Následující tabulka shrnuje pole ve struktuře.

Tabulka 541. Pole v MQRMH

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadná data	StrucLength
Encoding	Numerické kódování hromadných dat	Kódování
CodedCharSetId	Identifikátor znakové sady hromadných dat	CodedCharSetId
Format	Název formátu hromadných dat	Formát
Flags	Příznaky referenční zprávy	Příznaky
ObjectType	Typ objektu	ObjectType
ObjectInstanceId	Identifikátor instance objektu	ObjectInstance
SrcEnvLength	Délka dat zdrojového prostředí	SrcEnvDélka
SrcEnvOffset	Posunutí dat o zdrojovém prostředí	SrcEnvPosunutí
SrcNameLength	Délka názvu zdrojového objektu	DélkaSrcName
SrcNameOffset	Odsazení jména zdrojového objektu	SrcNamePosunutí
DestEnvLength	Délka dat cílového prostředí	DestEnvDélka
DestEnvOffset	Odsazení dat cílového prostředí	DestEnvPosunutí
DestNameLength	Délka názvu cílového objektu	DestNameDélka
DestNameOffset	Posunutí názvu cílového objektu	DestNameOffset
DataLogicalLength	Délka hromadných dat	DataLogicalDélka
DataLogicalOffset	Nízký posun hromadných údajů	DataLogicalposunutí

Tabulka 541. Pole v MQRMH (pokračování)		
Pole	**Popis**	**Téma**
DataLogicalOffset2	Vysoký posun hromadných dat	DataLogicalOffset2

Přehled pro MQRMH

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Okna, plus klienti WebSphere MQ připojené k těmto systémům.

Účel: Struktura MQRMH definuje formát záhlaví referenční zprávy. Toto záhlaví se používá s uživatelskými ukončovacími programy zpráv napsaných uživatelem k odeslání extrémně velkého množství dat (s názvem *bulk 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. To snižuje možnost vyčerpání prostředků produktu MQ o malý počet extrémně velkých zpráv.

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; tento údaj je dán atributem správce front *CodedCharSetId* . Numerická data v MQRMH musí být v nativním kódování počítače. Tato hodnota je dána hodnotou MQENC_NATIVE pro programovací jazyk C.

Nastavte znakovou sadu a kódování MQRMH do polí *CodedCharSetId* a *Encoding* 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. Když agent kanálu zpráv (MCA) čte zprávu z přenosové fronty, vyvolá se uživatelská procedura pro zpracování zpráv, která zpracuje záhlaví referenční zprávy. 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 musí existovat uživatelská procedura pro zprávy, která čeká na referenční zprávy, musí existovat. Při přijetí referenční zprávy musí uživatelská procedura vytvořit objekt z hromadného dat, která následuje za MQRMH ve zprávě, a poté 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, před strukturou MQRMH se nachází jedno nebo více dalších záhlaví.

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: Neposílejte referenční zprávu jako segmentovanou zprávu, protože uživatelská procedura pro zprávy ji správně nemůže zpracovat.

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 *StrucLength* bajtů na začátku struktury jsou buď vyřazeny, nebo mají nedefinované hodnoty po převodu dat. Hromadná data se převedou za předpokladu, že všechny následující podmínky jsou pravdivé:

- Hromadná data se nacházejí ve zprávě, když se provádí konverze dat.
- Pole *Format* v MQRMH má jinou hodnotu než MQFMT_NONE.
- Uživatelem zapsaná uživatelská procedura pro převod dat existuje s uvedeným názvem formátu.

Uvědomte si však, že obvykle hromadná data *nejsou* přítomná ve zprávě, když je zpráva ve frontě, a že hromadné údaje jsou převáděny volbou MQGMO_CONVERT.

Pole pro MQRMH

Struktura MQRMH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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 *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.

Nepoužívejte MQCCSI_INHERIT, je-li hodnota pole *PutApplType* v deskriptoru MQMD MQAT_BROKER.

Tato hodnota je podporována v následujících prostředích: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ , kteří jsou připojeni k těmto systémům.

Počáteční hodnota tohoto pole je MQCCSI_UNDEFINED.

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*. Tato hodnota *není* fyzickým posunem 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.

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 v systému Windows mohou být data cílového prostředí cestou k adresáři objektu, kde mají být hromadná data uložena. 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*; pokud je 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 produktem *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.

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.

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.

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.

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.

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* popsané výše.

Počáteční hodnota tohoto pole je 8 mezer.

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, v systému 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*; je-li tato délka nula, nejsou žádná data o zdrojovém prostředí a *SrcEnvOffset* se ignoruje. 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 libovolními 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*; pokud je tato délka nula, neexistuje žádné jméno zdrojového objektu a *SrcNameOffset* se ignoruje. 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.

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.

StrucLength (MQLONG)
Celková délka MQRMH, včetně řetězců na konci pevných polí, ale ne hromadná data.

Počáteční hodnota tohoto pole je nula.

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.

Počáteční hodnoty a deklarace jazyka pro MQRMH

Tabulka 542. Počáteční hodnoty polí v MQRMH pro MQRMH

Název pole	**Název konstanty**	**Hodnota konstanty**
StrucId	MQRMH_STRUCTION_ID	'RMH¬'
Version	MQRMH_VERSION_1	1
StrucLength	Není	0
Encoding	MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQRMHF_NOT_LAST	0
ObjectType	Není	Mezery
ObjectInstanceId	MQOII_NONE	Hodnoty null
SrcEnvLength	Není	0
SrcEnvOffset	Není	0
SrcNameLength	Není	0

Tabulka 542. Počáteční hodnoty polí v MQRMH pro MQRMH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
SrcNameOffset	Není	0
DestEnvLength	Není	0
DestEnvOffset	Není	0
DestNameLength	Není	0
DestNameOffset	Není	0
DataLogicalLength	Není	0
DataLogicalOffset	Není	0
DataLogicalOffset2	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQRMH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRMH MyRMH = {MQRMH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQRMH MQRMH;
struct tagMQRMH {
  MQCHAR4   StrucId;             /* Structure identifier */
  MQLONG    Version;             /* Structure version number */
  MQLONG    StrucLength;         /* Total length of MQRMH, including
                                    strings at end of fixed fields, but
                                    not the bulk data */
  MQLONG    Encoding;            /* Numeric encoding of bulk data */
  MQLONG    CodedCharSetId;      /* Character set identifier of bulk
                                    data */
  MQCHAR8   Format;              /* Format name of bulk data */
  MQLONG    Flags;               /* Reference message flags */
  MQCHAR8   ObjectType;          /* Object type */
  MQBYTE24  ObjectInstanceId;    /* Object instance identifier */
  MQLONG    SrcEnvLength;        /* Length of source environment data */
  MQLONG    SrcEnvOffset;        /* Offset of source environment data */
  MQLONG    SrcNameLength;       /* Length of source object name */
  MQLONG    SrcNameOffset;       /* Offset of source object name */
  MQLONG    DestEnvLength;       /* Length of destination environment
                                    data */
  MQLONG    DestEnvOffset;       /* Offset of destination environment
                                    data */
  MQLONG    DestNameLength;      /* Length of destination object name */
  MQLONG    DestNameOffset;      /* Offset of destination object name */
  MQLONG    DataLogicalLength;   /* Length of bulk data */
  MQLONG    DataLogicalOffset;   /* Low offset of bulk data */
  MQLONG    DataLogicalOffset2;  /* High offset of bulk data */
};
```

Deklarace COBOL

```
**   MQRMH structure
  10 MQRMH.
**    Structure identifier
   15 MQRMH-STRUCID            PIC X(4).
**    Structure version number
   15 MQRMH-VERSION            PIC S9(9) BINARY.
**    Total length of MQRMH, including strings at end of fixed fields,
**    but not the bulk data
```

```
   15 MQRMH-STRUCLENGTH        PIC S9(9) BINARY.
**    Numeric encoding of bulk data
   15 MQRMH-ENCODING           PIC S9(9) BINARY.
**    Character set identifier of bulk data
   15 MQRMH-CODEDCHARSETID     PIC S9(9) BINARY.
**    Format name of bulk data
   15 MQRMH-FORMAT             PIC X(8).
**    Reference message flags
   15 MQRMH-FLAGS              PIC S9(9) BINARY.
**    Object type
   15 MQRMH-OBJECTTYPE         PIC X(8).
**    Object instance identifier
   15 MQRMH-OBJECTINSTANCEID   PIC X(24).
**    Length of source environment data
   15 MQRMH-SRCENVLENGTH       PIC S9(9) BINARY.
**    Offset of source environment data
   15 MQRMH-SRCENVOFFSET       PIC S9(9) BINARY.
**    Length of source object name
   15 MQRMH-SRCNAMELENGTH      PIC S9(9) BINARY.
**    Offset of source object name
   15 MQRMH-SRCNAMEOFFSET      PIC S9(9) BINARY.
**    Length of destination environment data
   15 MQRMH-DESTENVLENGTH      PIC S9(9) BINARY.
**    Offset of destination environment data
   15 MQRMH-DESTENVOFFSET      PIC S9(9) BINARY.
**    Length of destination object name
   15 MQRMH-DESTNAMELENGTH     PIC S9(9) BINARY.
**    Offset of destination object name
   15 MQRMH-DESTNAMEOFFSET     PIC S9(9) BINARY.
**    Length of bulk data
   15 MQRMH-DATALOGICALLENGTH  PIC S9(9) BINARY.
**    Low offset of bulk data
   15 MQRMH-DATALOGICALOFFSET  PIC S9(9) BINARY.
**    High offset of bulk data
   15 MQRMH-DATALOGICALOFFSET2 PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQRMH based,
  3 StrucId            char(4),       /* Structure identifier */
  3 Version            fixed bin(31), /* Structure version number */
  3 StrucLength        fixed bin(31), /* Total length of MQRMH,
                                         including strings at end of
                                         fixed fields, but not the bulk
                                         data */
  3 Encoding           fixed bin(31), /* Numeric encoding of bulk
                                         data */
  3 CodedCharSetId     fixed bin(31), /* Character set identifier of
                                         bulk data */
  3 Format             char(8),       /* Format name of bulk data */
  3 Flags              fixed bin(31), /* Reference message flags */
  3 ObjectType         char(8),       /* Object type */
  3 ObjectInstanceId   char(24),      /* Object instance identifier */
  3 SrcEnvLength       fixed bin(31), /* Length of source environment
                                         data */
  3 SrcEnvOffset       fixed bin(31), /* Offset of source environment
                                         data */
  3 SrcNameLength      fixed bin(31), /* Length of source object name */
  3 SrcNameOffset      fixed bin(31), /* Offset of source object name */
  3 DestEnvLength      fixed bin(31), /* Length of destination
                                         environment data */
  3 DestEnvOffset      fixed bin(31), /* Offset of destination
                                         environment data */
  3 DestNameLength     fixed bin(31), /* Length of destination object
                                         name */
  3 DestNameOffset     fixed bin(31), /* Offset of destination object
                                         name */
  3 DataLogicalLength  fixed bin(31), /* Length of bulk data */
  3 DataLogicalOffset  fixed bin(31), /* Low offset of bulk data */
  3 DataLogicalOffset2 fixed bin(31); /* High offset of bulk data */
```

Deklarace High Level Assembler

```
MQRMH                   DSECT
MQRMH_STRUCID           DS   CL4   Structure identifier
```

```
MQRMH_VERSION             DS   F     Structure version number
MQRMH_STRUCLENGTH         DS   F     Total length of MQRMH, including
*                                    strings at end of fixed fields, but
*                                    not the bulk data
MQRMH_ENCODING            DS   F     Numeric encoding of bulk data
MQRMH_CODEDCHARSETID      DS   F     Character set identifier of bulk
*                                    data
MQRMH_FORMAT              DS   CL8   Format name of bulk data
MQRMH_FLAGS               DS   F     Reference message flags
MQRMH_OBJECTTYPE          DS   CL8   Object type
MQRMH_OBJECTINSTANCEID    DS   XL24  Object instance identifier
MQRMH_SRCENVLENGTH        DS   F     Length of source environment data
MQRMH_SRCENVOFFSET        DS   F     Offset of source environment data
MQRMH_SRCNAMELENGTH       DS   F     Length of source object name
MQRMH_SRCNAMEOFFSET       DS   F     Offset of source object name
MQRMH_DESTENVLENGTH       DS   F     Length of destination environment
*                                    data
MQRMH_DESTENVOFFSET       DS   F     Offset of destination environment
*                                    data
MQRMH_DESTNAMELENGTH      DS   F     Length of destination object name
MQRMH_DESTNAMEOFFSET      DS   F     Offset of destination object name
MQRMH_DATALOGICALLENGTH   DS   F     Length of bulk data
MQRMH_DATALOGICALOFFSET   DS   F     Low offset of bulk data
MQRMH_DATALOGICALOFFSET2  DS   F     High offset of bulk data
*
MQRMH_LENGTH              EQU  *-MQRMH
                          ORG  MQRMH
MQRMH_AREA                DS   CL(MQRMH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQRMH
  StrucId            As String*4 'Structure identifier'
  Version            As Long     'Structure version number'
  StrucLength        As Long     'Total length of MQRMH, including'
                                 'strings at end of fixed fields, but'
                                 'not the bulk data'
  Encoding           As Long     'Numeric encoding of bulk data'
  CodedCharSetId     As Long     'Character set identifier of bulk data'
  Format             As String*8 'Format name of bulk data'
  Flags              As Long     'Reference message flags'
  ObjectType         As String*8 'Object type'
  ObjectInstanceId   As MQBYTE24 'Object instance identifier'
  SrcEnvLength       As Long     'Length of source environment data'
  SrcEnvOffset       As Long     'Offset of source environment data'
  SrcNameLength      As Long     'Length of source object name'
  SrcNameOffset      As Long     'Offset of source object name'
  DestEnvLength      As Long     'Length of destination environment'
                                 'data'
  DestEnvOffset      As Long     'Offset of destination environment'
                                 'data'
  DestNameLength     As Long     'Length of destination object name'
  DestNameOffset     As Long     'Offset of destination object name'
  DataLogicalLength  As Long     'Length of bulk data'
  DataLogicalOffset  As Long     'Low offset of bulk data'
  DataLogicalOffset2 As Long     'High offset of bulk data'
End Type
```

MQRR-záznam odpovědi

Následující tabulka shrnuje pole ve struktuře.

Tabulka 543. Pole v objektu MQRR

Pole	Popis	Téma
CompCode	Kód dokončení pro frontu	CompCode
Reason	Kód příčiny pro frontu	Příčina

Přehled pro objekt MQRR

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Okna, plus klienti WebSphere MQ připojené k těmto systémům.

Účel: Použijte strukturu MQRR k přijetí kódu dokončení a kódu příčiny, který je výsledkem operace otevření nebo vložení pro jednu cílovou frontu, je-li cílem distribuční seznam. MQRR je výstupní struktura pro volání MQOPEN, MQPUT a MQPUT1 .

Znaková sada a kódování: Data v objektu MQRR musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Použití: Poskytnutím pole těchto struktur na voláních MQOPEN a MQPUT nebo na volání MQPUT1 můžete určit kódy dokončení a kódy příčin 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 MQRC_MULTIPLE_REASONS from the call indicates that the response records (if provided by the application) has been set by the queue manager.

Pole pro objekt MQRR

Struktura MQRR obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

Počáteční hodnoty a deklarace jazyka pro MQRR

Tabulka 544. Počáteční hodnoty polí v objektu MQRR pro objekt MQRR

Název pole	Název konstanty	Hodnota konstanty
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0

Notes:

1. V programovacím jazyce C-proměnná makraHodnota MQRR_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQRR MyRR = {MQRR_DEFAULT};
```

Deklarace C

```
typedef struct tagMQRR MQRR;
struct tagMQRR {
  MQLONG  CompCode;  /* Completion code for queue */
  MQLONG  Reason;    /* Reason code for queue */
};
```

Deklarace COBOL

```
**   MQRR structure
  10 MQRR.
**    Completion code for queue
   15 MQRR-COMPCODE PIC S9(9) BINARY.
```

```
**    Reason code for queue
   15 MQRR-REASON   PIC S9(9) BINARY.
```

Deklarace PL/I

```
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

```
Type MQRR
  CompCode As Long 'Completion code for queue'
  Reason   As Long 'Reason code for queue'
End Type
```

MQSCO-Volby konfigurace SSL

Následující tabulka shrnuje pole ve struktuře.

Tabulka 545. Pole v MQSCO.

Seznam polí v MQSCO, podle verze, s odkazy na témata popisující pole.

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
KeyRepository	Umístění úložiště klíčů	KeyRepository
CryptoHardware	Podrobnosti kryptografického hardwaru	CryptoHardware
AuthInfoRecCount	Počet existujících záznamů MQAIR	AuthInfoRecCount
AuthInfoRecOffset	Posunutí prvního záznamu MQAIR od začátku MQSCO	AuthInfoRecOffset
AuthInfoRecPtr	Adresa prvního záznamu MQAIR	AuthInfoRecPtr
Poznámka: Následující dvě pole se ignorují, pokud *Version* je menší než MQSCO_VERSION_2.		
KeyResetCount	Počet obnovení tajného klíče zabezpečení SSL	PočetKeyReset
FipsRequired	Použití šifrovacích algoritmů certifikovaných pomocí standardu FIPS v produktu WebSphere MQ	“FipsRequired (MQLONG)” na stránce 520
Poznámka: Následující pole je ignorováno, pokud *Version* je menší než MQSCO_VERSION_3.		
EncryptionPolicySuiteB	Použít pouze šifrovací algoritmy sady Suite B	EncryptionPolicySuiteB
Poznámka: Následující pole je ignorováno, pokud *Version* je menší než MQSCO_VERSION_4.		
CertificateValPolicy	Zásada ověření certifikátu	CertificateVal

Související odkazy

“MQCNO-Volby připojení” na stránce 292
Následující tabulka shrnuje pole ve struktuře.

“Přehled pro MQSCO” na stránce 518

Dostupnost: Klienti AIX, HP-UX, IBM i, Solaris, Linux a Windows .

“Pole pro MQSCO” na stránce 518

“Počáteční hodnoty a jazyková prohlášení pro MQSCO” na stránce 522

Přehled pro MQSCO

Dostupnost: Klienti AIX, HP-UX, IBM i, Solaris, Linux a Windows .

Účel: Struktura MQSCO (ve spojení s poli SSL ve struktuře MQCD) umožňuje aplikaci spuštěnou jako klient WebSphere MQ MQI k určení voleb konfigurace, které řídí použití protokolu SSL pro připojení klienta, je-li protokol kanálu TCP/IP. Struktura je vstupním parametrem volání MQCONNX.

Pokud není protokol kanálu pro kanál klienta TCP/IP, bude struktura MQSCO ignorována.

Znaková sada a kódování: Data ve struktuře znaků MQSCO se nacházejí ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front daného parametrem MQENC_NATIVE.

Pole pro MQSCO

Struktura MQSCO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

AuthInfoRecCount (MQLONG)

Jedná se o počet záznamů ověřovacích informací (MQAIR) adresovaných poli *AuthInfoRecPtr* nebo *AuthInfoRecOffset* . Další informace viz “MQAIR-záznam ověřovacích informací” na stránce 249. 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.

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.

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 musí být zcela v malých písmenech. Pokud jste nakonfigurovali hardware se smíšeným rozlišením velkých a malých písmen, překonfigurujte jej s tímto malým popiskem.

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.

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.

FipsRequired (MQLONG)
Produkt WebSphere MQ lze konfigurovat s kryptografickým hardwarem, aby použité kryptografické moduly byly ty, které jsou poskytovány hardwarovým produktem; tyto mohou být FIPS certifikovány na konkrétní úroveň v závislosti na používaném šifrovacím produktu hardwaru. Toto pole slouží k určení, že se budou používat pouze algoritmy certifikované podle standardu FIPS, je-li šifrování poskytováno v softwaru WebSphere MQ.

Při instalaci produktu WebSphere MQ je nainstalována i implementace šifrování SSL, která poskytuje některé moduly s certifikací 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 WebSphere MQ :
  - TLS_RSA_WITH_3DES_EDE_CBC_SHA
  - TLS_RSA_WITH_AES_128_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA

MQSSL_FIPS_YES
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 SSL 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, je-li nakonfigurováno pouze FIPS- CipherSpecs , pak klient MQI odmítne připojení, která určují non-FIPS CipherSpec s MQRC_SSL_INITIALIZATION_ERROR. Produkt WebSphere MQ nezaručuje odmítnutí všech takových připojení a je vaší odpovědností určit, zda je vaše konfigurace produktu WebSphere MQ kompatibilní se standardem FIPS-.

distributed *KeyRepository (MQCHAR256)*

Toto pole je relevantní pouze pro klienty WebSphere MQ MQI spuštěné na systémech UNIX, Linuxa 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.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 MQRC_KEY_REPOSITORY_ERROR.

Chcete-li spustit připojení SSL z klienta WebSphere MQ MQI, 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.

Počet KeyReset(MQLONG)
To představuje celkový počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL nebo TLS, než je znovu vyjednán tajný klíč.

Počet bajtů zahrnuje řídicí informace odeslané agentem MCA.

Zadáte-li počet obnovení tajných klíčů zabezpečení SSL nebo TLS v rozsahu od 1 bajtu do 32 KB, budou kanály SSL nebo 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 resetování tajného klíče SSL nebo 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.

StrucId (MQCHAR4)

Jedná se o identifikátor struktury; hodnota musí být:

ID_KONSTRUKCE_MQSCO_
Identifikátor struktury voleb konfigurace SSL.

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 voleb konfigurace SSL Version-1 .

MQSCO_VERSION_2
Struktura voleb konfigurace SSL Version-2 .

MQSCO_VERSION_3
Struktura voleb konfigurace SSL Version-3 .

MQSCO_VERSION_4
Struktura voleb konfigurace SSL Version-4 .

Následující konstanta uvádí číslo verze aktuální verze:

AKTUÁLNÍ_VERZE MQSCO_
Aktuální verze struktury voleb konfigurace SSL.

Toto je vždy vstupní pole. Počáteční hodnota tohoto pole je MQSCO_VERSION_1.

Počáteční hodnoty a jazyková prohlášení pro MQSCO

Tabulka 546. Počáteční hodnoty polí v MQSCO.

Popis polí v produktu MQSCO a jejich počátečních hodnot

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQSCO_STRUC_ID	'SCO¬'
Version	MQSCO_CURRENT_VERSION	1
KeyRepository	Není	Nulový řetězec nebo prázdné znaky
CryptoHardware	Není	Nulový řetězec nebo prázdné znaky
AuthInfoRecCount	Není	0
AuthInfoRecOffset	Není	0
AuthInfoRecPtr	Není	Nulový ukazatel nebo bajty null
KeyResetCount	MQSCO_RESET_COUNT_DEFAULT	0
FipsRequired	MQSSL_FIPS_NO	0
EncryptionPolicySuiteB	MQ_SUITE_B_NONE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE, MQ_SUITE_B_NOT_AVAILABLE	1, 0, 0, 0
CertificateValPolicy	MQ_CERT_VAL_POLICY_DEFAULT	0

Notes:

1. The symbol ¬ represents a single blank character.
2. In the C programming language, the macro variable MQSCO_DEFAULT contains the values listed above. Use it in the following way to provide initial values for the fields in the structure:

```
MQSCO MySCO = {MQSCO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQSCO MQSCO;
struct tagMQSCO {
   MQCHAR4    StrucId;                  /* Structure identifier */
   MQLONG     Version;                  /* Structure version number */
   MQCHAR256  KeyRepository;            /* Location of SSL key */
                                        /* repository */
   MQCHAR256  CryptoHardware;           /* Cryptographic hardware */
                                        /* configuration string */
   MQLONG     AuthInfoRecCount;         /* Number of MQAIR records */
                                        /* present */
   MQLONG     AuthInfoRecOffset;        /* Offset of first MQAIR */
                                        /* record from start of */
                                        /* MQSCO structure */
   PMQAIR     AuthInfoRecPtr;           /* Address of first MQAIR */
                                        /* record */
/* Ver:1 */
   MQLONG     KeyResetCount;            /* Number of unencrypted */
                                        /* bytes sent/received */
                                        /* before secret key is */
                                        /* reset */
   MQLONG     FipsRequired;             /* Using FIPS-certified */
/* Ver:2 */
```

```
                                        /* algorithms */
   MQLONG     EncryptionPolicySuiteB[4];  /* Use only Suite B */
/* Ver:3 */
                                        /* cryptographic algorithms */
   MQLONG     CertificateValPolicy;       /* Certificate validation */
                                        /* policy */
/* Ver:4 */
```

Deklarace COBOL

```
**   MQSCO structure
  10 MQSCO.
**    Structure identifier
   15 MQSCO-STRUCID                 PIC X(4).
**    Structure version number
   15 MQSCO-VERSION                 PIC S9(9) BINARY.
**    Location of SSL key repository
   15 MQSCO-KEYREPOSITORY           PIC X(256).
**    Cryptographic hardware configuration string
   15 MQSCO-CRYPTOHARDWARE          PIC X(256).
**    Number of MQAIR records present
   15 MQSCO-AUTHINFORECCOUNT        PIC S9(9) BINARY.
**    Offset of first MQAIR record from start of MQSCO structure
   15 MQSCO-AUTHINFORECOFFSET       PIC S9(9) BINARY.
** Address of first MQAIR record
    15  MQSCO-AUTHINFORECPTR POINTER.
** Version 1 **
** Number of unencrypted bytes sent/received before secret key is
** reset
15  MQSCO-KEYRESETCOUNT PIC S9(9) BINARY.
** Using FIPS-certified algorithms
15  MQSCO-FIPSREQUIRED PIC S9(9) BINARY.
** Version 2 **
** Use only Suite B cryptographic algorithms
15  MQSCO-ENCRYPTIONPOLICYSUITEB PIC S9(9) BINARY OCCURS 4.
** Version 3 **
** Certificate validation policy setting
15  MQSCO-CERTIFICATEVALPOLICY PIC S9(9) BINARY.
** Version 4
```

Deklarace PL/I

```
dcl
 1 MQSCO based,
  3 StrucId                   char(4),       /* Structure identifier */
  3 Version                   fixed bin(31), /* Structure version number */
  3 KeyRepository             char(256),     /* Location of SSL key
                                                repository */
  3 CryptoHardware            char(256),     /* Cryptographic hardware
                                                configuration string */
  3 AuthInfoRecCount          fixed bin(31), /* Number of MQAIR records
                                                present */
  3 AuthInfoRecOffset         fixed bin(31), /* Offset of first MQAIR record
                                                from start of MQSCO structure */
  3 AuthInfoRecPtr            pointer,       /* Address of first MQAIR record */
  3 KeyResetCount             fixed bin(31), /* Key reset count */
/* Version 1 */
  3 FipsRequired              fixed bin(31), /* FIPS required */
/* Version 2 */
  3 EncryptionPolicySuiteB (4) fixed bin(31), /* Suite B encryption policy */
/* Version 3 */
  3 CertificateValPolicy      fixed bin(31); /* Certificate validation policy */
/* Version 4 */
```

Deklarace jazyka Visual Basic

```
Type MQSCO
  StrucId           As String*4   'Structure identifier'
  Version           As Long       'Structure version number'
  KeyRepository     As String*256 'Location of SSL key repository'
  CryptoHardware    As String*256 'Cryptographic hardware configuration'
                                  'string'
  AuthInfoRecCount  As Long       'Number of MQAIR records present'
  AuthInfoRecOffset As Long       'Offset of first MQAIR record from'
```

```
                                    'start of MQSCO structure'
  AuthInfoRecPtr    As MQPTR        'Address of first MQAIR record'
  KeyResetCount     As Long         'Number of unencrypted bytes sent/received before secret key
is reset'
'Version 1'
  FipsRequired      As Long         'Mandatory FIPS CipherSpecs?'
'Version 2'
End Type
```

MQSD-Deskriptor odběru

Následující tabulka shrnuje pole ve struktuře.

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby
ObjectName	Název objektu	ObjectName
AlternateUserId	Jméno alternativního uživatele	AlternateUserid
AlternateSecurityId	Alternativní ID zabezpečení	AlternateSecurityId
SubExpiry	Vypršení platnosti odběru	SubExpiry
ObjectString	Řetězec objektu	ObjectString
SubName	Název odběru	SubName
SubUserData	Uživatelská data odběru	SubUserData
SubCorrelId	ID korelace odběru	SubCorrelId
PubPriority	Priorita publikování	PubPriority
PubAccountingToken	Token evidence publikování	PubAccountingToken
PubAppIdentityData	Data identity aplikace Publication	PubAppIdentityData
SelectionString	Řetězec poskytující kritéria výběru	SelectionString
SubLevel	Úroveň odběru	SubLevel
ResObjectString	Název dlouhého objektu	ResObjectString

Přehled pro MQSD

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OSa klienti WebSphere MQ MQI, kteří jsou připojeni k těmto systémům.

Účel: Struktura MQSD se používá k určení podrobností o vytvářeného odběru.

Struktura je vstupním/výstupním parametrem pro volání MQSUB. Další informace naleznete v tématu Poznámky k použití MQSUB.

Spravované odběry: Pokud aplikace nemá specifickou potřebu používat určitou frontu jako cíl pro ty publikace, které odpovídají jejímu odběru, může použít funkci spravovaného odběru. Pokud aplikace rozhodne o použití spravovaného odběru, informuje odběratele o místu určení, kam jsou odesílány publikované zprávy, a to poskytnutím obslužné rutiny objektu jako výstupu z volání MQSUB. Další informace najdete v tématu Hobj (MQHOBJ)-vstup/výstup.

Po odebrání odběru se správce front také zaváže k vyčištění zpráv, které nebyly načteny ze spravovaného místa určení, v následujících situacích:

- Je-li odběr odebrán-použitím funkce MQCLOSE s MQCO_REMOVE_SUB-a spravovaným objekt Hobj je uzavřen.

- Implicitní, je-li připojení ztraceno k aplikaci s použitím trvalého odběru (MQSO_NON_DURABLE)
- Po vypršení platnosti odběru dojde k vypršení platnosti odběru, protože jeho platnost vypršela a spravovaný objekt Hobj je uzavřen.

Je třeba použít spravované odběry s netrvalými odběry, aby mohlo dojít k vyčištění a aby se zprávy pro uzavřené dočasné odběry neprojevily ve vašem správci front. Trvalé odběry mohou také používat spravovaná místa určení.

Verze: Aktuální verze produktu MQSD je MQSD_VERSION_1.

Znaková sada a kódování: Data ve struktuře MQSD musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front zadaného MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQSD

Struktura MQSD obsahuje následující pole; pole jsou popsána v abecedním pořadí:

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 441 v datovém typu MQOD.

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 používá 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.

ObjectName (MQCHAR48)

Jedná se o název objektu tématu, jak je definován v lokálním správci front.

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. Použijte znak null pro označení konce významných dat v názvu; hodnoty null a libovolné znaky následující za ním jsou považovány za mezery. V označeném prostředí platí následující omezení:

- V systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OS:
  - Vyhněte se názvům, které začínají nebo končí podtržítkem; nemohou být zpracovány operacemi a řídicími panely.
  - Znak procentní části má speciální význam pro RACF. Je-li jako externí správce zabezpečení použit modul RACF , nesmí názvy obsahovat žádné procento. Pokud ano, tyto názvy nejsou při použití generických profilů RACF zahrnuty do žádných kontrol zabezpečení.
- V systému IBM imusí být názvy obsahující malá písmena, dopředné lomítko nebo procento, pokud jsou zadány v příkazech, uzavřeny do uvozovek. Tyto uvozovky nesmí být uvedeny pro názvy, které se vyskytují jako pole ve strukturách nebo jako parametry na voláních.

ObjectName se používá k vytvoření úplného názvu tématu.

Úplný název tématu může být sestaven ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o způsobu použití těchto dvou polí naleznete v tématu “Použití řetězců témat” na stránce 538.

Pokud nelze nalézt objekt identifikovaný polem *ObjectName* , volání selže s kódem příčiny MQRC_UNKNOWN_OBJECT_NAME i v případě, že je v souboru *ObjectString*zadán řetězec.

Při návratu z volání MQSUB s použitím volby MQSO_RESUME je toto pole nezměněno.

Délka tohoto pole je dána hodnotou MQ_TOPIC_NAME_LENGTH. Počáteční hodnota tohoto pole je řetězec s hodnotou null v C a 48 prázdných znaků v jiných programovacích jazycích.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze změnit název objektu tématu přihlášené k odběru. Toto pole a pole *ObjectString* lze vynechat. Pokud jsou k dispozici, musí se přeložit na stejný úplný název tématu. Pokud tomu tak není, volání selže s MQRC_TOPC_NOT_ALTERABLE.

ObjectString (MQCHARV)

Toto je dlouhé jméno objektu, které se má použít.

ObjectString se používá k vytvoření úplného názvu tématu.

Úplný název tématu může být sestaven ze dvou různých polí: *ObjectName* a *ObjectString*. Podrobnosti o způsobu použití těchto dvou polí naleznete v tématu “Použití řetězců témat” na stránce 538.

Maximální délka *ObjectString* je 10240.

Pokud *ObjectString* není správně zadáno, podle popisu 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 počáteční hodnoty v rámci struktury MQCHARV.

Pokud jsou v produktu *ObjectString* zástupné znaky, interpretace těchto zástupných znaků lze řídit pomocí voleb zástupných znaků zadaných v poli Volby MQSD.

Při návratu z volání MQSUB s použitím volby MQSO_RESUME je toto pole nezměněno. Úplný název tématu, který se používá, je vrácen v poli *ResObjectString* , je-li k dispozici vyrovnávací paměť.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze změnit dlouhý název objektu tématu, který je k odběru změněn. Toto pole a pole *ObjectName* lze vynechat. Pokud jsou poskytnuty, musí se přeložit na stejný úplný název tématu, nebo se volání nezdaří s MQRC_TOPC_NOT_ALTERABLE.

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

Hodnoty, které uvedete pro volby, lze použít následujícími způsoby:

- Hodnoty lze přidat. Nepřidávejte stejnou konstantu více než jednou.
- Hodnoty lze kombinovat s použitím 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. V tabulce jsou zobrazeny platné kombinace voleb přístupu a vytváření.

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 547. 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 529)	MQRC_GROUPING_NOT_ALTERABLE, pokusíte-li se změnit MQSO_GROUP_SUB
MQSD.	Volby publikování	Ano (viz poznámka "2" na stránce 529)	Není
MQSD.	Volby zástupného znaku	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	Další volby	Ne (viz poznámka "3" na stránce 529)	Není
MQSD.	ObjectName	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	AlternateUserid	Ne (viz poznámka "4" na stránce 529)	Není
MQSD.	AlternateSecurityId	Ne (viz poznámka "4" na stránce 529)	Není
MQSD.	SubExpiry	Ano	Není
MQSD.	ObjectString	Ne	MQRC_TOPIC_NOT_ALTERABLE
MQSD.	SubName	Ne (viz poznámka "5" na stránce 529)	Není
MQSD.	SubUserData	Ano	Není
MQSD.	SubCorrelId	Ano (viz poznámka "6" na stránce 529)	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 529)	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. Není-li aplikace spuštěna jako klient WebSphere 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.

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 dodává přesně význam poskytnutý v produktech WebSphere MQ V6 a WebSphere MB V6 při použití formátovaných zpráv MQRFH1 pro publikování a 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:

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
```

Poznámka: Toto použití zástupných znaků dodává význam poskytnutý v produktu WebSphere Message Broker verze 6 při použití formátovaných zpráv MQRFH2 pro publikování a odběr.

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 525.

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 536 .

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:

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. Tato volba v systému z/OSpro aplikaci CICS nebo IMS také vynutí selhání volání MQSUB, pokud je připojení ve stavu uvedení do klidového stavu.

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 386 .

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 388 .

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.

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 410.

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 410.

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.

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.

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 proměnné délky” na stránce 271 , nebo pokud překročí maximální délku, volání selže s kódem příčiny MQRC_SELECTION_STRING_ERROR.

Použití SelectionString je popsáno v Selektory .

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.

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.

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.

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.

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.

SubName (MQCHARV)

Určuje název odběru. Toto pole je povinné pouze v případě, že proměnná *Options* určuje volbu MQSO_DURABLE, ale bude-li ji zadán správce front pro MQSO_NON_DURABLE, bude tento parametr také použit.

Je-li tato volba zadána, musí být *SubName* v rámci správce front jedinečná, protože se jedná o metodu použitou k identifikaci odběru.

Maximální délka *SubName* je 10240.

Toto pole slouží dvěma účelům. Pro odběr MQSO_DURABLE můžete toto pole použít k identifikaci odběru, abyste jej mohli obnovit po vytvoření v případě, že jste buď zavřeli popisovač odběru (pomocí volby MQCO_KEEP_SUB), nebo jste byli odpojeni od správce front. To lze provést pomocí volání MQSUB s volbou MQSO_RESUME. Zobrazí se také v administrativním zobrazení odběrů v poli SUBNAME v DISPLAY SBSTATUS.

Pokud je parametr *SubName* zadán nesprávně, je v souladu s popisem použití struktury MQCHARV ponechán na požadovaném umístění (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 počáteční hodnoty v rámci struktury MQCHARV.

Pokud změníte existující odběr pomocí volby MQSO_ALTER, nelze název odběru změnit, protože se jedná o identifikující pole použité k vyhledání odkazovaného odběru. Ve výstupu z volání MQSUB s volbou MQSO_RESUME se tato hodnota nezmění.

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.

Použití řetězců témat

Téma je sestaveno z dílčího tématu identifikovaného v objektu tématu a z dílčího tématu poskytnutého aplikací. Jako název tématu můžete použít buď dílčí téma, nebo je zkombinovat a vytvořit nový název tématu.

V programu MQI je úplný název tématu vytvořen produktem MQOPEN. Skládá se ze dvou polí používaných ve voláních publikování/odběru MQI, v uvedeném pořadí:

1. Atribut **TOPICSTR** objektu tématu, pojmenovaný v poli **ObjectName** .
2. Parametr **ObjectString** definující dílčí téma poskytované aplikací.

Výsledný řetězec tématu se vrátí v parametru **ResObjectString** .

Tato pole se považují za přítomná, pokud první znak každého pole není prázdný znak nebo znak null a délka pole je větší než nula. Je-li přítomno pouze jedno z těchto polí, použije se nezměněno jako název tématu. Pokud žádné pole nemá hodnotu, volání selže s kódem příčiny MQRC_UNKNOWN_OBJECT_NAME nebo MQRC_TOPIC_STRING_ERROR, pokud úplný název tématu není platný.

Jsou-li přítomna obě pole, bude mezi dva prvky výsledného kombinovaného názvu tématu vložen znak '/'.

Tabulka 548 na stránce 539 uvádí příklady zřetězení řetězce tématu:

Tabulka 548. Příklady zřetězení řetězce tématu

TOPICSTR	ObjectString	Úplný název tématu	Komentář
Fotbal/Scores	' '	Fotbal/Scores	TOPICSTR se používá samostatně.
' '	Fotbal/Scores	Fotbal/Scores	Hodnota ObjectString se používá samostatně.
Fotbal	Skóre	Fotbal/Scores	Ve spojovacím bodu je přidán znak ' /'
Fotbal	/Skóre	Fotbal//skóre	Mezi těmito dvěma řetězci je vytvořen 'prázdný uzel'
/Fotbal	Skóre	/Fotbal/Scores	Téma začíná znakem 'prázdný uzel'

Znak '/' je považován za speciální znak poskytující strukturu úplnému názvu tématu v tématu Stromy témat a nesmí být použit pro žádný jiný důvod, protože je ovlivněna struktura stromu témat. Téma "/Football" není stejné jako téma "Football".

Následující zástupné znaky jsou speciální znaky:

- znaménko plus '+'
- číselný znak '#'
- hvězdička '*'
- otazník '?'

Tyto znaky se nepovažují za neplatné, musíte však zajistit, abyste pochopili, jak jsou používány. Při publikování můžete raději nepoužívat tyto znaky ve vašich řetězcích témat. Publikování na téma řetězce s '#' nebo '+' smíšeným s ostatními znaky (včetně samotných) v rámci úrovně tématu může být přihlášeno k odběru buď se zástupným schématem. Publikování v řetězci tématu s hodnotou '#' nebo '+' jako jediný znak mezi dvěma znaky '/' vytváří řetězec tématu, který nemůže být přihlášen k odběru explicitně aplikací pomocí schématu zástupného znaku MQSO_WILDCARD_TOPIC. Tato situace vede k tomu, že aplikace bude dostávat více publikací, než se očekávalo.

Příklad úseku kódu

Tento úsek kódu extrahovaný z ukázkového programu Příklad 2: Vydavatel na téma s proměnnoukombinuje objekt tématu s řetězcem tématu proměnné.

```
MQOD     td = {MQOD_DEFAULT};  /* Object Descriptor            */
td.ObjectType = MQOT_TOPIC;    /* Object is a topic            */
td.Version = MQOD_VERSION_4;   /* Descriptor needs to be V4    */
strncpy(td.ObjectName, topicName,  MQ_TOPIC_NAME_LENGTH);
td.ObjectString.VSPtr = topicString;
td.ObjectString.VSLength = (MQLONG)strlen(topicString);
td.ResObjectString.VSPtr = resTopicStr;
td.ResObjectString.VSBufSize = sizeof(resTopicStr)-1;
MQOPEN(Hconn, &td, MQOO_OUTPUT | MQOO_FAIL_IF_QUIESCING, &Hobj, &CompCode, &Reason);
```

Počáteční hodnoty a deklarace jazyka pro MQSD

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY OBJEKTU MQSD_STRUCT	'SD¬¬'
Version	MQSD_VERSION_1	1
Options	MQSO_NON_DURABLE	0
ObjectName	Není	Nulový řetězec nebo prázdné znaky
AlternateUserId	Není	Nulový řetězec nebo prázdné znaky
AlternateSecurityId	MQSID_NONE	Hodnoty null
SubExpiry	MQEI_UNLIMITED	-1
ObjectString	Není	Názvy a hodnoty definované pro MQCHARV
SubName	Není	Názvy a hodnoty definované pro MQCHARV
SubUserData	Není	Názvy a hodnoty definované pro MQCHARV
SubCorrelId	MQCI_NONE	Hodnoty null
PubPriority	MQPRI_PRIORITY_AS_Q_DEF	-3
PubAccountingToken	MQACT_NONE	Hodnoty null
PubApplIdentityData	Není	Nulový řetězec nebo prázdné znaky
Selection String	Není	Názvy a hodnoty definované pro MQCHARV
SubLevel	Není	1
ResObjectString	Není	Názvy a hodnoty definované pro MQCHARV

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.
3. V programovacím jazyce C-proměnná makraHodnota MQSD_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQSD MySD = {MQSD_DEFAULT};
```

Deklarace C

```
typedef struct tagMQSD MQSD;
struct tagMQSD {
  MQCHAR4   StrucId;             /* Structure identifier */
  MQLONG    Version;             /* Structure version number */
  MQLONG    Options;             /* Options associated with subscribing */
```

```
  MQCHAR48  ObjectName;           /* Object name */
  MQCHAR12  AlternateUserId;      /* Alternate user identifier */
  MQBYTE40  AlternateSecurityId;  /* Alternate security identifier */
  MQLONG    SubExpiry;            /* Expiry of Subscription */
  MQCHARV   ObjectString;         /* Object Long name */
  MQCHARV   SubName;              /* Subscription name */
  MQCHARV   SubUserData;          /* Subscription User data */
  MQBYTE24  SubCorrelId;          /* Correlation Id related to this subscription */
  MQLONG    PubPriority;          /* Priority set in publications */
  MQBYTE32  PubAccountingToken;   /* Accounting Token set in publications */
  MQCHAR32  PubApplIdentityData;  /* Appl Identity Data set in publications */
  MQCHARV   SelectionString;      /* Message selector structure */
  MQLONG    SubLevel;             /* Subscription level */
  MQCHARV   ResObjectString;      /* Resolved Long object name*/
  /* Ver:1 */
};
```

Deklarace COBOL

```
** Address of variable length string
   20  MQSD-OBJECTSTRING-VSPTR         POINTER.
** Offset of variable length string
   20  MQSD-OBJECTSTRING-VSOFFSET      PIC S9(9) BINARY.
** size of buffer
   20  MQSD-OBJECTSTRING-VSBUFSIZE      PIC S9(9) BINARY.
** Length of variable length string
   20  MQSD-OBJECTSTRING-VSLENGTH      PIC S9(9) BINARY.
** CCSID of variable length string
   20  MQSD-OBJECTSTRING-VSCCSID       PIC S9(9) BINARY.
** Subscription name
 15  MQSD-SUBNAME.
** Address of variable length string
   20  MQSD-SUBNAME-VSPTR              POINTER.
** Offset of variable length string
   20  MQSD-SUBNAME-VSOFFSET           PIC S9(9) BINARY.
** size of buffer
   20  MQSD-SUBNAME-VSBUFSIZE           PIC S9(9) BINARY.
** Length of variable length string
   20  MQSD-SUBNAME-VSLENGTH           PIC S9(9) BINARY.
** CCSID of variable length string
   20  MQSD-SUBNAME-VSCCSID            PIC S9(9) BINARY.
** Subscription User data
 15  MQSD-SUBUSERDATA.
** Address of variable length string
   20  MQSD-SUBUSERDATA-VSPTR          POINTER.
** Offset of variable length string
   20  MQSD-SUBUSERDATA-VSOFFSET       PIC S9(9) BINARY.
** size of buffer
   20  MQSD-SUBUSERDATA-VSBUFSIZE       PIC S9(9) BINARY.
** Length of variable length string
   20  MQSD-SUBUSERDATA-VSLENGTH       PIC S9(9) BINARY.
** CCSID of variable length string
   20  MQSD-SUBUSERDATA-VSCCSID        PIC S9(9) BINARY.
** Correlation Id related to this subscription
 15  MQSD-SUBCORRELID                 PIC X(24).
** Priority set in publications
 15  MQSD-PUBPRIORITY                 PIC S9(9) BINARY.
** Accounting Token set in publications
 15  MQSD-PUBACCOUNTINGTOKEN          PIC X(32).
** Appl Identity Data set in publications
 15  MQSD-PUBAPPLIDENTITYDATA         PIC X(32).
** Message Selector
 15 MQSD-SELECTIONSTRING.
** Address of variable length string
   20  MQSD-SELECTIONSTRING-VSPTR     POINTER.
** Offset of variable length string
   20  MQSD-SELECTIONSTRING-VSOFFSET       PIC S9(9) BINARY.
** size of buffer
   20  MQSD-SELECTIONSTRING-VSBUFSIZE      PIC S9(9) BINARY.
** Length of variable length string
   20  MQSD-SELECTIONSTRING-VSLENGTH       PIC S9(9) BINARY.
** CCSID of variable length string
   20  MQSD-SELECTIONSTRING-VSCCSID        PIC S9(9) BINARY.
** Selection criteria
   20  MQSD-SELECTIONSTRING-SUBLEVEL       PIC S9(9) BINARY.
** Long object name
   20  MQSD-SELECTIONSTRING-RESOBJSTRING   PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQSD based,
  3 StrucId             char(4),       /* Structure identifier */
  3 Version             fixed bin(31), /* Structure version number */
  3 Options             fixed bin(31), /* Options associated with subscribing */
  3 ObjectName          char(48),      /* Object name */
  3 AlternateUserId     char(12),      /* Alternate user identifier */
  3 AlternateSecurityId char(40),      /* Alternate security identifier */
  3 SubExpiry           fixed bin(31), /* Expiry of Subscription */
  3 ObjectString,                      /* Object Long name */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31); /* CCSID of variable length string */
  3 SubName,                           /* Subscription name */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31); /* CCSID of variable length string */
  3 SubUserData,                       /* Subscription User data */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31), /* CCSID of variable length string */
  3 SubCorrelId         char(24),      /* Correlation Id related to this subscription */
  3 PubPriority         fixed bin(31), /* Priority set in publications */
  3 PubAccountingToken  char(32),      /* Accounting Token set in publications */
  3 PubApplIdentityData char(32),      /* Appl Identity Data set in publications */
  3 SelectionString,                   /* Message Selection */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31), /* CCSID of variable length string */
  3 SubLevel            fixed bin(31), /* Subscription level */
  3 ResObjectString,                   /* Resolved Long object name */
   5 VSPtr              pointer,       /* Address of variable length string */
   5 VSOffset           fixed bin(31), /* Offset of variable length string */
   5 VSBufSize          fixed bin(31), /* size of buffer */
   5 VSLength           fixed bin(31), /* Length of variable length string */
   5 VSCCSID            fixed bin(31); /* CCSID of variable length string */
```

Deklarace High Level Assembler

```
MQSD                           DSECT
MQSD_STRUCID                   DS    CL4   Structure identifier
MQSD_VERSION                   DS    F     Structure version number
MQSD-OPTIONS                   DS    F     Options associated with subscribing
MQSD_OBJECTNAME                DS    CL48  Object name
MQSD_ALTERNATEUSERID           DS    CL12  Alternate user identifier
MQSD_ALTERNATESECURITYID       DS    CL40  Alternate security identifier
MQSD_SUBEXPIRY                 DS    F     Expiry of Subscription
MQSD_OBJECTSTRING              DS    0F    Object Long name
MQSD_OBJECTSTRING_VSPTR        DS    F     Address of variable length string
MQSD_OBJECTSTRING_VSOFFSET     DS    F     Offset of variable length string
MQSD_OBJECTSTRING_VSBUFSIZE    DS    F     size of buffer
MQSD_OBJECTSTRING_VSLENGTH     DS    F     Length of variable length string
MQSD_OBJECTSTRING_VSCCSID      DS    F     CCSID of variable length string
MQSD_OBJECTSTRING_LENGTH       EQU   *-MQSD_OBJECTSTRING
                               ORG   MQSD_OBJECTSTRING
MQSD_OBJECTSTRING_AREA         DS    CL(MQSD_OBJECTSTRING_LENGTH)
*
MQSD_SUBNAME                   DS    0F    Subscription name
MQSD_SUBNAME_VSPTR             DS    F     Address of variable length string
MQSD_SUBNAME_VSOFFSET          DS    F     Offset of variable length string
MQSD_SUBNAME_VSBUFSIZE         DS    F     size of buffer
MQSD_SUBNAME_VSLENGTH          DS    F     Length of variable length string
MQSD_SUBNAME_VSCCSID           DS    F     CCSID of variable length string
MQSD_SUBNAME_LENGTH            EQU   *-MQSD_SUBNAME
                               ORG   MQSD_SUBNAME
MQSD_SUBNAME_AREA              DS    CL(MQSD_SUBNAME_LENGTH)
*
MQSD_SUBUSERDATA               DS    0F    Subscription User data
```

```
MQSD_SUBUSERDATA_VSPTR          DS    F     Address of variable length string
MQSD_SUBUSERDATA_VSOFFSET       DS    F     Offset of variable length string
MQSD_SUBUSERDATA_VSBUFSIZE      DS    F     size of buffer
MQSD_SUBUSERDATA_VSLENGTH       DS    F     Length of variable length string
MQSD_SUBUSERDATA_VSCCSID        DS    F     CCSID of variable length string
MQSD_SUBUSERDATA_LENGTH         EQU   *-MQSD_SUBUSERDATA
                                ORG   MQSD_SUBUSERDATA
MQSD_SUBUSERDATA_AREA           DS    CL(MQSD_SUBUSERDATA_LENGTH)
*
MQSD_SUBCORRELID                DS    CL24  Correlation Id related to this subscription
MQSD_PUBPRIORITY                DS    F     Priority set in publications
MQSD_PUBACCOUNTINGTOKEN         DS    CL32  Accounting Token set in publications
MQSD_PUBAPPLIDENTITYDATA        DS    CL32  Appl Identity Data set in publications
*
MQSD_SELECTIONSTRING            DS   F     Message Selector
MQSD_SELECTIONSTRING_VSPTR      DS   F     Address of variable length string
MQSD_SELECTIONSTRING_VSOFFSET   DS   F     Offset of variable length string
MQSD_SELECTIONSTRING_VSBUFSIZE  DS   F     size of buffer
MQSD_SELECTIONSTRING_VSLENGTH   DS   F     Length of variable length string
MQSD_SELECTIONSTRING_VSCCSID    DS   F     CCSID of variable length string
MQSD_SELECTIONSTRING_LENGTH     EQU  *- MQSD_SELECTIONSTRING
                                ORG  MQSD_SELECTIONSTRING
MQSD_SELECTIONSTRING_AREA       DS   CL(MQSD_SELECTIONSTRING_LENGTH)
*
MQSD-SUBLEVEL                   DS   F     Subscription level
*
MQSD_RESOBJECTSTRING            DS   F     Resolved Long object name
MQSD_RESOBJECTSTRING_VSPTR      DS   F     Address of variable length string
MQSD_RESOBJECTSTRING_VSOFFSET   DS   F     Offset of variable length string
MQSD_RESOBJECTSTRING_VSBUFSIZE  DS   F     size of buffer
MQSD_RESOBJECTSTRING_VSLENGTH   DS   F     Length of variable length string
MQSD_RESOBJECTSTRING_VSCCSID    DS   F     CCSID of variable length string
MQSD_RESOBJECTSTRING_LENGTH     EQU  *- MQSD_RESOBJECTSTRING
                                ORG  MQSD_RESOBJECTSTRING
MQSD_RESOBJECTSTRING_AREA       DS   CL(MQSD_RESOBJECTSTRING_LENGTH)
*
MQSD_LENGTH                     EQU   *-MQSD
                                ORG   MQSD
MQSD_AREA                       DS    CL(MQSD_LENGTH)
```

MQSMPO-Nastavení voleb vlastností zprávy

Následující tabulka shrnuje pole ve struktuře.

Tabulka 549. Pole v MQSMPO		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby
ValueEncoding	Kódování hodnoty vlastnosti	ValueEncoding
ValueCCSID	Znaková sada hodnoty vlastnosti	ValueCCSID

Přehled pro MQSMPO

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

Účel: Struktura **MQSMPO** umožňuje aplikacím zadávat volby, které řídí způsob nastavení vlastností zpráv. Struktura je vstupním parametrem na volání **MQSETMP** .

Znaková sada a kódování: Data v souboru **MQSMPO** musí být ve znakové sadě aplikace a kódování aplikace (**MQENC_NATIVE**).

Pole pro MQSMPO

Struktura MQSMPO obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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**.

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**.

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**.

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**.

Počáteční hodnoty a deklarace jazyka pro MQSMPO

Tabulka 550. Počáteční hodnoty polí v MQSMPO

Název pole	Název konstanty	Hodnota konstanty
StrucId	MQSMPO_STRUCTION_ID	'SMPO'
Version	MQSMPO_VERSION_1	1
Options	MQSMPO_NONE	0
ValueEncoding	MQENC_NATIVE	Závisí na prostředí
ValueCCSID	MQCCSI_APPL	-3

Notes:

1. Hodnota Null řetězce nebo mezery označuje řetězec s hodnotou null v C a prázdné znaky v jiných programovacích jazycích.
2. V programovacím jazyce C-proměnná makraHodnota MQSMPO_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQSMPO MySMPO = {MQSMPO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQSMPO MQSMPO;
struct tagMQSMPO {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    Options;         /* Options that control the action of MQSETMP */
  MQLONG    ValueEncoding;   /* Encoding of Value */
  MQLONG    ValueCCSID;      /* Character set identifier of Value */
};
```

Deklarace COBOL

```
**   MQSMPO structure
  10 MQSMPO.
**    Structure identifier
   15 MQSMPO-STRUCID       PIC X(4).
**    Structure version number
   15 MQSMPO-VERSION       PIC S9(9) BINARY.
**    Options that control the action of MQSETMP
   15 MQSMPO-OPTIONS       PIC S9(9) BINARY.
**    Encoding of VALUE
   15 MQSMPO-VALUEENCODING PIC S9(9) BINARY.
**    Character set identifier of VALUE
   15 MQSMPO-VALUECCSID    PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQSMPO based,
   3 StrucId        char(4),       /* Structure identifier */
   3 Version        fixed bin(31), /* Structure version number */
   3 Options        fixed bin(31), /* Options that control the action of MQSETMP */
   3 ValueEncoding  fixed bin(31), /* Encoding of Value */
   3 ValueCCSID     fixed bin(31), /* Character set identifier of Value */
```

Deklarace High Level Assembler

```
MQSMPO                DSECT
MQSMPO_STRUCID        DS   CL4   Structure identifier
MQSMPO_VERSION        DS   F     Structure version number
MQSMPO_OPTIONS        DS   F     Options that control the action of
*                                MQSETMP
MQSMPO_VALUEENCODING  DS   F     Encoding of VALUE
MQSMPO_VALUECCSID     DS   F     Character set identifier of VALUE
MQSMPO_LENGTH         EQU  *-MQSMPO
MQSMPO_AREA           DS   CL(MQSMPO_LENGTH)
```

MQSRO-Volby požadavku na odběr

Tento oddíl popisuje volby požadavku na odběr, jaká pole obsahuje, a počáteční hodnoty těchto polí.

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
Options	Volby	Volby
NumPubs	Počet publikování	NumPubs

Přehled pro MQSRO

Dostupnost: AIX, HP-UX, IBM i, Solaris, Linux, Windows, z/OS a klienti WebSphere MQ MQI připojené k těmto systémům.

Účel: Struktura MQSRO umožňuje aplikaci určit volby, které řídí způsob provedení požadavku na odběr. Struktura je vstupním/výstupním parametrem pro volání MQSUBRQ.

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 kódováním lokálního správce front uvedeného MQENC_NATIVE. Je-li však aplikace spuštěna jako klient MQ MQI, musí být tato struktura ve znakové sadě a kódování klienta.

Pole pro MQSRO

Struktura MQSRO obsahuje následující pole; pole jsou popsána v abecedním pořadí:

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 položkou *Hsub* , bude jako výsledek tohoto volání odesláno nejvýše jedno publikování.

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 systému z/OSpro aplikaci CICS nebo IMS tato volba také vynutí selhání volání MQSUBRQ, pokud se připojení nachází ve stavu uvedení do klidového stavu.

Výchozí volba: Není-li výše popsaná volba vyžadována, musí být použita následující volba:

MQSRO_NONE

Tuto hodnotu použijte, chcete-li vyjádřit, že nebyly zadány žádné jiné volby. Všem volbám budou přiřazeny jejich výchozí hodnoty.

Funkce MQSRO_NONE pomáhá programovým dokumentaci. Ačkoli se nejedná o zamýšlené použití této volby, protože její hodnota je nulová, nelze toto použití detekovat.

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.

Počáteční hodnoty a deklarace jazyka pro MQSRO

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQSRO_STRUCTION_ID	'SRO¬'
Version	MQSRO_VERSION_1	1
Options	MQSRO_NONE	0
NumPubs	Není	0

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQSRO_DEFAULT obsahuje výše uvedené hodnoty. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQSRO MySRO = {MQSRO_DEFAULT};
```

Deklarace C

```
typedef struct tagMQSRO MQSRO;
struct tagMQSRO {
  MQCHAR4   StrucId;              /* Structure identifier */
  MQLONG    Version;              /* Structure version number */
  MQLONG    Options;              /* Options that control the action of MQSUBRQ */
  MQLONG    NumPubs;              /* Number of publications sent */
  /* Ver:1 */
};
```

Deklarace COBOL

```
** MQSRO structure
 10  MQSRO.
** Structure identifier
 15  MQSRO-STRUCID             PIC X(4).
** Structure version number
 15  MQSRO-VERSION             PIC S9(9) BINARY.
** Options that control the action of MQSUBRQ
 15  MQSRO-OPTIONS             PIC S9(9) BINARY.
** Number of publications sent
 15  MQSRO-NUMPUBS             PIC S9(9) BINARY.
```

Deklarace PL/I

```
dcl
 1 MQSRO based,
   3 StrucId        char(4),       /* Structure identifier */
   3 Version        fixed bin(31), /* Structure version number */
   3 Options        fixed bin(31), /* Options that control the action of MQSUBRQ */
   3 NumPubs        fixed bin(31); /* Number of publications sent */
```

Deklarace High Level Assembler

```
MQSRO                   DSECT
MQSRO_STRUCID           DS    CL4   Structure identifier
```

```
MQSRO_VERSION           DS    F     Structure version number
MQSRO_OPTIONS           DS    F     Options that control the action of MQSUBRQ
MQSRO_NUMPUBS           DS    F     Number of publications sent
*
MQSRO_LENGTH            EQU   *-MQSRO
                        ORG   MQSRO
MQSRO_AREA              DS    CL(MQSRO_LENGTH)
```

MQSTS-Struktura vytváření sestav o stavu

Následující tabulka shrnuje pole ve struktuře.

Tabulka 551. Pole v MQSTS

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
CompCode	Kód dokončení první chyby	CompCode
Reason	Kód příčiny první chyby	Příčina
PutSuccessCount	Počet úspěšných asynchronních volání vložení	SuccessCount
PutWarningCount	Počet asynchronních volání vložení, která měla varování	WarningCount
PutFailureCount	Počet nezdařených asynchronních volání vložení	FailureCount
ObjectType	Typ selhávajícího objektu	ObjectType
ObjectName	Název selhávajícího objektu	ObjectName
ObjectQMgrName	Název správce front, který vlastní selhávající objekt	ObjectQMgrName
ResolvedObjectName	Vyřešený název cílové fronty	ResolvedObjectNázev
ResolvedQMgrName	Vyřešený název správce cílové fronty	ResolvedQMgrNázev
Poznámka: Zbývající pole se budou ignorovat, pokud je verze nižší než MQSTS_VERSION_2.		
ObjectString	Název dlouhého objektu selhávajícího objektu	ObjectString
SubName	Název odběru selhávajícího odběru	SubName
OpenOptions	Volby otevření přidružené k selhání	OpenOptions
SubOptions	Volby odběru přidružené k selhání	SubOptions

Přehled pro MQSTS

Účel: Struktura MQSTS je výstupní parametr z příkazu MQSTAT.

Znaková sada a kódování: Znaková data v MQSTS se nacházejí ve znakové sadě lokálního správce front; to je dáno atributem správce front *CodedCharSetId* . Číselná data v MQSTS jsou v nativním kódování počítače; to je dáno *Kódováním*.

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-Načíst informace o stavu” na stránce 740.

Pole pro MQSTS

Struktura MQSTS obsahuje níže uvedená pole; pole jsou popsána v **abecedním pořadí**:

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.

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).

PŘEPOJENÍ MQSTAT_TYPE_RECONNECTION

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.

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.

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 146.

ObjectType je výstupní pole. Jeho počáteční hodnota je MQOT_Q.

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.

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.

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.

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.

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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQSTS

Tabulka 552. Počáteční hodnoty polí v MQSTS

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQSTS_	'STAT¬'
Version	MQSTS_VERSION_1	1
CompCode	MQCC_OK	0
Reason	MQRC_NONE	0
PutSuccessCount	Není	0
PutWarningCount	Není	0
PutFailureCount	Není	0
ObjectType	MQOT_Q	1
ObjectName	Není	Nulový řetězec nebo prázdné znaky
ObjectQMgrName	Není	Nulový řetězec nebo prázdné znaky
ResolvedObjectName	Není	Nulový řetězec nebo prázdné znaky

Tabulka 552. Počáteční hodnoty polí v MQSTS (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**
ResolvedQMgrName	Není	Nulový řetězec nebo prázdné znaky
ObjectString	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	{NULL,0,0,0,-3}
SubName	VÝCHOZÍ HODNOTA MQCHARV_DEFAULT	{NULL,0,0,0,-3}
OpenOptions	Není	0
SubOptions	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.
3. V programovacím jazyku C má proměnná makra MQSTS_DEFAULT hodnoty uvedené výše. Může být použit následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQSTS MySTS = {MQSTS_DEFAULT};
```

Deklarace C

```
typedef struct tagMQSTS MQSTS;
struct tagMQSTS {
 MQCHAR4   StrucId;            /* Structure identifier */
 MQLONG    Version;            /* Structure version number */
 MQLONG    CompCode;           /* Completion Code of first error */
 MQLONG    Reason;             /* Reason Code of first error */
 MQLONG    PutSuccessCount;    /* Number of Async calls succeeded */
 MQLONG    PutWarningCount;    /* Number of Async calls had warnings */
 MQLONG    PutFailureCount;    /* Number of Async calls had failures */
 MQLONG    ObjectType;         /* Failing object type */
 MQCHAR48  ObjectName;         /* Failing object name */
 MQCHAR48  ObjectQMgrName;     /* Failing object queue manager name */
 MQCHAR48  ResolvedObjectName; /* Resolved name of destination queue */
 MQCHAR48  ResolvedQMgrName;   /* Resolved name of destination qmgr */
/* Ver:1 */
 MQCHARV   ObjectString;       /* Failing object long name */
 MQCHARV   SubName;            /* Failing subscription name */
 MQLONG    OpenOptions;        /* Failing open options */
 MQLONG    SubOptions;         /* Failing subscription options */
/* Ver:2 */
};
```

Deklarace COBOL

```
      ** MQSTS structure
       10  MQSTS.
      ** Structure identifier
       15  MQSTS-STRUCID PIC X(4).
      ** Structure version number
       15  MQSTS-VERSION PIC S9(9) BINARY.
      ** Completion Code of first error
       15  MQSTS-COMPCODE PIC S9(9) BINARY.
      ** Reason Code of first error
       15  MQSTS-REASON PIC S9(9) BINARY.
      ** Number of Async put calls succeeded
       15  MQSTS-PUTSUCCESSCOUNT PIC S9(9) BINARY.
      ** Number of Async put calls had warnings
       15  MQSTS-PUTWARNINGCOUNT PIC S9(9) BINARY.
      ** Number of Async put calls had failures
       15  MQSTS-PUTFAILURECOUNT PIC S9(9) BINARY.
      ** Failing object type
       15  MQSTS-OBJECTTYPE PIC S9(9) BINARY.
      ** Failing object name
```

```
       15  MQSTS-OBJECTNAME PIC X(48).
      ** Failing object queue manager
       15  MQSTS-OBJECTQMGRNAME PIC X(48).
      ** Resolved name of destination queue
       15  MQSTS-RESOLVEDOBJECTNAME PIC X(48).
      ** Resolved name of destination qmgr
       15  MQSTS-RESOLVEDQMGRNAME PIC X(48).
      ** Ver:1 **
      ** Failing object long name
       15  MQSTS-OBJECTSTRING.
      ** Address of variable length string
       20  MQSTS-OBJECTSTRING-VSPTR POINTER.
      ** Offset of variable length string
       20  MQSTS-OBJECTSTRING-VSOFFSET PIC S9(9) BINARY.
      ** Size of buffer
       20  MQSTS-OBJECTSTRING-VSBUFSIZE PIC S9(9) BINARY.
      ** Length of variable length string
       20  MQSTS-OBJECTSTRING-VSLENGTH PIC S9(9) BINARY.
      ** CCSID of variable length string
       20  MQSTS-OBJECTSTRING-VSCCSID PIC S9(9) BINARY.
      ** Failing subscription name
       15  MQSTS-SUBNAME.
      ** Address of variable length string
       20  MQSTS-SUBNAME-VSPTR POINTER.
      ** Offset of variable length string
       20  MQSTS-SUBNAME-VSOFFSET PIC S9(9) BINARY.
      ** Size of buffer
       20  MQSTS-SUBNAME-VSBUFSIZE PIC S9(9) BINARY.
      ** Length of variable length string
       20  MQSTS-SUBNAME-VSLENGTH PIC S9(9) BINARY.
      ** CCSID of variable length string
       20  MQSTS-SUBNAME-VSCCSID PIC S9(9) BINARY.
      ** Failing open options
       15  MQSTS-OPENOPTIONS PIC S9(9) BINARY.
      ** Failing subscription options
       15  MQSTS-SUBOPTIONS PIC S9(9) BINARY.
      ** Ver:2 **
```

Deklarace PL/I

```
dcl
   1 MQSTS based,
     3 StrucId             char(4),       /* Structure identifier */
      3 Version            fixed bin(31), /* Structure version number */
      3 CompCode           fixed bin(31), /* Completion code */
      3 Reason             fixed bin(31), /* Reason code */
      3 PutSuccessCount    fixed bin(31), /* Put success count */
      3 PutWarningCount    fixed bin(31), /* Put warning count */
      3 PutFailureCount    fixed bin(31), /* Put failure count */
      3 ObjectType         fixed bin(31), /* Object type */
      3 ObjectName         char(48), /* Object name */
      3 ObjectQmgrName     char(48), /* Object queue manager */
      3 ResolvedObjectName char(48), /* Resolved Object name */
      3 ResolvedQmgrName   char(48); /* Resolved Object queue manager */
  /* Ver:1 */
      3 ObjectString,                /* Failing object long name */
         5 VSPtr pointer,            /* Address of variable length string */
         5 VSOffset fixed bin(31),   /* Offset of variable length string */
         5 VSBufSize fixed bin(31),  /* Size of buffer */
         5 VSLength fixed bin(31),   /* Length of variable length string */
         5 VSCCSID fixed bin(31);    /* CCSID of variable length string */
      3 SubName,                     /* Failing subscription name */
         5 VSPtr pointer,            /* Address of variable length string */
         5 VSOffset fixed bin(31),   /* Offset of variable length string */
         5 VSBufSize fixed bin(31),  /* Size of buffer */
         5 VSLength fixed bin(31),   /* Length of variable length string */
         5 VSCCSID fixed bin(31);    /* CCSID of variable length string */
     3 OpenOptions fixed bin(31),    /* Failing open options */
     3 SubOptions fixed bin(31);     /* Failing subscription options */
    /* Ver:2 */
```

Deklarace High Level Assembler

```
     MQSTS                          DSECT
     MQSTS_STRUCID                  DS    CL4   Structure identifier
     MQSTS_VERSION                  DS    F     Structure version number
     MQSTS_COMPCODE                 DS    F     Completion code
     MQSTS_REASON                   DS    F     Reason code
```

```
    MQSTS_PUTSUCCESSCOUNT         DS    F     Success count
    MQSTS_PUTWARNINGCOUNT         DS    F     Warning count
    MQSTS_PUTFAILURECOUNT         DS    F     Failure count
    MQSTS_OBJTYPE                 DS    F     Object type
    MQSTS_OBJNAME                 DS    CL48  Object name
    MQSTS_OBJQMGR                 DS    CL48  Object queue manager
    MQSTS_ROBJNAME                DS    CL48  Resolved object name
    MQSTS_ROBJQMGR                DS    CL48  Resolved object queue manager
    MQSTS_OBJECTSTRING            DS    0F   Force fullword alignment
    MQSTS_OBJECTSTRING_VSPTR      DS    A  Address of variable length string
    MQSTS_OBJECTSTRING_VSOFFSET   DS    F  Offset of variable length string
    MQSTS_OBJECTSTRING_VSBUFSIZE  DS    F  Size of buffer
    MQSTS_OBJECTSTRING_VSLENGTH   DS    F  Length of variable length string
    MQSTS_OBJECTSTRING_VSCCSID    DS    F  CCSID of variable length string
    MQSTS_OBJECTSTRING_LENGTH     EQU   *-MQSTS_OBJECTSTRING
                                  ORG   MQSTS_OBJECTSTRING
    MQSTS_OBJECTSTRING_AREA       DS    CL(MQSTS_OBJECTSTRING_LENGTH)
    *
    MQSTS_SUBNAME                 DS    0F Force fullword alignment
    MQSTS_SUBNAME_VSPTR           DS    A  Address of variable length string
    MQSTS_SUBNAME_VSOFFSET        DS    F  Offset of variable length string
    MQSTS_SUBNAME_VSBUFSIZE       DS    F  Size of buffer
    MQSTS_SUBNAME_VSLENGTH        DS    F  Length of variable length string
    MQSTS_SUBNAME_VSCCSID         DS    F  CCSID of variable length string
    MQSTS_SUBNAME_LENGTH          EQ    *-MQSTS_SUBNAME
                                  ORG   MQSTS_SUBNAME
    MQSTS_SUBNAME_AREA            DS    CL(MQSTS_SUBNAME_LENGTH)
    *
    MQSTS_OPENOPTIONS             DS    F  Failing open options
    MQSTS_SUBOPTIONS              DS    F  Failing subscription option
    MQSTS_LENGTH                  EQU   *-MQSTS
                                  ORG   MQSTS
    MQSTS_AREA              DS    CL(MQSTS_LENGTH)
```

MQTM-Zpráva spouštěče

Následující tabulka shrnuje pole ve struktuře.

Tabulka 553. Pole v MQTM

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
QName	Název spuštěné fronty	QName
ProcessName	Název objektu procesu	ProcessName
TriggerData	Data spouątěče	TriggerData
ApplType	Typ aplikace	ApplType
ApplId	Identifikátor aplikace	ApplId
EnvData	Data prostředí	EnvData
UserData	Data uživatele	UserData

Přehled pro MQTM

Účel: Struktura MQTM popisuje data ve zprávě spouštěče, která je odeslána správcem front do aplikace monitoru spouštěčů, když se vyskytne událost spouštěče pro frontu.

Tato struktura je součástí produktu WebSphere MQ Trigger Monitor Interface (TMI), který je jedním z rozhraní rámce produktu WebSphere MQ .

Název formátu: MQFMT_TRIGGER.

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 *CodedCharSetId* a *Encoding* 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 vyžadovat předání některých nebo všech informací ve zprávě spouštěče do aplikace, kterou spouští aplikace monitoru spouštěčů. Informace, které mohou být potřebné pro spuštěnou aplikaci, zahrnují *QName*, *TriggerData*a *UserData*. 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)" na stránce 564.

- V systému z/OSpro aplikaci MQAT_CICS, která je spuštěna s použitím transakce CKTI, je veškerá struktura zprávy spouštěče MQTM zpřístupněna pro spuštěnou transakci; informace lze načíst pomocí příkazu EXEC CICS RETRIEVE.
- V produktu IBM ipředává aplikace monitoru spouštěčů (trigger-monitor) s produktem WebSphere MQ strukturu MQTMC2 do spuštěné aplikace.

Informace o používání spouštěčů naleznete v tématu Spuštění aplikací produktu WebSphere MQ pomocí spouštěčů .

MQMD pro zprávu spouštěče: Pole v deskriptoru MQMD zprávy spouštěče generované správcem front jsou nastavena následujícím způsobem:

Pole v MQMD	Použitá hodnota
StrucId	ID_STRUKTURY MQM_STRUCT
Version	MQMD_VERSION_1
Report	MQRO_NONE
MsgType	MQM_DATAGRAM
Expiry	MQEI_UNLIMITED
Feedback	MQFB_NONE
Encoding	MQENC_NATIVE
CodedCharSetId	Atribut *CodedCharSetId* správce front
Format	SPOUŠTĚČ 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 případně pro agenta MCA (Message Channel Agent)
PutApplName	Prvních 28 bajtů názvu správce front
PutDate	Datum, kdy se odešle zpráva spouštěče
PutTime	Čas odeslání zprávy spouštěče

Pole v MQMD	Použitá hodnota
ApplOriginData	Mezery

Pro nastavení podobných hodnot se doporučuje použít aplikaci, která vygeneruje zprávu spouštěče, s výjimkou následujících:

- Pole *Priority* může být nastaveno na hodnotu MQPRI_PRIORITY_AS_Q_DEF (správce front to změní na výchozí prioritu pro inicializační frontu, když je zpráva vložena).
- Pole *ReplyToQMgr* může být nastaveno na prázdné místo (správce front to změní na název lokálního správce front, když je zpráva vložena).
- Nastavte pole kontextu jako odpovídající pro aplikaci.

Pole pro MQTM

Struktura MQTM obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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 820 . Obsah těchto dat nemá význam pro správce front.

Význam *ApplId* je určen aplikací pro monitor spouštěčů. Monitor spouštěčů poskytovaný produktem WebSphere MQ vyžaduje, aby byl produktem *ApplId* název spustitelného programu. Níže uvedené poznámky se vztahují na uvedená prostředí:

- V systému z/OSje *ApplId* :
  - Identifikátor transakce systému CICS pro aplikace spuštěné pomocí transakce monitoru 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/.
- Na systémech UNIX může 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.

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 820 . 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
Transakce CICS .

MOST MQAT_CICS_BRIDGE
Aplikace mostu CICS .

MQAT_CICS_VSE
TransakceCICS/VSE .

MQAT_DOS
Aplikace klienta WebSphere MQ MQI v systému PC DOS.

MQAT_IMS
Aplikace IMS .

MOST MQAT_IMS_BRIDGE
Aplikace mostu IMS .

MQAT_JAVA
Aplikace Java.

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_NOTES_AGENT
Aplikace agenta Lotus Notes .

MQAT_NSK
HP Integrity NonStop Server .

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_RRS_BATCH
Dávková aplikace RRS.

MQAT_UNIX
Aplikace UNIX .

MQAT_UNKNOWN
Aplikace neznámého typu.

UŽIVATEL MQAT_USER
Uživatelsky definovaný typ aplikace.

MQAT_VOS
Aplikace Stratus VOS.

MQAT_WINDOWS
16bitová aplikace systému Windows .

POČ MQAT_WINDOWS_NT
32bitovou aplikaci systému Windows .

MQAT_WLM
Aplikace správce pracovní zátěže systému z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Aplikace 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.

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 820 . Obsah těchto dat nemá význam pro správce front.

V systému z/OSpro aplikaci CICS spuštěnou pomocí transakce CKTI nebo pomocí aplikace 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.

ProcessName (MQCHAR48)

Jedná se o název objektu procesu správce front zadaného 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 787 .

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.

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 787 .

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.

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.

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 787 . Obsah těchto dat nemá význam pro správce front.

V systému z/OS, pro aplikaci CICS spuštěnou pomocí transakce CKTI, se tyto informace nepoužijí.

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.

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 820 . Obsah těchto dat nemá význam pro správce front.

V systému Microsoft Windowsnesmí znakový řetězec obsahovat uvozovky, pokud se definice procesu předá do 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.

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.

Počáteční hodnoty a deklarace jazyka pro MQTM

Tabulka 554. Počáteční hodnoty polí v produktu MQTM pro MQTM

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQTM_STRUCT	'TM¬¬'
Version	MQTM_VERSION_1	1
QName	Není	Nulový řetězec nebo prázdné znaky
ProcessName	Není	Nulový řetězec nebo prázdné znaky
TriggerData	Není	Nulový řetězec nebo prázdné znaky
ApplType	Není	0
ApplId	Není	Nulový řetězec nebo prázdné znaky
EnvData	Není	Nulový řetězec nebo prázdné znaky
UserData	Není	Nulový řetězec nebo prázdné znaky

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.
3. V programovacím jazyce C-proměnná makraObjekt MQTM_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQTM MyTM = {MQTM_DEFAULT};
```

Deklarace C

```
typedef struct tagMQTM MQTM;
struct tagMQTM {
  MQCHAR4    StrucId;      /* Structure identifier */
  MQLONG     Version;      /* Structure version number */
  MQCHAR48   QName;        /* Name of triggered queue */
  MQCHAR48   ProcessName;  /* Name of process object */
  MQCHAR64   TriggerData;  /* Trigger data */
  MQLONG     ApplType;     /* Application type */
  MQCHAR256  ApplId;       /* Application identifier */
  MQCHAR128  EnvData;      /* Environment data */
  MQCHAR128  UserData;     /* User data */
};
```

Deklarace COBOL

```
**   MQTM structure
  10 MQTM.
**    Structure identifier
   15 MQTM-STRUCID     PIC X(4).
**    Structure version number
   15 MQTM-VERSION     PIC S9(9) BINARY.
**    Name of triggered queue
   15 MQTM-QNAME       PIC X(48).
**    Name of process object
   15 MQTM-PROCESSNAME PIC X(48).
**    Trigger data
   15 MQTM-TRIGGERDATA PIC X(64).
**    Application type
   15 MQTM-APPLTYPE    PIC S9(9) BINARY.
**    Application identifier
   15 MQTM-APPLID      PIC X(256).
**    Environment data
   15 MQTM-ENVDATA     PIC X(128).
**    User data
   15 MQTM-USERDATA    PIC X(128).
```

Deklarace PL/I

```
dcl
 1 MQTM based,
   3 StrucId     char(4),       /* Structure identifier */
   3 Version     fixed bin(31), /* Structure version number */
   3 QName       char(48),      /* Name of triggered queue */
   3 ProcessName char(48),      /* Name of process object */
   3 TriggerData char(64),      /* Trigger data */
   3 ApplType    fixed bin(31), /* Application type */
   3 ApplId      char(256),     /* Application identifier */
   3 EnvData     char(128),     /* Environment data */
   3 UserData    char(128);     /* User data */
```

Deklarace High Level Assembler

```
MQTM              DSECT
MQTM_STRUCID      DS   CL4    Structure identifier
MQTM_VERSION      DS   F      Structure version number
MQTM_QNAME        DS   CL48   Name of triggered queue
MQTM_PROCESSNAME  DS   CL48   Name of process object
MQTM_TRIGGERDATA  DS   CL64   Trigger data
MQTM_APPLTYPE     DS   F      Application type
MQTM_APPLID       DS   CL256  Application identifier
MQTM_ENVDATA      DS   CL128  Environment data
MQTM_USERDATA     DS   CL128  User data
*
MQTM_LENGTH       EQU  *-MQTM
                  ORG  MQTM
MQTM_AREA         DS   CL(MQTM_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQTM
  StrucId     As String*4   'Structure identifier'
  Version     As Long       'Structure version number'
  QName       As String*48  'Name of triggered queue'
  ProcessName As String*48  'Name of process object'
  TriggerData As String*64  'Trigger data'
  ApplType    As Long       'Application type'
  ApplId      As String*256 'Application identifier'
  EnvData     As String*128 'Environment data'
  UserData    As String*128 'User data'
End Type
```

MQTMC2 -Spouštěcí zpráva 2 (znakový formát)

Následující tabulka shrnuje pole ve struktuře.

Tabulka 555. Pole v MQTMC2

Pole	Popis	Téma
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
QName	Název spuštěné fronty	QName
ProcessName	Název objektu procesu	ProcessName
TriggerData	Data spouątěče	TriggerData
ApplType	Typ aplikace	ApplType
ApplId	Identifikátor aplikace	ApplId
EnvData	Data prostředí	EnvData
UserData	Data uživatele	UserData
QMgrName	Název správce front	QMgrName

Přehled pro MQTMC2

Účel: Když aplikace monitoru spouštěčů načte zprávu spouštěče (MQTM) z inicializační fronty, může být nutné spustit monitor spouštěčů některé nebo všechny informace ve zprávě spouštěče do aplikace, kterou spouští monitor spouštěčů.

Informace, které může spuštěná aplikace potřebovat, zahrnují *QName*, *TriggerData* a *UserData*. 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 WebSphere MQ Trigger Monitor Interface (TMI), který je jedním z rozhraní rámce produktu WebSphere MQ .

Znaková sada a kódování: Znaková data v souboru MQTMC2 jsou ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front *CodedCharSetId* .

Použití: Struktura MQTMC2 je velmi podobná formátu 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 z/OSpro aplikaci MQAT_IMS, která je spuštěna pomocí aplikace CSQQTRMN, je struktura MQTMC2 zpřístupněna pro spuštěnou aplikaci.
- V systému IBM ipředává aplikace monitoru spouštěčů, která je součástí produktu WebSphere MQ , strukturu MQTMC2 do spuštěné aplikace.

Pole pro MQTMC2

Struktura MQTMC2 obsahuje následující pole; tato pole jsou popsána v **abecedním pořadí**:

ApplId (MQCHAR256)

Identifikátor aplikace.

Viz pole *ApplId* 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.

EnvData (MQCHAR128)
Data prostředí.

Viz pole *EnvData* ve struktuře MQTM.

ProcessName (MQCHAR48)
Název objektu procesu.

Viz pole *ProcessName* ve struktuře MQTM.

QMgrName (MQCHAR48)
Název správce front.

Jedná se o název správce front, v němž došlo k události spouštěče.

QName (MQCHAR48)
Název spuštěné fronty.

Viz pole *QName* ve struktuře MQTM.

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.

TriggerData (MQCHAR64)
Data spouątěče.

Viz pole *TriggerData* ve struktuře MQTM.

UserData (MQCHAR128)
Uživatelská data.

Viz pole *UserData* ve struktuře MQTM.

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).

Počáteční hodnoty a deklarace jazyka pro MQTMC2

Tabulka 556. Počáteční hodnoty polí v MQTMC2 pro MQTMC2

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID_STRUKTURY MQTC_STRUCT	'TMC¬'
Version	MQTMC_VERSION_2	'¬¬¬2'
QName	Není	Nulový řetězec nebo prázdné znaky
ProcessName	Není	Nulový řetězec nebo prázdné znaky
TriggerData	Není	Nulový řetězec nebo prázdné znaky
ApplType	Není	Mezery
ApplId	Není	Nulový řetězec nebo prázdné znaky
EnvData	Není	Nulový řetězec nebo prázdné znaky
UserData	Není	Nulový řetězec nebo prázdné znaky
QMgrName	Není	Nulový řetězec nebo prázdné znaky

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.
3. V programovacím jazyce C-proměnná makraHodnota MQTMC2_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQTMC2 MyTMC = {MQTMC2_DEFAULT};
```

Deklarace C

```
typedef struct tagMQTMC2 MQTMC2;
struct tagMQTMC2 {
  MQCHAR4    StrucId;      /* Structure identifier */
  MQCHAR4    Version;      /* Structure version number */
  MQCHAR48   QName;        /* Name of triggered queue */
  MQCHAR48   ProcessName;  /* Name of process object */
  MQCHAR64   TriggerData;  /* Trigger data */
  MQCHAR4    ApplType;     /* Application type */
  MQCHAR256  ApplId;       /* Application identifier */
```

```
  MQCHAR128  EnvData;      /* Environment data */
  MQCHAR128  UserData;     /* User data */
  MQCHAR48   QMgrName;     /* Queue manager name */
};
```

Deklarace COBOL

```
**   MQTMC2 structure
  10 MQTMC2.
**    Structure identifier
   15 MQTMC2-STRUCID     PIC X(4).
**    Structure version number
   15 MQTMC2-VERSION     PIC X(4).
**    Name of triggered queue
   15 MQTMC2-QNAME       PIC X(48).
**    Name of process object
   15 MQTMC2-PROCESSNAME PIC X(48).
**    Trigger data
   15 MQTMC2-TRIGGERDATA PIC X(64).
**    Application type
   15 MQTMC2-APPLTYPE    PIC X(4).
**    Application identifier
   15 MQTMC2-APPLID      PIC X(256).
**    Environment data
   15 MQTMC2-ENVDATA     PIC X(128).
**    User data
   15 MQTMC2-USERDATA    PIC X(128).
**    Queue manager name
   15 MQTMC2-QMGRNAME    PIC X(48).
```

Deklarace PL/I

```
dcl
 1 MQTMC2 based,
  3 StrucId     char(4),   /* Structure identifier */
  3 Version     char(4),   /* Structure version number */
  3 QName       char(48),  /* Name of triggered queue */
  3 ProcessName char(48),  /* Name of process object */
  3 TriggerData char(64),  /* Trigger data */
  3 ApplType    char(4),   /* Application type */
  3 ApplId      char(256), /* Application identifier */
  3 EnvData     char(128), /* Environment data */
  3 UserData    char(128), /* User data */
  3 QMgrName    char(48);  /* Queue manager name */
```

Deklarace High Level Assembler

```
MQTMC              DSECT
MQTMC_STRUCID      DS   CL4    Structure identifier
MQTMC_VERSION      DS   CL4    Structure version number
MQTMC_QNAME        DS   CL48   Name of triggered queue
MQTMC_PROCESSNAME  DS   CL48   Name of process object
MQTMC_TRIGGERDATA  DS   CL64   Trigger data
MQTMC_APPLTYPE     DS   CL4    Application type
MQTMC_APPLID       DS   CL256  Application identifier
MQTMC_ENVDATA      DS   CL128  Environment data
MQTMC_USERDATA     DS   CL128  User data
MQTMC_QMGRNAME     DS   CL48   Queue manager name
*
MQTMC_LENGTH       EQU  *-MQTMC
                   ORG  MQTMC
MQTMC_AREA         DS   CL(MQTMC_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQTMC2
  StrucId     As String*4   'Structure identifier'
  Version     As String*4   'Structure version number'
  QName       As String*48  'Name of triggered queue'
  ProcessName As String*48  'Name of process object'
  TriggerData As String*64  'Trigger data'
```

```
   ApplType    As String*4   'Application type'
   ApplId      As String*256 'Application identifier'
   EnvData     As String*128 'Environment data'
   UserData    As String*128 'User data'
   QMgrName    As String*48  'Queue manager name'
End Type
```

MQWIH-Záhlaví informací o práci

Následující tabulka shrnuje pole ve struktuře.

Tabulka 557. Pole v MQWIH		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
StrucLength	Délka struktury MQWIH	StrucLength
Encoding	Číselné kódování dat, která následuje za MQWIH	Kódování
CodedCharSetId	Identifikátor znakové sady dat, která následuje za MQWIH	CodedCharSetId
Format	Název formátu dat, který následuje za MQWIH	Formát
Flags	Příznaky	Příznaky
ServiceName	Název služby	ServiceName
ServiceStep	Název kroku služby	ServiceStep
MsgToken	Token zpráv	MsgToken
Reserved	Vyhrazené	Vyhrazené

Přehled pro MQWIH

Dostupnost: Všechny systémy WebSphere MQ a klienti produktu WebSphere MQ , kteří jsou připojeni k těmto systémům.

Účel: Struktura MQWIH popisuje informace, které musí být přítomny na začátku zprávy, kterou má zpracovat správce pracovní zátěže z/OS .

Název formátu: MQFMT_WORK_INFO_HEADER.

Znaková sada a kódování: Pole ve struktuře MQWIH jsou ve znakové sadě a kódování dána 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, které jsou platné v názvech front.

Použití: Je-li zpráva zpracovávána správcem pracovní zátěže z/OS , musí zpráva začínat strukturou MQWIH.

Pole pro MQWIH

Struktura MQWIH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

Příznaky (MQLONG)

Hodnota musí být:

MQWIH_NONE
Žádné vlajky.

Počáteční hodnota tohoto pole je MQWIH_NONE.

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.

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.

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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQWIH

Tabulka 558. Počáteční hodnoty polí v MQWIH pro MQWIH

Název pole	**Název konstanty**	**Hodnota konstanty**
StrucId	ID_STRUKTURY MQWIH_	'WIH¬'
Version	MQWIH_VERSION_1	1
StrucLength	MQWIH_LENGTH_1	120
Encoding	Není	0
CodedCharSetId	MQCSI_UNDEFINED	0
Format	MQFMT_NONE	Mezery
Flags	MQWIH_NONE	0
ServiceName	Není	Mezery
ServiceStep	Není	Mezery
MsgToken	MQMTOK_NONE	Hodnoty null
Reserved	Není	Mezery

Tabulka 558. Počáteční hodnoty polí v MQWIH pro MQWIH (pokračování)		
Název pole	**Název konstanty**	**Hodnota konstanty**

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. V programovacím jazyce C-proměnná makraHodnota MQWIH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQWIH MyWIH = {MQWIH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQWIH MQWIH;
struct tagMQWIH {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQLONG    StrucLength;     /* Length of MQWIH structure */
  MQLONG    Encoding;        /* Numeric encoding of data that follows
                                MQWIH */
  MQLONG    CodedCharSetId;  /* Character-set identifier of data that
                                follows MQWIH */
  MQCHAR8   Format;          /* Format name of data that follows
                                MQWIH */
  MQLONG    Flags;           /* Flags */
  MQCHAR32  ServiceName;     /* Service name */
  MQCHAR8   ServiceStep;     /* Service step name */
  MQBYTE16  MsgToken;        /* Message token */
  MQCHAR32  Reserved;        /* Reserved */
};
```

Deklarace COBOL

```
**   MQWIH structure
  10 MQWIH.
**    Structure identifier
   15 MQWIH-STRUCID        PIC X(4).
**    Structure version number
   15 MQWIH-VERSION        PIC S9(9) BINARY.
**    Length of MQWIH structure
   15 MQWIH-STRUCLENGTH    PIC S9(9) BINARY.
**    Numeric encoding of data that follows MQWIH
   15 MQWIH-ENCODING       PIC S9(9) BINARY.
**    Character-set identifier of data that follows MQWIH
   15 MQWIH-CODEDCHARSETID PIC S9(9) BINARY.
**    Format name of data that follows MQWIH
   15 MQWIH-FORMAT         PIC X(8).
**    Flags
   15 MQWIH-FLAGS          PIC S9(9) BINARY.
**    Service name
   15 MQWIH-SERVICENAME    PIC X(32).
**    Service step name
   15 MQWIH-SERVICESTEP    PIC X(8).
**    Message token
   15 MQWIH-MSGTOKEN       PIC X(16).
**    Reserved
   15 MQWIH-RESERVED       PIC X(32).
```

Deklarace PL/I

```
dcl
 1 MQWIH based,
   3 StrucId        char(4),       /* Structure identifier */
   3 Version        fixed bin(31), /* Structure version number */
   3 StrucLength    fixed bin(31), /* Length of MQWIH structure */
   3 Encoding       fixed bin(31), /* Numeric encoding of data that
                                      follows MQWIH */
```

```
  3 CodedCharSetId fixed bin(31), /* Character-set identifier of data
                                     that follows MQWIH */
  3 Format         char(8),       /* Format name of data that follows
                                     MQWIH */
  3 Flags          fixed bin(31), /* Flags */
  3 ServiceName    char(32),      /* Service name */
  3 ServiceStep    char(8),       /* Service step name */
  3 MsgToken       char(16),      /* Message token */
  3 Reserved       char(32);      /* Reserved */
```

Deklarace High Level Assembler

```
MQWIH                DSECT
MQWIH_STRUCID        DS   CL4   Structure identifier
MQWIH_VERSION        DS   F     Structure version number
MQWIH_STRUCLENGTH    DS   F     Length of MQWIH structure
MQWIH_ENCODING       DS   F     Numeric encoding of data that follows
*                               MQWIH
MQWIH_CODEDCHARSETID DS   F     Character-set identifier of data that
*                               follows MQWIH
MQWIH_FORMAT         DS   CL8   Format name of data that follows MQWIH
MQWIH_FLAGS          DS   F     Flags
MQWIH_SERVICENAME    DS   CL32  Service name
MQWIH_SERVICESTEP    DS   CL8   Service step name
MQWIH_MSGTOKEN       DS   XL16  Message token
MQWIH_RESERVED       DS   CL32  Reserved
*
MQWIH_LENGTH         EQU  *-MQWIH
                     ORG  MQWIH
MQWIH_AREA           DS   CL(MQWIH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQWIH
  StrucId        As String*4  'Structure identifier'
  Version        As Long      'Structure version number'
  StrucLength    As Long      'Length of MQWIH structure'
  Encoding       As Long      'Numeric encoding of data that follows'
                              'MQWIH'
  CodedCharSetId As Long      'Character-set identifier of data that'
                              'follows MQWIH'
  Format         As String*8  'Format name of data that follows MQWIH'
  Flags          As Long      'Flags'
  ServiceName    As String*32 'Service name'
  ServiceStep    As String*8  'Service step name'
  MsgToken       As MQBYTE16  'Message token'
  Reserved       As String*32 'Reserved'
End Type
```

MQXP-Blok parametrů ukončení

Následující tabulka shrnuje pole ve struktuře.

Tabulka 559. Pole v MQXP		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
ExitId	Identifikátor ukončení	ExitId
ExitReason	Důvod vyvolání uživatelské procedury	ExitReason
ExitResponse	Odezva z uživatelské procedury	ExitResponse
ExitCommand	Kód volání rozhraní API	ExitCommand
ExitParmCount	Počet parametrů	ExitParmPočet

Tabulka 559. Pole v MQXP (pokračování)		
Pole	**Popis**	**Téma**
ExitUserArea	Uživatelská oblast	OblastExitUser

Přehled pro MQXP

Dostupnost: z/OS.

Účel: Struktura MQXP se používá jako vstupní/výstupní parametr pro ukončení rozhraní API. Další informace o této uživatelské proceduře naleznete v tématu Uživatelská procedura přejezdu rozhraní API.

Znaková sada a kódování: Znaková data v aplikaci MQXP jsou ve znakové sadě lokálního správce front; tento údaj je dán atributem správce front *CodedCharSetId* . Numerická data ve struktuře MQXP jsou v nativním kódování počítače. Tato hodnota je dána rozhraním MQENC_NATIVE.

Pole pro MQXP

Struktura MQXP obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

ExitId (MQLONG)

Toto je nastaveno na vstupu do uživatelské procedury a označuje typ uživatelské procedury:

UŽIVATELSKÁ PROCEDURA MQXT_API_CROSSING_EXIT
Uživatelská procedura rozhraní API pro CICS.

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. Patří mezi ně:

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.

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.

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.

Rezervováno (MQLONG)

Jedná se o vyhrazené pole. Jeho hodnota není významná pro ukončení.

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.

Deklarace jazyka

Tato struktura je podporována v následujících programovacích jazycích.

Deklarace C

```
typedef struct tagMQXP MQXP;
struct tagMQXP {
```

```
  MQCHAR4   StrucId;        /* Structure identifier */
  MQLONG    Version;        /* Structure version number */
  MQLONG    ExitId;         /* Exit identifier */
  MQLONG    ExitReason;     /* Reason for invocation of exit */
  MQLONG    ExitResponse;   /* Response from exit */
  MQLONG    ExitCommand;    /* API call code */
  MQLONG    ExitParmCount;  /* Parameter count */
  MQLONG    Reserved;       /* Reserved */
  MQBYTE16  ExitUserArea;   /* User area */
};
```

Deklarace COBOL

```
**   MQXP structure
  10 MQXP.
**    Structure identifier
   15 MQXP-STRUCID       PIC X(4).
**    Structure version number
   15 MQXP-VERSION       PIC S9(9) BINARY.
**    Exit identifier
   15 MQXP-EXITID        PIC S9(9) BINARY.
**    Reason for invocation of exit
   15 MQXP-EXITREASON    PIC S9(9) BINARY.
**    Response from exit
   15 MQXP-EXITRESPONSE  PIC S9(9) BINARY.
**    API call code
   15 MQXP-EXITCOMMAND   PIC S9(9) BINARY.
**    Parameter count
   15 MQXP-EXITPARMCOUNT PIC S9(9) BINARY.
**    Reserved
   15 MQXP-RESERVED      PIC S9(9) BINARY.
**    User area
   15 MQXP-EXITUSERAREA  PIC X(16).
```

Deklarace PL/I

```
dcl
 1 MQXP based,
  3 StrucId       char(4),       /* Structure identifier */
  3 Version       fixed bin(31), /* Structure version number */
  3 ExitId        fixed bin(31), /* Exit identifier */
  3 ExitReason    fixed bin(31), /* Reason for invocation of exit */
  3 ExitResponse  fixed bin(31), /* Response from exit */
  3 ExitCommand   fixed bin(31), /* API call code */
  3 ExitParmCount fixed bin(31), /* Parameter count */
  3 Reserved      fixed bin(31), /* Reserved */
  3 ExitUserArea  char(16);      /* User area */
```

Deklarace High Level Assembler

```
MQXP                DSECT
MQXP_STRUCID        DS   CL4   Structure identifier
MQXP_VERSION        DS   F     Structure version number
MQXP_EXITID         DS   F     Exit identifier
MQXP_EXITREASON     DS   F     Reason for invocation of exit
MQXP_EXITRESPONSE   DS   F     Response from exit
MQXP_EXITCOMMAND    DS   F     API call code
MQXP_EXITPARMCOUNT  DS   F     Parameter count
MQXP_RESERVED       DS   F     Reserved
MQXP_EXITUSERAREA   DS   XL16  User area
*
MQXP_LENGTH         EQU  *-MQXP
                    ORG  MQXP
MQXP_AREA           DS   CL(MQXP_LENGTH)
```

MQXQH-záhlaví přenosové fronty

Následující tabulka shrnuje pole ve struktuře.

Tabulka 560. Pole v MQXQH		
Pole	**Popis**	**Téma**
StrucId	Identifikátor struktury	StrucId
Version	Číslo verze struktury	verze
RemoteQName	Název cílové fronty	RemoteQName
RemoteQMgrName	Název správce cílové fronty	RemoteQmgrName
MsgDesc	Deskriptor původní zprávy	MsgDesc

Přehled pro MQXQH

Dostupnost: Všechny systémy WebSphere MQ a klienti WebSphere MQ .

Účel: Struktura MQXQH popisuje informace, které jsou uvedeny předponou zprávy zpráv aplikace při jejich přenosu do přenosových front. Přenosová fronta je speciální typ lokální fronty, která dočasně uchovává zprávy určené pro vzdálené fronty (tj. určené pro fronty, které nenáleží do lokálního správce front). Přenosová fronta je označena atributem fronty *Usage* , který má hodnotu MQUS_TRANSMISSION.

Název formátu: MQFMT_XMIT_Q_HEADER.

Znaková sada a kódování: Data v MQXOH musí být ve znakové sadě poskytnuté atributem správce front *CodedCharSetId* a kódováním lokálního správce front uvedeného MQENC_NATIVE.

Nastavte znakovou sadu a kódování MQXQH do polí *CodedCharSetId* a *Encoding* 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 *MsgDesc* 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 *MsgDesc* v rámci volání MQGET při odebrání zprávy z fronty konečného cíle.

Pole v odděleném deskriptoru zpráv: Pole v samostatném deskriptoru zpráv jsou nastavena správcem front, jak je zobrazeno. Pokud správce front nepodporuje MQMD version-2 , použije se MQMD version-1 bez ztráty funkce.

Pole v samostatném deskriptoru MQMD	**Použitá hodnota**
StrucId	ID_STRUKTURY MQM_STRUCT

Pole v samostatném deskriptoru MQMD	Použitá hodnota
Version	MQMD_VERSION_2
Report	Zkopírováno z vloženého deskriptoru zpráv, ale s bity identifikovanými hodnotou MQRO_ACCEPT_UNSUP_IF_XMIT_MASK nastavenou na nulu. (To zabraňuje generování zprávy COA nebo CHSK, když je zpráva vložena nebo odebrána z přenosové fronty.)
MsgType	Zkopírováno z vloženého deskriptoru zpráv.
Expiry	Zkopírováno z vloženého deskriptoru zpráv.
Feedback	Zkopírováno z vloženého deskriptoru zpráv.
Encoding	MQENC_NATIVE (viz poznámka)
CodedCharSetId	Atribut *CodedCharSetId* správce front.
Format	ZÁHLAVÍ MQFMT_XMIT_Q_HEADER
Priority	Zkopírováno z vloženého deskriptoru zpráv.
Persistence	Zkopírováno z vloženého deskriptoru zpráv.
MsgId	Nová hodnota je generována správcem front. Tento identifikátor zprávy se liší od identifikátoru *MsgId* , který správce front mohl vygenerovat pro deskriptor vloženého zprávy (viz výše).
CorrelId	*MsgId* z deskriptoru vložených zpráv. Pro zprávy vkládané do systému SYSTEM.CLUSTER.TRANSMIT.QUEUE, *CorrelId* je vyhrazena pro vnitřní použití.
BackoutCount	0
ReplyToQ	Zkopírováno z vloženého deskriptoru zpráv.
ReplyToQMgr	Zkopírováno z vloženého deskriptoru zpráv.
UserIdentifier	Zkopírováno z vloženého deskriptoru zpráv.
AccountingToken	Zkopírováno z vloženého deskriptoru zpráv. Pro zprávy vkládané do systému SYSTEM.CLUSTER.TRANSMIT.QUEUE, *AccountingToken* je vyhrazena pro vnitřní použití.
ApplIdentityData	Zkopírováno z vloženého deskriptoru zpráv.
PutApplType	MQAT_QMGR
PutApplName	Prvních 28 bajtů názvu správce front.
PutDate	Datum, kdy byla zpráva vložena do přenosové fronty.
PutTime	Čas, kdy byla zpráva vložena do přenosové fronty.
ApplOriginData	Mezery
GroupId	MQGI_NONE
MsgSeqNumber	1
Offset	0
MsgFlags	MQMF_NONE
OriginalLength	MQOL_UNDEFINED

- V systému Windowsse hodnota MQENC_NATIVE for Micro Focus COBOL liší od hodnoty pro C. Hodnota v poli *Encoding* v odděleném deskriptoru zpráv je vždy hodnota pro C v těchto prostředích; tato

hodnota je 546 v desítkovém zápisu. Také celočíselné pole ve struktuře MQXQH se nacházejí v kódování, které odpovídá této hodnotě (nativní kódování Intel).

Pole v deskriptoru vloženého zpráv: Pole v deskriptoru vloženého zprávy mají stejné hodnoty jako pole v parametru *MsgDesc* volání MQPUT nebo MQPUT1 , s výjimkou následujících:

- Pole *Version* má vždy hodnotu MQMD_VERSION_1.
- Má-li pole *Priority* hodnotu MQPRI_PRIORITY_AS_Q_DEF, je nahrazena hodnotou atributu *DefPriority* fronty.
- Má-li pole *Persistence* hodnotu MQPER_PERSISTENCE_AS_Q_DEF, je nahrazena hodnotou atributu *DefPersistence* fronty.
- Pokud má pole *MsgId* hodnotu MQMI_NONE, nebo byla zadána volba MQPMO_NEW_MSG_ID nebo zpráva je zpráva rozdělovníku, *MsgId* 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 *MsgId* v každém z nových deskriptorů vložených zpráv stejné jako v původní zprávě distribučního seznamu.
- Pokud byla zadána volba MQPMO_NEW_CORREL_ID, je hodnota *CorrelId* nahrazena novým identifikátorem korelace generovaným správcem front.
- Pole kontextu jsou nastavena tak, jak je označeno volbami MQPMO_ * _CONTEXT zadané v parametru *PutMsgOpts* ; jsou to pole kontextu:
  - *AccountingToken*
  - *ApplIdentityData*
  - *ApplOriginData*
  - *PutApplName*
  - *PutApplType*
  - *PutDate*
  - *PutTime*
  - *UserIdentifier*
- 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: Aplikace může také vložit 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 pole *Format* v parametru *MsgDesc* volání MQPUT nebo MQPUT1 obsahovat hodnotu MQFMT_XMIT_Q_HEADER.

Znaková data ve struktuře MQXQH vytvořená aplikací musí být ve znakové sadě lokálního správce front (definované atributem správce front *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.

Správce front však nekontroluje, zda je přítomna struktura MQXQH, nebo že pro pole byly zadány platné hodnoty.

Aplikace by neměly vkládat své zprávy přímo do systému SYSTEM.CLUSTER.TRANSMIT.QUEUE.

Získávání zpráv z přenosových front: Aplikace, které získávají zprávy z přenosové fronty, musí zpracovávat informace ve struktuře MQXQH vhodným způsobem. Přítomnost struktury MQXQH na začátku dat zprávy aplikace je označena hodnotou MQFMT_XMIT_Q_HEADER, která je vrácena v poli *Format* v parametru *MsgDesc* volání MQGET. Hodnoty vrácené v polích *CodedCharSetId* a *Encoding* v parametru *MsgDesc* udávají znakovou sadu a kódování znakových a celočíselných dat ve struktuře MQXQH. Znaková sada a kódování dat zprávy aplikace jsou definovány v polích *CodedCharSetId* a *Encoding* v deskriptoru vložených zpráv.

Pole pro MQXQH

Struktura MQXQH obsahuje následující pole; pole jsou popsána v **abecedním pořadí**:

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.

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.

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.

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.

Počáteční hodnoty a deklarace jazyka pro MQXQH

Tabulka 561. Počáteční hodnoty polí v MQXQH pro MQXQH

Název pole	Název konstanty	Hodnota konstanty
StrucId	ID STRUKTURY MQXQ_STRUCTION_ID	'XQH¬'
Version	MQXQH_VERSION_1	1
RemoteQName	Není	Nulový řetězec nebo prázdné znaky
RemoteQMgrName	Není	Nulový řetězec nebo prázdné znaky
MsgDesc	Stejné názvy a hodnoty jako MQMD; viz Tabulka 515 na stránce 427	-

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.
3. V programovacím jazyce C-proměnná makraHodnota MQXQH_DEFAULT obsahuje výše uvedené hodnoty. Použijte ji následujícím způsobem, abyste poskytli počáteční hodnoty pro pole ve struktuře:

```
MQXQH MyXQH = {MQXQH_DEFAULT};
```

Deklarace C

```
typedef struct tagMQXQH MQXQH;
struct tagMQXQH {
  MQCHAR4   StrucId;         /* Structure identifier */
  MQLONG    Version;         /* Structure version number */
  MQCHAR48  RemoteQName;     /* Name of destination queue */
  MQCHAR48  RemoteQMgrName;  /* Name of destination queue manager */
  MQMD1     MsgDesc;         /* Original message descriptor */
};
```

Deklarace COBOL

```
**   MQXQH structure
  10 MQXQH.
**    Structure identifier
   15 MQXQH-STRUCID                 PIC X(4).
**    Structure version number
   15 MQXQH-VERSION                 PIC S9(9) BINARY.
**    Name of destination queue
   15 MQXQH-REMOTEQNAME             PIC X(48).
**    Name of destination queue manager
   15 MQXQH-REMOTEQMGRNAME          PIC X(48).
**    Original message descriptor
   15 MQXQH-MSGDESC.
**     Structure identifier
    20 MQXQH-MSGDESC-STRUCID         PIC X(4).
**     Structure version number
    20 MQXQH-MSGDESC-VERSION         PIC S9(9) BINARY.
**     Report options
    20 MQXQH-MSGDESC-REPORT          PIC S9(9) BINARY.
**     Message type
    20 MQXQH-MSGDESC-MSGTYPE         PIC S9(9) BINARY.
**     Expiry time
    20 MQXQH-MSGDESC-EXPIRY          PIC S9(9) BINARY.
```

```
**     Feedback or reason code
    20 MQXQH-MSGDESC-FEEDBACK         PIC S9(9) BINARY.
**     Numeric encoding of message data
    20 MQXQH-MSGDESC-ENCODING         PIC S9(9) BINARY.
**     Character set identifier of message data
    20 MQXQH-MSGDESC-CODEDCHARSETID   PIC S9(9) BINARY.
**     Format name of message data
    20 MQXQH-MSGDESC-FORMAT           PIC X(8).
**     Message priority
    20 MQXQH-MSGDESC-PRIORITY         PIC S9(9) BINARY.
**     Message persistence
    20 MQXQH-MSGDESC-PERSISTENCE      PIC S9(9) BINARY.
**     Message identifier
    20 MQXQH-MSGDESC-MSGID            PIC X(24).
**     Correlation identifier
    20 MQXQH-MSGDESC-CORRELID         PIC X(24).
**     Backout counter
    20 MQXQH-MSGDESC-BACKOUTCOUNT     PIC S9(9) BINARY.
**     Name of reply-to queue
    20 MQXQH-MSGDESC-REPLYTOQ         PIC X(48).
**     Name of reply queue manager
    20 MQXQH-MSGDESC-REPLYTOQMGR      PIC X(48).
**     User identifier
    20 MQXQH-MSGDESC-USERIDENTIFIER   PIC X(12).
**     Accounting token
    20 MQXQH-MSGDESC-ACCOUNTINGTOKEN  PIC X(32).
**     Application data relating to identity
    20 MQXQH-MSGDESC-APPLIDENTITYDATA PIC X(32).
**     Type of application that put the message
    20 MQXQH-MSGDESC-PUTAPPLTYPE      PIC S9(9) BINARY.
**     Name of application that put the message
    20 MQXQH-MSGDESC-PUTAPPLNAME      PIC X(28).
**     Date when message was put
    20 MQXQH-MSGDESC-PUTDATE          PIC X(8).
**     Time when message was put
    20 MQXQH-MSGDESC-PUTTIME          PIC X(8).
**     Application data relating to origin
    20 MQXQH-MSGDESC-APPLORIGINDATA   PIC X(4).
```

Deklarace PL/I

```
dcl
 1 MQXQH based,
  3 StrucId           char(4),       /* Structure identifier */
  3 Version           fixed bin(31), /* Structure version number */
  3 RemoteQName       char(48),      /* Name of destination queue */
  3 RemoteQMgrName    char(48),      /* Name of destination queue
                                        manager */
  3 MsgDesc,                         /* Original message descriptor */
   5 StrucId          char(4),       /* Structure identifier */
   5 Version          fixed bin(31), /* Structure version number */
   5 Report           fixed bin(31), /* Report options */
   5 MsgType          fixed bin(31), /* Message type */
   5 Expiry           fixed bin(31), /* Expiry time */
   5 Feedback         fixed bin(31), /* Feedback or reason code */
   5 Encoding         fixed bin(31), /* Numeric encoding of message
                                        data */
   5 CodedCharSetId   fixed bin(31), /* Character set identifier of
                                        message data */
   5 Format           char(8),       /* Format name of message data */
   5 Priority         fixed bin(31), /* Message priority */
   5 Persistence      fixed bin(31), /* Message persistence */
   5 MsgId            char(24),      /* Message identifier */
   5 CorrelId         char(24),      /* Correlation identifier */
   5 BackoutCount     fixed bin(31), /* Backout counter */
   5 ReplyToQ         char(48),      /* Name of reply-to queue */
   5 ReplyToQMgr      char(48),      /* Name of reply queue manager */
   5 UserIdentifier   char(12),      /* User identifier */
   5 AccountingToken  char(32),      /* Accounting token */
   5 ApplIdentityData char(32),      /* Application data relating to
                                        identity */
   5 PutApplType      fixed bin(31), /* Type of application that put the
                                        message */
   5 PutApplName      char(28),      /* Name of application that put the
                                        message */
   5 PutDate          char(8),       /* Date when message was put */
   5 PutTime          char(8),       /* Time when message was put */
```

```
   5 ApplOriginData   char(4);       /* Application data relating to
                                        origin */
```

Deklarace High Level Assembler

```
MQXQH                           DSECT
MQXQH_STRUCID                   DS   CL4   Structure identifier
MQXQH_VERSION                   DS   F     Structure version number
MQXQH_REMOTEQNAME               DS   CL48  Name of destination queue
MQXQH_REMOTEQMGRNAME            DS   CL48  Name of destination queue
*                                          manager
MQXQH_MSGDESC                   DS   0F    Force fullword alignment
MQXQH_MSGDESC_STRUCID           DS   CL4   Structure identifier
MQXQH_MSGDESC_VERSION           DS   F     Structure version number
MQXQH_MSGDESC_REPORT            DS   F     Report options
MQXQH_MSGDESC_MSGTYPE           DS   F     Message type
MQXQH_MSGDESC_EXPIRY            DS   F     Expiry time
MQXQH_MSGDESC_FEEDBACK          DS   F     Feedback or reason code
MQXQH_MSGDESC_ENCODING          DS   F     Numeric encoding of message
*                                          data
MQXQH_MSGDESC_CODEDCHARSETID    DS   F     Character set identifier of
*                                          message data
MQXQH_MSGDESC_FORMAT            DS   CL8   Format name of message data
MQXQH_MSGDESC_PRIORITY          DS   F     Message priority
MQXQH_MSGDESC_PERSISTENCE       DS   F     Message persistence
MQXQH_MSGDESC_MSGID             DS   XL24  Message identifier
MQXQH_MSGDESC_CORRELID          DS   XL24  Correlation identifier
MQXQH_MSGDESC_BACKOUTCOUNT      DS   F     Backout counter
MQXQH_MSGDESC_REPLYTOQ          DS   CL48  Name of reply-to queue
MQXQH_MSGDESC_REPLYTOQMGR       DS   CL48  Name of reply queue manager
MQXQH_MSGDESC_USERIDENTIFIER    DS   CL12  User identifier
MQXQH_MSGDESC_ACCOUNTINGTOKEN   DS   XL32  Accounting token
MQXQH_MSGDESC_APPLIDENTITYDATA  DS   CL32  Application data relating to
*                                          identity
MQXQH_MSGDESC_PUTAPPLTYPE       DS   F     Type of application that put
*                                          the message
MQXQH_MSGDESC_PUTAPPLNAME       DS   CL28  Name of application that put
*                                          the message
MQXQH_MSGDESC_PUTDATE           DS   CL8   Date when message was put
MQXQH_MSGDESC_PUTTIME           DS   CL8   Time when message was put
MQXQH_MSGDESC_APPLORIGINDATA    DS   CL4   Application data relating to
*                                          origin
MQXQH_MSGDESC_LENGTH            EQU  *-MQXQH_MSGDESC
                                ORG  MQXQH_MSGDESC
MQXQH_MSGDESC_AREA              DS   CL(MQXQH_MSGDESC_LENGTH)
*
MQXQH_LENGTH                    EQU  *-MQXQH
                                ORG  MQXQH
MQXQH_AREA                      DS   CL(MQXQH_LENGTH)
```

Deklarace jazyka Visual Basic

```
Type MQXQH
  StrucId        As String*4  'Structure identifier'
  Version        As Long      'Structure version number'
  RemoteQName    As String*48 'Name of destination queue'
  RemoteQMgrName As String*48 'Name of destination queue manager'
  MsgDesc        As MQMD1     'Original message descriptor'
End Type
```

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.

Popisy volání

Tento oddíl popisuje volání MQI.

- “MQBACK-Vrátit změny” na stránce 586
- “MQBEGIN-Begin unit of work” na stránce 589

- “MQBUFMH-Převedení vyrovnávací paměti na popisovač zprávy” na stránce 593
- “MQCB-Správa zpětného volání” na stránce 596
- “MQCB_FUNCTION-Funkce zpětného volání” na stránce 606
- “MQCLOSE-Zavření objektu” na stránce 607
- “MQCMIT-Potvrdit změny” na stránce 615
- “MQCONN-Připojit správce front” na stránce 619
- “MQCONNX-Připojit správce front (rozšířený)” na stránce 627
- “MQCRTMH-Vytvoření manipulátoru zprávy” na stránce 632
- “MQCTL-Řízení zpětných volání” na stránce 635
- “MQDISC-Odpojení správce front” na stránce 641
- “MQDLTMH-Výmaz manipulátoru zprávy” na stránce 645
- “MQDLTMP-Odstranění vlastnosti zprávy” na stránce 647
- “MQGET-Získat zprávu” na stránce 650
- “MQINQ-Dotaz na atributy objektu” na stránce 662
- “MQINQMP-Dotaz na vlastnost zprávy” na stránce 679
- “MQMHBUF-Převést popisovač zprávy do vyrovnávací paměti” na stránce 685
- “MQOPEN-Otevřít objekt” na stránce 689
- “MQPUT-Vložit zprávu” na stránce 706
- “MQPUT1 -Vložení jedné zprávy” na stránce 719
- “MQSET-Nastavit atributy objektu” na stránce 729
- “MQSETMP-nastavení vlastnosti zprávy” na stránce 736
- “MQSTAT-Načíst informace o stavu” na stránce 740
- “MQMHBUF-Převést popisovač zprávy do vyrovnávací paměti” na stránce 685
- “MQSUB-Registrace odběru” na stránce 744
- “MQSUBRQ-Požadavek na odběr” na stránce 750

K dispozici jsou nápověda online na platformách UNIX ve formě stránek *man* , která 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 855.

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 WebSphere 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 zde více informací o datovém typu nebo struktuře v produktu “Elementární datové typy” na stránce 217.

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 824.

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 50 .

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.

Jako hodnotu volání se nevrací žádný parametr; v terminologii C to znamená, že všechna volání vrátí `void`.

Deklarace parametru Vyrovnávací paměť

Volání `MQGET`, `MQPUT`a `MQPUT1` má 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 systému IMS).
- Na serveru IBM itoto volání není podporováno pro aplikace spuštěné v režimu kompatibility.

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.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
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_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 Kódy příčiny .

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 589.
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 IMS) mohou použít volání MQBACK, pokud má jednotka práce vliv pouze na prostředky MQ . Pokud však jednotka práce má vliv na prostředky a prostředky MQ a prostředky patřící k jiným správcům prostředků (například DB2), použijte volání SRRBACK poskytované službou z/OS Recoverable Resource Service (RRS). Volání SRRBACK vrací změny prostředků náležejících ke správcům prostředků, kteří byli povoleni pro koordinaci RRS.
     - Aplikace CICS musí použít příkaz EXEC CICS SYNCPOINT ROLLBACK k zálohování jednotky práce. Nepoužívejte volání MQBACK pro aplikace CICS.
     - Aplikace systému IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. produkt ROLB , aby se odvrátila jednotka práce. Nepoužívejte volání MQBACK pro aplikace systému IMS (jiné než dávkové programy DL/I).
   - 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 části “MQDISC-Odpojení správce front” na stránce 641 .
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 prohlédlo zprávu ve frontě (*nemůže* být součástí jednotky práce).
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 460a v části MQGMO_LOGICAL_ORDER popsané v části “MQGMO-Získat-volby zprávy” na stránce 337.

   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-Připojit správce front” na stránce 619 .
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. Chcete-li se této možnosti vyhnout, musí administrátor nastavit atribut správce front *MaxUncommittedMsgs* na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale jsou dostatečně vysoké, 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 části "MQCNO-Volby připojení" na stránce 292 .

BeginOptions
Typ: MQBO-input/output

Jedná se o volby, které řídí akci MQBEGIN, jak je popsáno v tématu "MQBO-Začátek voleb" na stránce 255.

Nejsou-li vyžadovány žádné volby, programy napsané v C nebo S/390 assembler mohou uvádět adresu parametru s hodnotou 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.

reason
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 příčiny najdete v tématu Kódy příčiny.

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:
   - **Koordinovaná lokální transakce správce front**: Pracovní jednotka, v níž je správce front jediným účastníkem správce prostředků, a správce front tak vystupuje jako koordinátor jednotky 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.
   - **Globální jednotka práce koordinovaná správcem front**: 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 části “MQDISC-Odpojení správce front” na stránce 641 .

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*, *Buffer*, *BufferLength*, *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-Vyrovnávací paměť pro volby obsluhy zprávy” na stránce 253.

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 589.

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* nula, *Buffer* není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assemblerem 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.

reason
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 Kódy příčiny.

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 WebSphere 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.

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 a v produktu IBM i pro aplikace spuštěné v režimu kompatibility můžete pro produkt *MQHC_DEF_HCONN* určit následující speciální hodnotu, která má použít manipulátor připojení přidružený k této prováděcí jednotce.

operation
Typ: MQLONG-vstup

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, hodnoty mohou být následující:

- sečíst (žádnou konstantu nepřičítejte vícekrát než jednou) nebo
- zkombinovat pomocí bitové operace OR (jestliže 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-Získat-volby zprávy” na stránce 337, 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.

reason
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 systémem 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.

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 Kódy příčiny.

Poznámky k použití

1. MQCB se používá k definování akce, která má být vyvolána pro každou zprávu, odpovídající zadaným kritériím, která je k dispozici ve frontě. Když je akce zpracována, buď je zpráva odebrána z fronty a předána definovanému spotřebiteli zpráv, nebo je poskytnut token zprávy, který se 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 562. 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 WebSphere 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 563. 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 WebSphere 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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility lze volání MQCONN vynechat a následující hodnotu určenou pro Hconn:

Objekt MQHC_DEF_CONN

Výchozí popisovač připojení.

MsgDesc
Typ: MQMD-vstup

Tato struktura popisuje atributy načtené zprávy.

Podrobnosti viz “MQMD-deskriptor zprávy” na stránce 383.

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 257.

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 pomocí CICS LINK nebo CICS START načítají parametry pomocí služeb CICS prostřednictvím pojmenovaných objektů známých jako kontejnery kanálů. Názvy kontejnerů jsou stejné jako názvy parametrů. Další informace naleznete v dokumentaci k produktu CICS .
4. Rutiny zpětného volání mohou 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-Zavření objektu

Volání MQCLOSE se vzdá přístupu k objektu a je inverzní k volání MQOPEN a MQSUB.

Syntaxe

MQCLOSE (*Hconn*, *Hobj*, *Options*, *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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility můžete vynechat volání MQCONN a zadat pro produkt *Hconn*následující hodnotu:

MQC_DEF_HCONN
Výchozí popisovač připojení.

HOBJ
Typ: MQHOBJ-vstupní/výstupní

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:

MQHO_UNUSABLE_HOBJ
Nepoužitelná obsluha objektu.

V systému z/OSje hodnota *Hobj* nastavena na nedefinovanou hodnotu.

Volby
Typ: MQLONG-vstup

Tento parametr řídí, jak je objekt uzavřen.

Pouze trvalé dynamické fronty a odběry lze zavřít více než jedním způsobem, protože tyto fronty musí být 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 787). Volby zavření jsou shrnuty v tomto tématu.

Trvalé odběry lze buď zachovat, nebo odebrat; tyto odběry jsou vytvářeny pomocí volání MQSUB s volbou MQSO_DURABLE.

Při zavírání popisovače do spravovaného místa určení (tj. parametr *Hobj* vrácený při volání MQSUB, který používal volbu MQSO_MANAGED) správce front vyčistí všechny publikace, které nebyly načteny při odebrání přidruženého odběru. Odběr se odebere s použitím volby MQCO_REMOVE_SUB na parametru *Hsub* vráceného ve volání MQSUB. Poznámka MQCO_REMOVE_SUB je výchozí chování MQCLOSE pro netrvalý odběr.

Při zavírání popisovače do nespravovaného místa určení jste zodpovědni za vyčištění fronty, kde jsou publikování odesílána. Ukončete odběr nejprve pomocí funkce MQCO_REMOVE_SUB a poté všechny zprávy z fronty zpracujte, dokud nezůstalo žádné levé.

Musíte uvést jednu volbu pouze z následujících možností:

Volby dynamické fronty: Tyto volby řídí, jak jsou zavírány trvalé dynamické fronty.

MQCO_DELETE
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 MQRC_OPTION_NOT_VALID_FOR_TYPE a objekt není odstraněn.

Pokud je v systému z/OSfronta dynamická fronta, která byla logicky odstraněna, a toto je poslední manipulátor fronty, je fronta fyzicky odstraněna. Další podrobnosti viz “Poznámky k použití” na stránce 613 .

MQCO_DELETE_PURGE
Fronta se odstraní a všechny zprávy na ní budou vyprázdněny, pokud je splněna jedna 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 MQRC_OPTION_NOT_VALID_FOR_TYPE a objekt není odstraněn.

Tabulka zobrazuje, které volby zavření jsou platné, a zda je objekt zachován nebo odstraněn.			
Typ objektu nebo fronty	**MQCO_NONE**	**MQCO_DELETE**	**MQCO_DELETE_PURGE**
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 nadále vyčištěny publikace, které čekají na čtení aplikací. Tyto volby jsou platné pouze pro použití s manipulátorem na objekt vrácený v parametru *Hsub* volání MQSUB.

MQCO_KEEP_SUB
Manipulátor s odběrem je uzavřen, ale odběr je zachován. Publikování budou nadále odesílána do místa určení určeného v odběru. Tato volba je platná pouze v případě, že byl proveden odběr s volbou MQSO_DURABLE.

MQCO_KEEPP_SUB je výchozí hodnota, je-li odběr trvalý

MQCO_REMOVE_SUB
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, odeberou se všechny nenačtené publikace.

Hodnota MQCO_REMOVE_SUB je výchozí hodnotou, pokud je odběr netrvalý.

Tyto volby uzavření odběru jsou shrnuty v následujících tabulkách.

Chcete-li zavřít trvalý popisovač odběru, ale zachovat odběr, použijte následující volby uzavření odběru:

Úloha	Volba uzavření odběru
Ponechat publikace na obslužné rutiny MQOPENed	MQCO_KEEP_SUB
Odebrat publikace na obslužné rutiny MQOPENed	Akce není povolena
Ponechat publikace na obslužné rutiny MQSO_MANAGED	MQCO_KEEP_SUB
Odebrat publikace na obslužné rutiny MQSO_MANAGED	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:

Úloha	Volba uzavření odběru
Ponechat publikace na obslužné rutiny MQOPENed	MQCO_REMOVE_SUB
Odebrat publikace na obslužné rutiny MQOPENed	Akce není povolena
Ponechat publikace na obslužné rutiny MQSO_MANAGED	MQCO_REMOVE_SUB

Volby dopředného čtení: Následující volby řídí, co se stane s netrvalými zprávami, které byly odeslány klientovi dříve, než je aplikace požadovala a ještě nebyla využita aplikací. 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.

MQCO_IMMEDIATE
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.

MQCO_QUIESCE
Je učiněn požadavek na uzavření objektu, ale pokud byly všechny zprávy, které byly odeslány klientovi před požadovanou aplikací, stále umístěny v vyrovnávací paměti čtení napřed klienta, volání MQCLOSE se vrátí s varováním MQRC_READ_AHEAD_MSGS a popisovač objektu zůstává 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 se klientovi neodešlou před tím, než je aplikace požaduje, čtení napřed je nyní vypnuto.

Aplikace jsou doporučeny pro použití funkce MQCO_QUIESCE místo pokusů o dosažení bodu, kdy v klientské vyrovnávací paměti čtení napřed nejsou žádné další zprávy, protože by mohla být doručena zpráva mezi posledním voláním MQGET a následujícím příkazem MQCLOSE, které by bylo vyřazeno, pokud byl použit příkaz MQCO_IMMEDIATE.

Je-li funkce MQCLOSE s MQCO_QUIESCE vydána v rámci asynchronní funkce zpětného volání, použije se stejné chování při čtení zpráv s dopředným čtením. Je-li vráceno varování MQRC_READ_AHEAD_MSGS, pak je funkce zpětného volání volána alespoň jednou. Když poslední zbývající zpráva, která byla dopředným čtením, byla předána do funkce zpětného volání, pole MQCBC ConsumerFlags je nastaveno na MQCBCF_READA_BUFFER_EMPTY.

Výchozí volba: Pokud nepožadujete žádnou z výše popsaných voleb, můžete použít následující volbu:

MQCO_NONE
Není vyžadováno žádné volitelné ukončení zpracování.

Tato *musí* být uvedena 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 výše uvedených 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.

Pokud je v systému z/OSfronta dynamická fronta, která byla logicky odstraněna, a toto je poslední manipulátor fronty, je fronta fyzicky odstraněna. Další podrobnosti viz “Poznámky k použití” na stránce 613 .

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.

reason
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:

SKUPINA MQRC_INCOMPLETE_GROUP
(2241, X'8C1') Skupina zpráv není úplná.

ZPRÁVA MQRC_INCOMPLETE_MSG
(2242, X'8C2') Logická zpráva není úplná.

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_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_CICS_WAIT_FAILED
(2140, X'85C') Požadavek na čekání byl odmítnut systémem 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.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

MQRC_OBJECT_DAMAGED
(2101, X'835 ') Objekt je poškozen.

MQRC_OPTION_NOT_VALID_FOR_TYPE
(2045, X'7FD') Při volání MQOPEN nebo MQCLOSE: volba není platná pro daný 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.

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.

MQRC_Q_NOT_EMPTY
(2055, X'807 ') Fronta obsahuje jednu nebo více zpráv nebo nepotvrzené vložení nebo získání požadavků.

MQRC_READ_AHEAD_MSGS
(nnnn, X'xxx ') Klient četl zprávy s dopředným čtením, které dosud aplikace nespotřebovaly.

PROBLÉM MQRC_RESOURCE_PROBLEM
(2102, X'836 ') Není k dispozici dostatek systémových prostředků.

MQRC_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

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.

Podrobné informace o těchto kódech najdete v tématu Kódy příčiny.

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 MQCO_NONE.
2. Následující body se používají, je-li zavřen objekt *queue*:
   - Jsou-li operace ve frontě prováděny jako součást pracovní jednotky, lze frontu zavřít před nebo po bodu synchronizace bez ovlivnění výsledku synchronizačního bodu. Je-li fronta spuštěna, provedení odvolání před uzavřením fronty může způsobit, že bude vydána zpráva spouštěče. Další informace o zprávách spouštěče najdete v tématu Vlastnosti zpráv spouštěče.
   - Pokud byla fronta otevřena s volbou MQOO_BROWSE, je kurzor procházení zničen. Je-li fronta znovu otevřena pomocí volby MQOO_BROWSE, bude vytvořen nový kurzor procházení (viz MQOO_BROWSE).
   - Je-li v době volání funkce MQCLOSE zamknuta zpráva pro tento manipulátor, zámek se uvolní (viz MQGMO_LOCK).
   - Pokud v systému z/OSexistuje požadavek MQGET s volbou MQGMO_SET_SIGNAL s nevyřízeným manipulátorem s manipulátorem fronty, je požadavek zrušen (viz MQGMO_SET_SIGNAL). Požadavky na signál pro stejnou frontu, ale složené proti různým popisům (*Hobj*), nejsou ovlivněny (pokud se odstraňuje dynamická fronta, v tom případě jsou také zrušena).
3. Následující body se použijí, pokud objekt, který se uzavírá, je *dynamická fronta* (buď trvalá, nebo dočasná):
   - U dynamické fronty můžete zadat volby MQCO_DELETE a MQCO_DELETE_PURGE bez ohledu na volby určené v odpovídajícím volání MQOPEN.
   - Když je odstraněna dynamická fronta, všechna volání MQGET s volbou MQGMO_WAIT, která jsou nevyřízeny proti frontě, jsou zrušena a vrátí se kód příčiny MQRC_Q_DELETED. Viz MQGMO_WAIT.

     Ačkoli aplikace nemohou přistupovat k odstraněné frontě, 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.

     V systému z/OSje fronta, která byla logicky odstraněna, ale dosud nebyla odebrána ze systému, brání vytvoření nové fronty se stejným názvem jako odstraněná fronta; volání MQOPEN selže s kódem příčiny MQRC_NAME_IN_USE v tomto případě. Taková fronta se také může stále zobrazovat pomocí příkazů MQSC, i když k ní aplikace nemají přístup.
   - Je-li odstraněna trvalá dynamická fronta, je-li popisovač *Hobj* uvedený v volání MQCLOSE *ne* ten, který byl vrácen voláním MQOPEN, který vytvořil frontu, byla provedena kontrola, že identifikátor uživatele, který byl použit k ověření volání MQOPEN, je oprávněn k odstranění fronty. Pokud byla v rámci volání MQOPEN určena volba MQOO_ALTERNATE_USER_AUTHORITY, kontrolovaný identifikátor uživatele je *AlternateUserId*.

     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 jednotky práce se neselžou. Jak je však popsáno výše, prostředky přidružené k pracovním jednotkám se neuvolní, dokud není každá z jednotek práce potvrzena nebo vrácena zpět.
4. Následující body se použijí, je-li objekt, který se zavírá, *distribuční seznam*:

- Jedinou platnou volbou zavření pro distribuční seznam je MQCO_NONE; volání selže s kódem příčiny MQRC_OPTIONS_ERROR nebo MQRC_OPTION_NOT_VALID_FOR_TYPE, 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 *CompCode* 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 *CompCode* a *Reason* volání jsou nastaveny tak, aby vracely informace popisující selhání. Je možné, aby kód dokončení byl MQCC_FAILED, přestože 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 *CompCode* a *Reason* .

5. Pokud byla v produktu IBM ipři zadání prvního volání MQOPEN implicitně připojena aplikace implicitně, dojde k implicitní MQDISC při vydání posledního příkazu MQCLOSE.

   Implicitně lze připojit pouze aplikace spuštěné v režimu kompatibility; jiné aplikace musí explicitně zadat volání MQCONN nebo MQCONNX pro explicitní připojení ke správci front.

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 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.

- V systému z/OSje volání používáno pouze dávkovými programy (včetně dávkově dávkových programů DL/I IMS).
- Na serveru IBM itoto volání není podporováno pro aplikace spuštěné v režimu kompatibility.

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.

reason
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 Kódy příčiny.

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 prostředků produktu WebSphere MQ .
   - Globální jednotka práce, kde mohou změny ovlivnit prostředky patřící jiným správcům prostředků a které ovlivňují prostředky produktu WebSphere MQ .

   Další podrobnosti o lokálních a globálních jednotkách práce viz “MQBEGIN-Begin unit of work” na stránce 589.
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 IMS) mohou použít volání MQCMIT, pokud jednotka práce ovlivňuje pouze prostředky produktu WebSphere MQ . Pokud však jednotka práce má vliv na prostředky a prostředky produktu WebSphere MQ i na prostředky patřící k jiným správcům prostředků (například DB2), použijte volání SRRCMIT poskytované službou 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 CICS .
     - Aplikace IMS (jiné než dávkové DL/I programy) musí používat volání IMS , jako např. GU a CHKP , aby potvrzujte jednotku práce. Volání MQCMIT nelze použít pro aplikace IMS (jiné než dávkové programy DL/I).
   - 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 manipulátor připojení; všechny volání WebSphere MQ , které ovlivňují konkrétní jednotku práce, musí být prováděny 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. Pro ochranu před tímto administrátorem musí administrátor nastavit atribut správce front *MaxUncommittedMsgs* na hodnotu, která je dostatečně nízká, aby zabránila úniku aplikací, které zaplňují fronty, ale jsou dostatečně vysoké, aby umožnily správné fungování očekávaných aplikací systému zpráv.
8. Pokud je u systémů UNIX a 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-Připojit správce front

Volání MQCONN připojí aplikační program ke správci front.

Poskytuje manipulátor připojení ke správci front, který aplikace používá při následných voláních front zpráv.

- Na systémech z/OSnemusí aplikace CICS volat toto volání. Tyto aplikace jsou automaticky připojeny ke správci front, ke kterému je připojen systém CICS . Nicméně volání MQCONN a MQDISC jsou stále přijímána z aplikací CICS .
- V systému IBM i nemusí být aplikace spuštěné v režimu kompatibility k tomuto volání vydány. Tyto aplikace jsou automaticky připojeny ke správci front, když vydají první volání MQOPEN. Nicméně volání MQCONN a MQDISC jsou nadále přijímána z aplikací IBM i.

  Jiné aplikace (tj. aplikace, které nejsou spuštěny v režimu kompatibility), musí pro připojení ke správci front používat volání MQCONN nebo MQCONNX a volání MQDISC pro odpojení od správce front. To je doporučený styl programování.

Připojení klienta nelze provést na instalaci pouze serveru a lokální připojení nelze provést pouze u instalace klienta.

Syntaxe

MQCONN (*QMgrName*, *Hconn*, *CompCode*, *Reason*)

Parametry

QMgrName
Typ: MQCHAR48 -Vstup

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 systémech, které používají EBCDIC Katakana, nelze použít malá písmena.
- V systému z/OSnemohou být názvy, které začínají nebo končí podtržítkem, zpracovány operacemi a řídicími panely. Z tohoto důvodu se takovým jménům vyhýbejte.
- V systému IBM i ohraničte názvy obsahující malá písmena, dopředné lomítko nebo procento v uvozovkách, je-li to uvedeno v příkazech. Neuvádějte tyto uvozovky do parametru *QMgrName* .

Je-li název tvořen zcela mezerami, použije se název *výchozího* správce front.

Název zadaný pro *QMgrName* musí být název správce front *connectable* .

V systému z/OSjsou správci front, k nimž je možné připojit, určovány prostředím:

- Pro CICSmůžete použít pouze správce front, ke kterému je připojen systém CICS . Parametr *QMgrName* musí být stále zadán, ale jeho hodnota je ignorována; doporučuje se mezery.
- Pro IMSjsou připojitelné pouze správce front, kteří jsou uvedeni v tabulce definic subsystému (CSQQDEFV), *a* vypsaný v tabulce SSM v IMS(viz poznámka k použití 6).
- Pro dávky z/OS a TSO, pouze správci front, kteří jsou umístěni ve stejném systému jako aplikace, jsou připojitelné (viz poznámka o použití 6).

Skupiny sdílení front:V systémech, ve kterých existuje několik správců front a jsou konfigurováni pro vytvoření skupiny sdílení front, lze název skupiny sdílení front zadat pro produkt *QMgrName* na místě názvu správce front. To umožňuje aplikaci připojit se k *libovolnému* správci front, který je k dispozici ve skupině sdílení front, a který se nachází ve stejném obrazu z/OS jako aplikace. Systém může být také konfigurován tak, aby se místo výchozího správce front připojoval do skupiny sdílení front prázdná hodnota *QMgrName* .

Pokud parametr *QMgrName* určuje název skupiny sdílení front, ale v systému je také správce front s tímto názvem, bude připojení k původní skupině preferované. Pouze v případě, že připojení selže, je pokus o připojení k jednomu ze správců front v dané skupině sdílení front.

Je-li připojení úspěšné, můžete použít popisovač vrácený voláním MQCONN nebo MQCONNX pro přístup ke *všem* prostředkům (sdíleným i nesdíleným), které patří ke správci front, k němuž došlo k připojení. Přístup k těmto prostředkům je předmětem typického řízení autorizace.

Pokud aplikace vydá dvě volání MQCONN nebo MQCONNX k vytvoření souběžných připojení a jedno nebo obě volání určuje název skupiny sdílení front, druhý volání vrátí kód dokončení MQCC_WARNING a kód příčiny MQRC_ALREADY_CONNECTED, když se připojuje 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 WebSphere MQ MQI: Pro klientské aplikace WebSphere MQ MQI se pro každou definici kanálu připojení klienta s určeným názvem správce front pokusí o připojení, dokud nebude jedna z nich úspěšná. Správce front však musí mít stejný název jako určený název. Je-li zadán název all-blank, bude každý kanál připojení klienta se všemi mezerové názvy správce front úspěšný, dokud nebude jeden úspěšný. V tomto případě se nekontroluje skutečné jméno správce front.

Klientské aplikace WebSphere MQ nejsou podporovány v systému z/OS, ale produkt z/OS může pracovat jako server WebSphere MQ , ke kterému se mohou aplikace klienta WebSphere MQ připojit.

WebSphere MQ -Skupiny správců front klienta MQI: Pokud zadaný název začíná hvězdičkou (*), může mít správce front, k němuž je vytvořeno připojení, 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 pokusí o každou z nich, dokud nebude nalezeno připojení, na které lze navázat spojení. Pořadí pokusů o připojení je ovlivněno hodnotami váhy kanálu klienta a afinity připojení kandidátských kanálů. Není-li pro připojení k dispozici žádný správce front ve skupině, volání se nezdaří. Každý správce front je zkoušeni pouze jednou. Je-li pro název uvedena hvězdička, použije se výchozí skupina správce front definovaná implementací.

Skupiny správců front jsou podporovány pouze pro aplikace spuštěné v prostředí klienta MQ. Volání se nezdaří, pokud aplikace typu non-client určuje název správce front začínající hvězdičkou. Skupina je definována poskytnutím několika definic kanálů připojení klienta se stejným jménem správce front (zadaným názvem 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 objektu určovat mezery jako název správce front, ke kterému je aplikace připojena (*lokální správce front*). Pokud aplikace potřebuje znát tento název, použijte volání MQINQ k dotazu na atribut správce front *QMgrName* .

Při určení předpony názvu připojení je nutné, aby aplikace nebyla závislá na připojení ke konkrétnímu správci front ve skupině. Vhodné aplikace jsou:

- Aplikace, které vložila zprávy, ale nedostali zprávy.
- Aplikace, které vložila zprávy požadavků a poté získaly zprávy odpovědí z *dočasné dynamické* fronty.

Nevhodné aplikace jsou takové, které potřebují získat zprávy z určité fronty v konkrétním správci front; takové aplikace nesmí před názvy s hvězdičkou předponu.

Pokud uvedete hvězdičku, maximální délka zbytku názvu je 47 znaků.

Skupiny správců front nejsou v systému z/OSpodporovány.

Délka tohoto parametru je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

Hconn

Typ: MQHCONN-výstup

Tento manipulátor představuje připojení ke správci front. Určete ji ve všech následných voláních front zpráv vydaných aplikací. Po zadání volání MQDISC přestane být platná, nebo když se ukončí jednotka zpracování, která definuje rozsah manipulátorů.

Produkt WebSphere MQ nyní dodává knihovnu mqm s klientskými balíky a s balíky serveru. To znamená, že při volání MQI, které bylo nalezeno v knihovně mqm, je zkontrolován typ připojení a zjistí se, zda se jedná o připojení klienta nebo serveru, a pak se provede správné základní volání. Proto je možné ukončit proceduru *Hconn* , která je nyní propojena s knihovnou mqm, ale je použita při instalaci klienta.

Rozsah manipulátoru:Obor vráceného manipulátoru závisí na volání, které se používá k připojení ke správci front (MQCONN nebo MQCONNX). Je-li použito volání MQCONNX, závisí rozsah manipulátoru také na volbě MQCNO_HANDLE_SHARE_ * určenou v poli *Options* struktury MQCNO.

- Je-li volání MQCONN nebo je zadána volba MQCNO_HANDLE_SHARE_NONE, vrácený popisovač je *nesdílený* popisovač.

  Rozsah nesdílené obslužné rutiny je nejmenší jednotka paralelního zpracování podporovaná platformou, na které je aplikace spuštěna (podrobnosti viz Tabulka 564 na stránce 622); popisovač není platný mimo jednotku paralelního zpracování, ze které bylo volání vydáno.
- Určíte-li volbu MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrácený popisovač bude *sdílený* popisovač.

  Rozsah sdílené popisovače je proces, který vlastní podproces, ze kterého bylo volání vydáno; popisovač lze použít z libovolného podprocesu, který patří k tomuto procesu. Ne všechny platformy podporují podprocesy.
- Pokud volání MQCONN nebo MQCONNX selže s kódem dokončení rovnajícím se MQCC_FAILED, hodnota Hconn není definována.

Tabulka 564. Rozsah nesdílených manipulátorů na různých platformách	
Platforma	**Rozsah nesdíleného manipulátoru**
z/OS	• CICS: úloha CICS • IMS: úloha až do dalšího bodu synchronizace (kromě dílčích úloh dané úlohy). • Dávka systému z/OS a TSO: úloha (kromě dílčích úloh dané úlohy)
IBM i	Úloha
Systémy UNIX	Podproces
16bitové aplikace systému Windows	Proces
32bitové aplikace systému Windows	Podproces

V systému z/OS pro aplikace CICS a v systému IBM i pro aplikace spuštěné v režimu kompatibility je vrácena tato hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

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.

reason
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Aplikace je již připojena.

CHYBA MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Nelze načíst uživatelskou proceduru pracovní zátěže klastru.

MQRC_SSL_ALREADY_INITIALIZOVÁNO
(2391, X' 957 ') SSL je již inicializováno.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851 ') Nelze načíst modul připojení adaptéru.

CHYBA MQRC_ADAPTER_DEFS_ERROR
(2131, X'853 ') Modul definice subsystému adaptéru není platný.

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.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ADAPTER_STORAGE_NEDOSTATEK
(2127, X'84F') Nedostatek paměti pro adaptér.

PŘIPOJENÉ MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837 ') Jiný správce front je již připojen.

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.

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_CONN_ID_IN_USE
(2160, X'870 ') Identifikátor připojení je již používán.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

CHYBA PŘIPOJENÍ MQRC_CONNECTION_ERROR
(2273, X'8E1') Chyba při zpracování volání MQCONN.

PŘIPOJENÍ MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') Vyskytuje se na volání MQCONN nebo MQCONNX, když správce front není schopen poskytnout připojení požadovaného typu připojení v aktuální instalaci. Připojení klienta nelze provést pouze na instalaci serveru. Lokální připojení nelze provést pouze u instalace klienta.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CRYPTO_HARDWARE_ERROR
(2382, X'94E') Chyba konfigurace kryptografického hardwaru.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873 ') Koordinátor pro zotavení existuje.

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_HOST_NOT_AVAILABLE
(2538, X'9EA') Volání MQCONN bylo vydáno z klienta pro připojení ke správci front, ale pokus o alokaci konverzace se vzdáleným systémem selhal.

NESHODA MQRC_INSTALLATION_MATCH
(2583, X'A17') Neshoda mezi instalací správce front a vybranou knihovnou.

CHYBA MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Klíčové úložiště není platné.

MQRC_MAX_CONNS_LIMIT_DOSAŽEN
(2025, X'7E9') Bylo dosaženo maximálního počtu připojení.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

FUNKCE MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

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_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

CHYBA MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959 ') Chyba inicializace SSL.

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 Kódy příčiny.

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 které 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. Do těchto front je možné vkládat zprávy, nikoli však přijímat zprávy z těchto front.
3. Pokud správce front selže při spuštění aplikace, musí aplikace znovu vydat volání MQCONN, aby získal nový manipulátor připojení pro použití při následných voláních produktu WebSphere MQ . Aplikace může volání MQCONN periodicky volat, dokud nebude volání úspěšné.

   Pokud si aplikace není jistá, zda je připojena ke správci front, může aplikace bezpečně vydat volání MQCONN pro získání manipulátoru připojení. Je-li aplikace již připojena, vrácený popisovač je stejný jako vrácený předchozí volání MQCONN, ale s kódem dokončení MQCC_WARNING a kódem příčiny MQRC_ALREADY_CONNECTED.
4. Pokud byla aplikace dokončena s použitím volání produktu WebSphere MQ , musí aplikace použít volání MQDISC k odpojení od správce front.
5. Pokud se volání MQCONN nezdaří s kódem dokončení rovnajícím se MQCC_FAILED, hodnota Hconn není definována.
6. V systému z/OS:
   - Aplikace Batch, TSO a IMS musí při volání volání MQCONN volat další volání produktu WebSphere MQ . Tyto aplikace se mohou souběžně připojovat k více než jednomu správci front.

Pokud správce front selže, musí aplikace po restartování správce front znovu vyvolat volání, aby získal nový manipulátor připojení.

Ačkoli aplikace IMS mohou volat volání MQCONN opakovaně, a to i v případě, že je již připojeno, není to doporučováno pro programy zpracování zpráv online (MPP).

- Aplikace CICS nemusí volat volání MQCONN pro použití dalších volání produktu WebSphere MQ , ale mohou tak učinit, pokud chtějí, volání MQCONN a volání MQDISC jsou akceptovány. Souběžně se však nelze připojit k více než jednomu správci front.

  Pokud správce front selže, jsou tyto aplikace automaticky znovu připojeny, když se správce front restartuje, a proto není třeba volat volání MQCONN.

7. Chcete-li definovat dostupné správce front v systému z/OS, postupujte takto:
   - Pro dávkové aplikace mohou systémoví programátoři použít makro CSQBDEF k vytvoření modulu (CSQBDEFV), který definuje výchozí název správce front, nebo název skupiny sdílení front.
   - Pro aplikace IMS mohou programátoři systému 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 ke správci front. Chcete-li to provést, musíte vytvořit člena subsystému v IMS.Knihovna PROCLIB a identifikace člena subsystému s příslušnými oblastmi IMS . Pokusí-li se aplikace o připojení ke správci front, který není definován ve členu subsystému pro jeho oblast IMS , dojde k ukončení aplikace.
8. V systému IBM i mohou být aplikace napsané pro předchozí verze správce front spuštěny bez opětovné kompilace. Tomu se říká *režim kompatibility*. Tento režim provozu poskytuje kompatibilní běhové prostředí pro aplikace. Skládá se z následujících:
   - Servisní program AMQZSTUB se nachází v knihovně QMQM.

     AMQZSTUB poskytuje stejné veřejné rozhraní jako předchozí vydání a má stejný podpis. Tento servisní program použijte pro přístup k rozhraní MQI prostřednictvím volání procedury bound.
   - Program QMQM se nachází v knihovně QMQM.

     QMQM poskytuje prostředky pro přístup k rozhraní MQI prostřednictvím dynamických volání programů.
   - Programy MQCLOSE, MQCONN, MQDISC, MQGET, MQINQ, MQOPEN, MQPUT, MQPUT1a MQSET umístěné v knihovně QMQM.

     Tyto programy také poskytují prostředky pro přístup k rozhraní MQI prostřednictvím dynamických volání programu, ale s použitím seznamu parametrů, který odpovídá standardním popisům volání produktu WebSphere MQ .

   Tato tři rozhraní nezahrnují schopnosti, které byly představeny v produktu WebSphere MQ verze 5.1. Volání MQBACK, MQCMIT a MQCONNX například nejsou podporována. Podpora poskytovaná těmito rozhraními je určena pouze pro jednovláknové aplikace.

   Podpora nových volání produktu WebSphere MQ v aplikacích s jedním vláknem a pro všechny volání WebSphere MQ v aplikacích s podporou podprocesů je poskytována prostřednictvím servisních programů LIBMQM a LIBMQM_R.
9. V systému IBM i nejsou programy, které se nestandardně ukončí, automaticky odpojeny od správce front. Aplikace pro zápis umožňující možnost volání MQCONN nebo MQCONNX při vracení kódu dokončení MQCC_WARNING a kódu příčiny MQRC_ALREADY_CONNECTED. Použijte obslužnou rutinu připojení vrácenou v této situaci 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 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 v následných voláních produktu WebSphere 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 na všech klientech WebSphere MQ a klientech WebSphere MQ připojených k těmto systémům.
- Na serveru IBM itoto volání není podporováno pro aplikace spuštěné v režimu kompatibility.

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-Připojit správce front” na stránce 619 .

ConnectOpts
Typ: MQCNO-input/output

Podrobnosti viz “MQCNO-Volby připojení” na stránce 292.

Hconn
Typ: MQHCONN-výstup

Tento manipulátor představuje připojení ke správci front. Určete ji ve všech následných voláních front zpráv vydaných aplikací. Po zadání volání MQDISC přestane být platná, nebo když se ukončí jednotka zpracování, která definuje rozsah manipulátorů.

Produkt WebSphere MQ nyní dodává knihovnu mqm s klientskými balíky a s balíky serveru. To znamená, že při volání MQI, které bylo nalezeno v knihovně mqm, je zkontrolován typ připojení a zjistí se, zda se jedná o připojení klienta nebo serveru, a pak se provede správné základní volání. Proto je možné ukončit proceduru *Hconn* , která je nyní propojena s knihovnou mqm, ale je použita při instalaci klienta.

Rozsah manipulátoru:Obor vráceného manipulátoru závisí na volání, které se používá k připojení ke správci front (MQCONN nebo MQCONNX). Je-li použito volání MQCONNX, závisí rozsah manipulátoru také na volbě MQCNO_HANDLE_SHARE_ * určenou v poli *Options* struktury MQCNO.

- Je-li volání MQCONN nebo je zadána volba MQCNO_HANDLE_SHARE_NONE, vrácený popisovač je *nesdílený* popisovač.

  Rozsah nesdílené obslužné rutiny je nejmenší jednotka paralelního zpracování podporovaná platformou, na které je aplikace spuštěna (podrobnosti viz Tabulka 565 na stránce 628); popisovač není platný mimo jednotku paralelního zpracování, ze které bylo volání vydáno.
- Určíte-li volbu MQCNO_HANDLE_SHARE_BLOCK nebo MQCNO_HANDLE_SHARE_NO_BLOCK, vrácený popisovač bude *sdílený* popisovač.

  Rozsah sdílené popisovače je proces, který vlastní podproces, ze kterého bylo volání vydáno; popisovač lze použít z libovolného podprocesu, který patří k tomuto procesu. Ne všechny platformy podporují podprocesy.
- Pokud volání MQCONN nebo MQCONNX selže s kódem dokončení rovnajícím se MQCC_FAILED, hodnota Hconn není definována.

Tabulka 565. Rozsah nesdílených manipulátorů na různých platformách	
Platforma	**Rozsah nesdíleného manipulátoru**
z/OS	• CICS: úloha CICS • IMS: úloha až do dalšího bodu synchronizace (kromě dílčích úloh dané úlohy). • Dávka systému z/OS a TSO: úloha (kromě dílčích úloh dané úlohy)
IBM i	Úloha
Systémy UNIX	Podproces
16bitové aplikace systému Windows	Proces
32bitové aplikace systému Windows	Podproces

V systému z/OS pro aplikace CICS a v systému IBM i pro aplikace spuštěné v režimu kompatibility je vrácena tato hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

CompCode
Typ: MQLONG-výstup

Podrobné informace naleznete v popisu parametru *CompCode* popsaného v příručce “MQCONN-Připojit správce front” na stránce 619 .

reason
Typ: MQLONG-výstup

Volání MQCONN a MQCONNX mohou vracet následující kódy. Seznam dalších kódů, které mohou být vráceny voláním MQCONNX, najdete v následujících kódech.

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_ALREADY_CONNECTED
(2002, X'7D2') Aplikace je již připojena.

CHYBA MQRC_CLUSTER_EXIT_LOAD_ERROR
(2267, X'8DB') Nelze načíst uživatelskou proceduru pracovní zátěže klastru.

MQRC_SSL_ALREADY_INITIALIZOVÁNO
(2391, X' 957 ') SSL je již inicializováno.

Je-li *CompCode* MQCC_FAILED:

CHYBA MQRC_ADAPTER_CONN_LOAD_ERROR
(2129, X'851 ') Nelze načíst modul připojení adaptéru.

CHYBA MQRC_ADAPTER_DEFS_ERROR
(2131, X'853 ') Modul definice subsystému adaptéru není platný.

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.

CHYBA MQRC_ADAPTER_SERV_LOAD_ERROR
(2130, X'852 ') Nelze načíst modul služby adaptéru.

MQRC_ADAPTER_STORAGE_NEDOSTATEK
(2127, X'84F') Nedostatek paměti pro adaptér.

PŘIPOJENÉ MQRC_ANOTHER_Q_MGR_CONNECTED
(2103, X'837 ') Jiný správce front je již připojen.

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.

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_CONN_ID_IN_USE
(2160, X'870 ') Identifikátor připojení je již používán.

PORCC_CONNECTION_CONNECTION_LO
(2009, X'7D9') Připojení ke správci front bylo ztraceno.

CHYBA PŘIPOJENÍ MQRC_CONNECTION_ERROR
(2273, X'8E1') Chyba při zpracování volání MQCONN.

PŘIPOJENÍ MQRC_CONNECTION_NOT_AVAILABLE
(2568, X'A08') Vyskytuje se na volání MQCONN nebo MQCONNX, když správce front není schopen poskytnout připojení požadovaného typu připojení v aktuální instalaci. Připojení klienta nelze provést pouze na instalaci serveru. Lokální připojení nelze provést pouze u instalace klienta.

MQRC_CONNECTION_QUIESCING
(2202, X'89A') Připojení je uváděno do klidového stavu.

ZASTAVIT_PŘIPOJENÍ_MQRC
(2203, X'89B') Spojení se vypíná.

CHYBA MQRC_CRYPTO_HARDWARE_ERROR
(2382, X'94E') Chyba konfigurace kryptografického hardwaru.

MQRC_DUPLICATE_RECOV_COORD
(2163, X'873 ') Koordinátor pro zotavení existuje.

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_HOST_NOT_AVAILABLE
(2538, X'9EA') Volání MQCONN bylo vydáno z klienta pro připojení ke správci front, ale pokus o alokaci konverzace se vzdáleným systémem selhal.

NESHODA MQRC_INSTALLATION_MATCH
(2583, X'A17') Neshoda mezi instalací správce front a vybranou knihovnou.

CHYBA MQRC_KEY_REPOSITORY_ERROR
(2381, X'94D') Klíčové úložiště není platné.

MQRC_MAX_CONNS_LIMIT_DOSAŽEN
(2025, X'7E9') Bylo dosaženo maximálního počtu připojení.

AUTORIZOVANÝ MQRC_NOT_AUTHORIZED
(2035, X'7F3') Chybí autorizace pro přístup.

FUNKCE MQRC_OPEN_FAILED
(2137, X'859 ') Objekt nebyl úspěšně otevřen.

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_SECURITY_ERROR
(2063, X'80F') Došlo k chybě zabezpečení.

CHYBA MQRC_SSL_INITIALIZATION_ERROR
(2393, X' 959 ') Chyba inicializace SSL.

MQRC_STORAGE_NOT_AVAILABLE
(2071, X'817 ') Není k dispozici dostatek paměti.

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 Kódy příčiny.

Poznámky k použití

Pro programovací jazyk Visual Basic se používá následující bod:

- Parametr *ConnectOpts* je deklarován jako typ MQCNO. Je-li aplikace spuštěna jako klient WebSphere MQ MQI a chcete-li 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 manipulátoru zprávy

Volání MQCRTMH vrací popisovač zprávy.

Aplikace může volání MQCRTMH 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 obsluhy zprávy k vlastnostem vkládaných zpráv. Podobně zadáním manipulátoru zprávy v rámci volání MQGET lze při dokončení volání MQGET přistupovat k vlastnostem načítané zprávy pomocí manipulátoru zprávy.

K odstranění manipulátoru zprávy použijte příkaz MQDLTMH .

Syntaxe

MQCRTMH (*Hconn*, *CrtMsgHOpts*, *Hmsg*, *CompCode*, *Reason*)

Parametry

Hconn

Typ: MQHCONN-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 žádný volání produktu WebSphere MQ není na obslužné rutiny zpráv funkční, volání MQDLTMH je implicitně voláno pro odstranění zprávy.

Případně můžete zadat následující hodnotu:

PŘIPOJENÍ MQC_UNASSOCIATED_HCONN
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; produkt WebSphere MQ nikdy implicitně neodstraní popisovač zprávy.

Musí existovat alespoň jedno platné připojení ke správci front zavedenému na podprocesu, který vytváří obslužnou rutinu zpráv, jinak volání selže s chybou 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. Je vrácen kód příčiny MQRC_HMSG_NOT_AVAILABLE, pokud je popisovač zprávy zadán pro jinou instalaci.

V systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility lze volání MQCONN vynechat a pro produkt *Hconn*můžete zadat následující hodnotu:

MQC_DEF_CONN
Výchozí popisovač připojení

CrtMsgHOpts
Typ: MQCMHO-vstup

Volby, které řídí akci MQCRTMH. Podrobnosti viz MQCMHO .

Hmsg
Typ: MQHMSG-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 tato hodnota 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 argument *Hconn* zadán jako hodnota MQHC_UNASSOCIATED_HCONN, lze obslužnou rutinu vrácené zprávy použít pro volání MQGET, MQPUT nebo MQPUT1 s jakýmkoli připojením v rámci jednotky zpracování, ale může být v daném okamžiku používána pouze jedním voláním WebSphere MQ . Je-li popisovač používán, když se druhý volání WebSphere MQ pokusí použít stejný popisovač zprávy, dojde k selhání druhého volání WebSphere MQ s kódem příčiny MQRC_MSG_HANDLE_IN_USE.

Není-li parametr *Hconn* MQHC_UNASSOCIATED_HCONN, lze s použitím manipulátoru vrácené zprávy použít pouze určené připojení.

Následující hodnota parametru *Hconn* musí být použita v 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.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
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.

MQRC_CALL_IN_PROGRESS
(2219, X'08AB') Volání MQI bylo zadáno před dokončením předchozího volání.

CHYBA MQRC_CMHO_ERROR
(2461, X'099D') Není platná struktura voleb popisovače zprávy vytvoření zprávy.

PORCC_CONNECTION_CONNECTION_LO
(2273, X'7D9') Připojení ke správci front bylo ztraceno.

MQRC_HANDLE_NOT_AVAILABLE
(2017, X'07E1') Nejsou k dispozici žádné další popisovače.

CHYBA MQRC_HCONN_ERROR
(2018, X'7E2') Popisovač připojení není platný.

CHYBA MQRC_HMSG_ERROR
(2460, X'099C') Ukazatel obsluhy zprávy není platný.

CHYBA MQRC_OPTIONS_ERROR
(2046, X'07FE') Volby nejsou platné nebo nejsou konzistentní.

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 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, CRTMSGOPTS, 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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility lze volání MQCONN vynechat a pro produkt *Hconn*můžete zadat následující speciální hodnotu:

MQC_DEF_HCONN
Výchozí popisovač připojení.

operation
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 WebSphere MQ a volání nepochází z funkce odběratele 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 na systému z/OS nebo je-li aplikace svázána s knihovnou WebSphere 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 naleznete v příručce “MQCTLO-Struktura voleb zpětného volání řídicího prvku” na stránce 311 .

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.

reason
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 systémem 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.

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 Kódy příčiny.

Poznámky k použití

1. Rutiny zpětného volání musí zkontrolovat odezvy ze všech služeb, které vyvolávají, a pokud rutina zjistí podmínku, kterou nelze vyřešit, musí vydat příkaz MQCB MQOP_DEREGISTER, který zabrání opakovaným voláním rutiny zpětného volání.
2. V systému z/OS, je-li operace MQOP_START:
   - Programy, které používají asynchronní rutiny zpětného volání, musí být autorizováni pro použití systémových služeb z/OS UNIX System Services (USS).
   - 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 USS pthread_create (callable service BPX1PTC).

3. MQCTL není podporováno v rámci adaptéru IMS .

Poznámka: V produktu CICSnení podporováno MQOP_START. 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, musí hlavní řídicí podproces vydat před ukončením volání MQDISC. Další podrobnosti naleznete v tématu Asynchronní spotřeba zpráv produktu WebSphere MQ .
- V systému z/OSnemusí aplikace CICS vydávat toto volání k odpojení od správce front, ale mohou jej vyžadovat ukončení použití značky připojení.

- Na serveru IBM inemusí být aplikace spuštěné v režimu kompatibility vydávat toto volání. Další informace viz “MQCONN-Připojit správce front” na stránce 619.

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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility můžete vynechat volání MQCONN a zadat pro produkt *Hconn*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 hodnota *Hconn* nastavena na nedefinovanou hodnotu.

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.

reason
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 Kódy příčiny.

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 transakci správce front odesílá 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 systému z/OS, CICS, IMS (jiné než dávkové DL/1 programy) a aplikace RRS jsou podobné.

b. Pokud aplikace skončí normálně, ale bez zadání volání MQDISC, závisí akce na daném prostředí:

   - V systému 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 CICS nemusí vydávat volání MQDISC k odpojení od správce front, protože se sám systém CICS připojuje ke správci front, a volání MQDISC nemá na toto připojení žádný vliv.
   - CICS, IMS (kromě dávkových programů DL/1) a aplikace RRS používají jednotky práce, které jsou koordinovány 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.

4. V produktu IBM inemusí být aplikace spuštěné v režimu kompatibility k tomuto volání vydány; další podrobnosti naleznete v rámci volání MQCONN.

Poznámka: V produktu CICSnení podporováno MQOP_START. 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-Výmaz manipulátoru zprávy

Volání MQDLTMH odstraní popisovač zprávy a je inverzní k volání MQCRTMH.

Syntaxe

MQDLTMH (*Hconn*, *Hmsg*, *DltMsgHOpts*, *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 odstraňuje popisovač zprávy, jinak se volání nezdaří s MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-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:

MAHL_UNUSABLE_HMSG
Nepoužitelná obsluha zprávy.

Popisovač zprávy nelze odstranit, pokud jiný volání WebSphere MQ probíhá, že byl předán stejný popisovač zprávy.

DltMsgHOpts
Typ: MQDMHO-vstup

Podrobnosti viz “MQDMHO-Odstranění voleb zpracování zpráv” na stránce 328.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

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.

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_DMHO_ERROR
(2462, X'099E') Struktura obslužného programu odstranění zprávy není platná.

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.

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 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, DLTMSGOPTS, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
** Connection handle
01  HCONN  PIC S9(9) BINARY.

** Options that control the action of MQDLTMH
01  DLTMSGHOPTS.
   COPY CMQDLMHOV.

** Completion code
01 COMPCODE  PIC S9(9) BINARY.

** Reason code qualifying COMPCODE
 01  REASON   PIC S9(9) BINARY.
```

Vyvolání PL/I

```
call MQDLTMH (Hconn, Hmsg, DltMsgHOpts, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl DltMsgHOpts like MQDMHO;    /* Options that control the action of MQDLTMH */
dcl CompCode    fixed bin(31);  /* Completion code */
dcl Reason      fixed bin(31);  /* Reason code qualifying CompCode */
```

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-Odstranění vlastnosti zprávy

Volání MQDLTMP odstraní vlastnost z manipulátoru zprávy a je inverzní k volání MQSETMP.

Syntaxe

MQDLTMP (*Hconn*, *Hmsg*, *DltPropOpts*, *Name*, *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 odstraňuje manipulační prostředek zprávy, jinak volání selže při selhání MQRC_CONNECTION_BROKEN.

Hmsg
Typ: MQHMSG-vstup

Jedná se o popisovač zprávy obsahující vlastnost, která má být odstraněna. Hodnota byla vrácena předchozím voláním MQCRTMH.

DltPropOpts
Typ: MQDMPO-vstup

Podrobnosti naleznete v datovém typu MQDMPO .

název
Typ: 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.

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.

reason
Typ: MQLONG-výstup

Je-li *CompCode* MQCC_OK:

MQRC_NONE
(0, X'000 ') Chybí důvod k vytvoření sestavy.

Je-li *CompCode* MQCC_WARNING:

MQRC_PROPERTY_NOT_AVAILABLE
(2471, X'09A7') Vlastnost není k dispozici.

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.

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_DMPO_ERROR
(2481, X'09B1') Odstranění struktury voleb vlastnosti zprávy není platné.

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'07FE') Volby nejsou platné nebo nejsou konzistentní.

CHYBA MQRC_PROPERTY_NAME_ERROR
(2442, X'098A') Neplatný název vlastnosti.

CHYBA MQRC_SOURCE_CCSID_ERROR
(2111, X'083F') Identifikátor kódované znakové sady názvu vlastnosti není platný.

CHYBA MQRC_UNEXPECTED_ERROR
(2195, X'0893 ') Vyskytla se neočekávaná chyba.

Podrobné informace o těchto kódech najdete v tématech:

- Kódy příčiny pro produkt WebSphere MQ for z/OS
- Kódy příčin rozhraní API pro další platformy WebSphere MQ

.

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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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 689):

- 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 zprávy" na stránce 383.

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 předponu MQMDE s předponou o datech zprávy aplikace, ale *pouze* , pokud jedno nebo více polí v MQMDE má jinou než výchozí hodnotu. Pokud mají všechna pole v MQMDE výchozí hodnoty, MQMDE se vynechá. Název formátu MQFMT_MD_EXTENSION v poli *Formát* v MQMD označuje, 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-Získat-volby zprávy" na stránce 337.

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 787.

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 WebSphere MQ), ale některé zprávy mohou vyžadovat přísnější zarovnání. Například zpráva obsahující 64bitové binární celé číslo může vyžadovat 8 bajtů zarovnání.

Je-li produkt *BufferLength* menší než délka zprávy, přesune se do produktu *Buffer*co nejvíce zpráv; k tomu dojde, je-li 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. Volbu MQGMO_CONVERT lze použít (v případě potřeby s uživatelem napsanou uživatelskou procedurou) k převodu dat zprávy; podrobnosti o této volbě naleznete v části "MQGMO-Získat-volby zprávy" na stránce 337 .

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* nula, *Buffer* není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assemblerem 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 *nekonvertovaných* 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_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 systémem 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 ') Db2 subsystém 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 Kódy příčiny.

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, který provádí volání MQGET v zastoupení aplikace, nemůže zjistit ztrátu aplikace, dokud náhradní osoba nevrátí zprávu do aplikace, *po* zprávě 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 viz téma Aspekty synchronizačních bodů v aplikacích produktu WebSphere MQ.

Tato situace může nastat u klientů IBM WebSphere MQ a také u aplikací spuštěných 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, zprávy v rámci jednotky práce se stanou dostupnými 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 (pouzez/OS)
     - MQFMT_COMMAND_1
     - MQFMT_COMMAND_2
     - HLAVIČKA MQFMT_DEAD_LETTER_HEADER
     - ZÁHLAVÍ MQFMT_DICT_HEADER
     - MQFMT_EVENT verze 1
     - MQFMT_EVENT verze 2 (pouzez/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 (pouzez/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. Názvy formátů zadaných uživatelem nesmí začínat písmeny WebSphere MQ.

     Podrobné informace o ukončení konverze dat naleznete v části “Uživatelská procedura konverze dat” na stránce 855 .

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 produktu WebSphere MQ , zprávu nelze převést ze znakové sady s dvoubajtovými nebo vícebajtovými znaky 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. Sada znaků Unicode UCS-2 je příkladem takové znakové sady.

Při návratu z MQGET označuje následující kód příčiny, že zpráva byla úspěšně převedena:

- MQRC_NONE

Následující kód příčiny informuje o tom, že zpráva *mohla* byla 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 true pro převody provedené uživatelem napsanou uživatelskou procedurou *pouze* , pokud uživatelská procedura odpovídá pokynům pro zpracování popsaným v tématu “Uživatelská procedura konverze dat” na stránce 855.

9. Když používáte objektově orientované rozhraní k získání zpráv, můžete se rozhodnout nezadat vyrovnávací paměť, která má obsahovat data zprávy pro volání MQGET. V předchozích verzích produktu WebSphere MQvšak bylo možné provést příkaz MQGET s kódem příčiny MQRC_CONVERTERED_MSG_TO_BIG, a to ani v případě, že vyrovnávací paměť nebyla zadána. Když v produktu WebSphere MQ verze 7 obdržíte zprávu s použitím objektově orientované aplikace bez omezení velikosti vyrovnávací paměti pro příjem zpráv, aplikace selže s hodnotou MQRC_CONVERTED_MSG_TOO_BIG a obdrží převedenou zprávu. To platí pro následující prostředí:

   - .NET, včetně plně spravovaných aplikací
   - JAZYK C++
   - Java (třídy WebSphere MQ pro jazyk Java)

   Poznámka: Pro všechny klienty platí, že pokud je hodnota proměnné *sharingConversations* nula, kanál bude pracovat tak, jak byl před verzí produktu WebSphere MQ verze 7.0, a zpracování zpráv se vrátí k chování verze 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 převod dat zprávy aplikace se vyskytne, pouze pokud *vše* platí následující:
      - 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 566. 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. Klientské aplikace si musí být vědomy, že pokud jsou hodnoty MsgId a CorrelId mezi voláními MQGET pozměněny, mohou být předchozí hodnoty již odeslány klientovi a zůstanou ve vyrovnávací paměti pro čtení napřed klienta, dokud nebudou spotřebovány (nebo automaticky vymazány).

    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 a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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 × *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 676 , kde jsou další podrobnosti.

Selektory můžete určit v libovolném pořadí. Hodnoty atributu odpovídající celočíselným selektorům atributů (selektoryMQIA_*) jsou vraceny v produktu *IntAttrs* ve stejném pořadí, ve kterém se tyto selektory vyskytují v produktu *Selectors*. Hodnoty atributu, které odpovídají selektorům znakových atributů (selektoryMQCA_*), 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_USEDnejvyšší 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 napsaným v C nebo S/390 sestavovacím programem 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 567 na stránce 664 MQINQ pro fronty
- Selektory atributů Tabulka 568 na stránce 666 MQINQ pro seznamy názvů
- Selektory atributů produktu Tabulka 569 na stránce 667 MQINQ pro definice procesu
- Selektory atributů produktu Tabulka 570 na stránce 667 MQINQ pro správce front

Všechny selektory jsou podporovány na všech platformách produktu IBM WebSphere MQ kromě těch, které jsou uvedeny ve sloupci **Poznámka** takto:

NEz/OS
Podporováno na všech platformách **kromě** z/OS

z/OS
Podporováno **pouze** v systému z/OS

Tabulka 567. Selektory atributů MQINQ pro fronty

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_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.	NEz/OS
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_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	

Tabulka 567. Selektory atributů MQINQ pro fronty (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_CLWL_Q_RANK	MQLONG	Pořadí fronty	
MQIA_CLWL_USEQ	MQLONG	Použít vzdálené fronty	
MQIA_CURRENT_Q_DEPTH	MQLONG	Počet zpráv ve frontě	
MQIA_DEF_BIND	MQLONG	Výchozí vazba	
MQIA_DEF_INPUT_OPEN_OPTION	MQLONG	Výchozí volba 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

Tabulka 567. Selektory atributů MQINQ pro fronty (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQIA_Q_SERVICE_INTERVAL	MQLONG	Limit pro interval služby fronty	NEz/OS
MQIA_Q_SERVICE_INTERVAL_EVENT	MQLONG	Řídicí atribut pro události intervalu služby fronty	NEz/OS
MQIA_Q_TYPE	MQLONG	Typ fronty	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS
MQIA_RETENTION_INTERVAL	MQLONG	Interval uchování fronty	
MQIA_SCOPE	MQLONG	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 568. 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 × Počet názvů v seznamu	Názvy v seznamu názvů	
MQIA_NAME_COUNT	MQLONG	Počet názvů v seznamu názvů	
MQIA_QSG_DISP	MQLONG	Dispozice skupiny sdílení front	z/OS

Tabulka 569. 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	
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 570. 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_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	

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
MQCA_DNS_GROUP	MQ_DNS_GROUP_NAME_LENGTH	Název skupiny pro modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, ke které má být připojen. 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
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 nainstalován přidruženýIBM WebSphere 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

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
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
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 produktu 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	

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_CHINIT_ADAPTERS	MQLONG	Počet podúloh adaptéru, které mají být použity pro zpracování volání IBM WebSphere 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
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.	NEz/OS
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

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_INTRA_GROUP_QUEUING	MQLONG	Podpora řazení do front v rámci skupiny	z/OS
MQIA_LISTENER_TIMER	MQLONG	Časový interval (v sekundách) mezi pokusy produktu IBM WebSphere MQ 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
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	

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
MQIA_PROT_POLICY_CAPABILITY	MQLONG	Určuje, zda jsou k dispozici funkce zabezpečení produktu WebSphere MQ Advanced Message Security pro správce front.	
MQIA_PUBSUB_MAXMSG_RETRY_COUNT	MQLONG	Počet pokusů o opětovné zpracování nezdařené zprávy příkazu pod bodem synchronizace	
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

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)

Selektor	Délka pole	Popis	Poznámka
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
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 klíče SSL	
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	NEz/OS
MQIA_STATISTICS_CHANNEL	MQLONG	Ovládá shromažďování statistických dat pro kanály	NEz/OS
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

Tabulka 570. Selektory atributů produktu MQINQ pro správce front (pokračování)			
Selektor	**Délka pole**	**Popis**	**Poznámka**
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 ×*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 napsaným v C nebo S/390 sestavovacím programem null.

CharAttrDélka
Typ: MQLONG -vstup

Toto je délka v bajtech parametru *CharAttrs* .

Hodnota CharAttrLength musí být alespoň součtem délek požadovaných znakových atributů (viz *Selectors*). Nula je platná hodnota.

CharAttrs
Typ: MQCHAR × *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 *Selectors*) a hodnota v něm je vyplněna doprava s 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 napsaným v C nebo S/390 sestavovacím programem null.

CompCode
Typ: MQLONG -výstup

Kód dokončení:

MQCC_OK
Úspěšné dokončení.

MQCC_WARNING
Varování (částečné dokončení).

MQCC_FAILED
Volání se nezdařilo.

reason
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 zamítnutý produktem 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 Kódy příčiny .

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 573 na stránce 789 .

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 787
   - “Atributy pro seznamy názvů” na stránce 818
   - “Atributy pro definice procesu” na stránce 820
   - “Atributy správce front” na stránce 753

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 WebSphere MQ . Například "MQTopicString" je spíše vrácený název než "mqps.Top"
2. Vlastnosti s předponou "jms." nebo "mcd." jsou vráceny 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 .

type
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.

reason
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 Kódy příčiny.

Vyvolání jazyka C

```
MQINQMP (Hconn, Hmsg, &InqPropOpts, &Name, &PropDesc, &Type,
ValueLength, Value, &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN Hconn;       /* Connection handle */
MQHMSG  Hmsg;        /* Message handle */
MQIMPO InqPropOpts; /* Options that control the action of MQINQMP */
MQCHARV Name;        /* Property name */
MQPD    PropDesc;    /* Property descriptor */
MQLONG  Type;        /* Property data type */
MQLONG  ValueLength; /* Length in bytes of the Value area */
MQBYTE  Value[n];    /* Area to contain the property value */
MQLONG  DataLength;  /* Length of the property value */
MQLONG  CompCode;    /* Completion code */
MQLONG  Reason;      /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
CALL 'MQINQMP' USING HCONN, HMSG, INQMSGOPTS, NAME, PROPDESC, TYPE,
VALUELENGTH, VALUE, DATALENGTH, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   Connection handle
 01  HCONN       PIC S9(9) BINARY.
**   Message handle
 01  HMSG            PIC S9(18) BINARY.
**   Options that control the action of MQINQMP
 01  INQMSGOPTS.
     COPY CMQIMPOV.
**   Property name
 01  NAME.
     COPY CMQCHRVV.
**   Property descriptor
 01  PROPDESC.
     COPY CMQPDV.
**   Property data type
 01  TYPE        PIC S9(9) BINARY.
**   Length in bytes of the VALUE area
 01  VALUELENGTH PIC S9(9) BINARY.
**   Area to contain the property value
 01  VALUE       PIC X(n).
**   Length of the property value
 01  DATALENGTH  PIC S9(9) BINARY.
**   Completion code
 01  COMPCODE    PIC S9(9) BINARY.
**   Reason code qualifying COMPCODE
 01  REASON      PIC S9(9) BINARY.
```

Vyvolání PL/I

```
call MQINQMP (Hconn, Hmsg, InqPropOpts, Name, PropDesc, Type,
ValueLength, Value, DataLength, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl InqPropOpts like MQIMPO;    /* Options that control the action of MQINQMP */
```

```
dcl Name        like MQCHARV;   /* Property name */
dcl PropDesc    like MQPD;      /* Property descriptor */
dcl Type        fixed bin (31); /* Property data type */
dcl ValueLength fixed bin (31); /* Length in bytes of the Value area */
dcl Value       char (n);       /* Area to contain the property value */
dcl DataLength  fixed bin (31); /* Length of the property value */
dcl CompCode    fixed bin (31); /* Completion code */
dcl Reason      fixed bin (31); /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
CALL MQINQMP,(HCONN,HMSG,INQMSGOPTS,NAME,PROPDESC,TYPE,
VALUELENGTH,VALUE,DATALENGTH,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN        DS         F     Connection handle
HMSG         DS         D     Message handle
INQMSGOPTS   CMQIMPOA   ,     Options that control the action of MQINQMP
NAME         CMQCHRVA   ,     Property name
PROPDESC     CMQPDA     ,     Property descriptor
TYPE         DS         F     Property data type
VALUELENGTH  DS         F     Length in bytes of the VALUE area
VALUE        DS         CL(n) Area to contain the property value
DATALENGTH   DS         F     Length of the property value
COMPCODE     DS         F     Completion code
REASON       DS         F     Reason code qualifying COMPCODE
```

MQMHBUF-Převést popisovač zprávy do vyrovnávací paměti

Volání MQMHBUF převádí 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-Popisovač zpráv pro volby vyrovnávací paměti” na stránce 438.

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.

reason
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 Kódy příčiny.

Vyvolání jazyka C

```
MQMHBUF (Hconn, Hmsg, &MsgHBufOpts, &Name, &MsgDesc, BufferLength, Buffer,
         &DataLength, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN Hconn;        /* Connection handle */
MQHMSG  Hmsg;         /* Message handle */
MQMHBO  MsgHBufOpts;  /* Options that control the action of MQMHBUF */
MQCHARV Name;         /* Property name */
MQMD    MsgDesc;      /* Message descriptor */
MQLONG  BufferLength; /* Length in bytes of the Buffer area */
MQBYTE  Buffer[n];    /* Area to contain the properties */
MQLONG  DataLength;   /* Length of the properties */
MQLONG  CompCode;     /* Completion code */
MQLONG  Reason;       /* Reason code qualifying CompCode */
```

Poznámky k použití

MQMHBUF převádí 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.

V produktu z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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-Deskriptor objektu” na stránce 440 .

Je-li pole *ObjectName* v parametru *ObjDesc* název modelové fronty, je dynamická lokální frontaje 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 787). 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. Je-li požadována více než jedna volba, mohou být tyto hodnoty:

- Přidáno společně (nepřidávejte stejnou konstantu více než jednou), nebo
- zkombinovat pomocí bitové operace OR (jestliže 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* . V následující tabulce jsou uvedeny platné volby MQOPEN pro dotazy 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

Volba	Alias [1]	Lokální a model	Vzdálený	Nelokální klastr	Distribuční seznam	Téma
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

Poznámka:

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 787 .

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. Je to 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, 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.

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. Je 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.

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 různých trasách. 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 vyžadují 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 460.

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 MQOH_READAHEAD klient WebSphere MQ povolí čtení napřed, pokud jsou splněny určité podmínky. Tyto podmínky zahrnují:

- Jak klient, tak i vzdálený správce front musí používat produkt WebSphere MQ verze 7 nebo novější.
- Klientská aplikace musí být zkompilována a propojena s knihovnami klienta WebSphere MQ MQI s podporou podprocesů.
- 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.
- Není-li aplikace spuštěna jako klient IBM WebSphere 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.

U z/OSv případě aplikace CICS nebo IMS tato volba také vynutí selhání volání MQOPEN, je-li připojení ve stavu uvedení do klidového stavu.

Tato volba je platná pro všechny typy objektů.

Informace o kanálech klienta naleznete v tématu Přehled klientů produktu IBM WebSphere MQ MQI .

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 WebSphere 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.

reason
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 systémem 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 tématech:

- Kódy příčiny pro všechny ostatní platformy IBM WebSphere MQ kromě platformy z/OS.

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 689.

     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
   - (pouzez/OS) Rozlišení názvu vzdáleného správce front na název sdílené přenosové fronty použité agentem IGQ (použije se pouze tehdy, pokud lokální a vzdálení správci 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 -Spouštěcí zpráva 2 (znakový formát)” na stránce 564 .
6. V produktu IBM ijsou aplikace spuštěné v režimu kompatibility automaticky připojeny ke správci front prvním voláním MQOPEN vydaným aplikací (pokud aplikace již není k danému správci front připojena pomocí volání MQCONN).

   Aplikace, které nejsou spuštěné v režimu kompatibility, musí před použitím volání MQOPEN k otevření objektu explicitně zadat volání MQCONN nebo MQCONNX pro připojení ke správci front.

Volby dopředného čtení

Při volání MQOPEN s MQOH_READAHEAD klient WebSphere MQ povolí čtení napřed, pokud jsou splněny určité podmínky. Tyto podmínky zahrnují:

- Jak klient, tak i vzdálený správce front musí používat produkt WebSphere MQ verze 7 nebo novější.
- Klientská aplikace musí být zkompilována a propojena s knihovnami klienta WebSphere MQ MQI s podporou podprocesů.
- 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 WebSphere 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, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti MQI IBM WebSphere MQ 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 manipulátorů (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-Deskriptor objektu” na stránce 440 .

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.

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 je výsledkem této definice jako alias fronty 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 alias správce front.
    - (pouzez/OS) Změna na hodnotu 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 *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* 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.

  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

```
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.

V systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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 zprávy" na stránce 383.

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 431.

Aplikace nemusí poskytovat strukturu MQMD, je-li v polích *OriginalMsgManipulátor* nebo *NewMsgManipulátor* 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 460.

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 výše uvedené podmínky nejsou splněny, *BufferLength* nemůže překročit menší z atributů *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 má 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 struktury záhlaví produktu WebSphere 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* nula, *Buffer* není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assemblerem 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.

reason
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_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 systémem 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 Kódy příčiny.

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.

- 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.

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:

     – V systému z/OSje aplikace následující:

       - Pro CICS, úloha CICS
       - Pro IMS, úloha
       - Pro dávku z/OS úloha

     – V systému IBM ije aplikací úloha.

     – V systémech Windows a UNIX je aplikací 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: AIX, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ MQI, kteří jsou připojeni 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 *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 produktu WebSphere MQ na začátku dat zprávy aplikace, provede správce front určité kontroly struktury záhlaví za účelem ověření, 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 *nejsou* platné ve většině struktur záhlaví WebSphere 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 produktu WebSphere MQ nesmí být zkrácena, 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.)
- Struktura produktu WebSphere 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

```
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.

V systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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-Deskriptor objektu” na stránce 440.

Je-li struktura fronta, uživatel musí mít autorizaci k otevření fronty pro výstup. Fronta **nesmí** být modelovou frontou.

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 zprávy” na stránce 383.

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 *Formát* v MQMD na MQFMT_MD_EXTENSION, abyste označili, že je přítomen MQMDE. Další informace viz část “MQMDE-Rozšíření deskriptoru zpráv” na stránce 431.

Aplikace nemusí poskytovat strukturu MQMD, je-li v poli *MsgHandle* struktury MQGMO nebo v polích *OriginalMsgManipulátor* nebo *NewMsgManipulátor* 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 460.

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; další podrobnosti naleznete v popisu parametru *BufferLength* volání MQPUT.

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 struktury záhlaví WebSphere MQ), ale některé zprávy mohou vyžadovat striktnější 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 v kódování lokálního správce front (daného 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* nula, *Buffer* není v tomto případě označen; v tomto případě může být adresa parametru předávaná programům napsaným v C nebo System/390 assemblerem 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.

reason
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 systémem 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 produktu WebSphere 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 Kódy příčiny.

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, HP-UX, IBM i, Solaris, Linux, Windows, plus klienti WebSphere MQ , kteří jsou připojeni 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 produktu WebSphere MQ na začátku dat zprávy aplikace, správce front provede 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

```
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.

V systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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 programy zapsaným v C nebo System/390 assembler hodnoty 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 571. 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 v klientech AIX, HP-UX, IBM i, Solaris, Linux, Windowsa WebSphere MQ MQI připojených 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, na *IntAttrs* není odkazováno; v tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 assembler s hodnotou null.

CharAttrDélka
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: MQCHARxCharAttrLength -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 znakového atributu je pevná (viz *Selectors*). 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, na *CharAttrs* není odkazováno; v tomto případě může být adresa parametru předávaná programy napsanými v C nebo System/390 assembler s hodnotou null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
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_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 systémem 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 Kódy příčiny.

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 662 .

   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 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 787
- “Atributy pro seznamy názvů” na stránce 818
- “Atributy pro definice procesu” na stránce 820
- “Atributy správce front” na stránce 753

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

Pomocí volání MQSET nastavte nebo upravte vlastnost 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 .

type
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 podepsané celé číslo.*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 assembler s hodnotou null.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
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 Kódy příčiny.

Vyvolání jazyka C

```
MQSETMP (Hconn, Hmsg, &SetPropOpts, &Name, &PropDesc, Type,
ValueLength, &Value, &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;       /* Connection handle */
MQHMSG   Hmsg;        /* Message handle */
MQSMPO   SetPropOpts; /* Options that control the action of MQSETMP */
MQCHARV  Name;        /* Property name */
MQPD     PropDesc;    /* Property descriptor */
MQLONG   Type;        /* Property data type */
MQLONG   ValueLength; /* Length of property value in Value */
MQBYTE   Value[n];    /* Property value */
MQLONG   CompCode;    /* Completion code */
MQLONG   Reason;      /* Reason code qualifying CompCode */
```

Vyvolání COBOL

```
     CALL 'MQSETMP' USING HCONN, HMSG, SETMSGOPTS, NAME, PROPDESC, TYPE,
                          VALUELENGTH, VALUE, COMPCODE, REASON.
```

Deklarujte parametry následujícím způsobem:

```
**   Connection handle
 01  HCONN     PIC S9(9) BINARY.
**   Message handle
 01  HMSG    PIC S9(18) BINARY.
**   Options that control the action of MQSETMP
 01  SETMSGOPTS.
     COPY CMQSMPOV.
**   Property name
 01  NAME
    COPY CMQCHRVV.
**   Property descriptor
 01  PROPDESC.
     COPY CMQPDV.
**   Property data type
 01  TYPE          PIC S9(9) BINARY.
**   Length of property value in VALUE
 01  VALUELENGTH   PIC S9(9) BINARY.
**   Property value
 01  VALUE         PIC X(n).
**   Completion code
 01  COMPCODE  PIC S9(9) BINARY.
**   Reason code qualifying COMPCODE
 01  REASON    PIC S9(9) BINARY.
```

Vyvolání PL/I

```
call MQSETMP (Hconn, Hmsg, SetPropOpts, Name, PropDesc, Type, ValueLength,
              Value, CompCode, Reason);
```

Deklarujte parametry následujícím způsobem:

```
dcl Hconn       fixed bin(31);  /* Connection handle */
dcl Hmsg        fixed bin(63);  /* Message handle */
dcl SetPropOpts like MQSMPO;    /* Options that control the action of MQSETMP */
dcl Name        like MQCHARV;   /* Property name */
dcl PropDesc    like MQPD;      /* Property descriptor */
dcl Type        fixed bin(31);  /* Property data type */
dcl ValueLength fixed bin(31);  /* Length of property value in Value */
dcl Value       char(n);        /* Property value */
```

```
dcl CompCode    fixed bin(31);  /* Completion code */
dcl Reason      fixed bin(31);  /* Reason code qualifying CompCode */
```

Vyvolání High Level Assembler

```
         CALL MQSETMP,(HCONN,HMSG,SETMSGHOPTS,NAME,PROPDESC,TYPE,VALUELENGTH,
               VALUE,COMPCODE,REASON)
```

Deklarujte parametry následujícím způsobem:

```
HCONN        DS         F     Connection handle
HMSG         DS         D     Message handle
SETMSGOPTS   CMQSMPOA   ,     Options that control the action of MQSETMP
NAME         CMQCHRVA   ,     Property name
PROPDESC     CMQPDA     ,     Property descriptor
TYPE         DS         F     Property data type
VALUELENGTH  DS         F     Length of property value in VALUE
VALUE        DS         CL(n) Property value
COMPCODE     DS         F     Completion code
REASON       DS         F     Reason code qualifying COMPCODE
```

MQSTAT-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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility lze volání MQCONN vynechat a pro produkt *Hconn* je určena následující hodnota:

MQC_DEF_HCONN
Výchozí popisovač připojení.

type
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 vytváření sestav o stavu" na stránce 549.

CompCode
Typ: MQLONG-výstup

Kód dokončení; je to jeden z následujících:

MQCC_OK
Úspěšné dokončení.

SELHÁNÍ MQCC_FAILED
Volání se nezdařilo.

reason
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 Kódy příčiny.

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 asembleru

```
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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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 524.

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.

V tabulce je shrnuto použití tohoto parametru 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.
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.

Volby	*Hobj*	Popis
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 manipulátor nelze použít pro žádné jiné volání produktu WebSphere MQ než MQCLOSE nebo MQSUBRQ. Předání tohoto popisovače do jiných volání produktu WebSphere MQ má za následek MQRC_HODB_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

reason
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.

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 nesleduje syntaxi selektoru produktu WebSphere 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 nelze nalézt.

Podrobné informace o těchto kódech najdete v tématu Kódy příčiny.

Poznámky k použití

1. 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 524.
2. 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.
3. Pokud jako vstup nezadáte žádný objekt Hobj, volání MQSUB přidělí dva popisovače, popisovač objektu (Hobj) a popisovač odběru (Hsub).
4. 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.
5. 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.
6. 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.

Výstup MQSD z MQSUB	
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.

Výstup MQSD z MQSUB *(pokračování)*	
Název pole v MQSD	**Speciální aspekty.**
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.
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 systému z/OS pro aplikace CICS a v produktu IBM i pro aplikace spuštěné v režimu kompatibility 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.

Action
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 546 .

Nejsou-li vyžadovány žádné volby, programy napsané v C nebo S/390 assembler 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

reason
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 Kódy příčiny.

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

V této kolekci témat jsou uvedeny pouze ty objekty produktu WebSphere MQ , které mohou být předmětem volání funkce MQINQ, a poskytují podrobnosti o atributech, které lze požadovat, a selektorů, které mají být použity.

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`. Většina atributů správce front může být dotazovaná otevřením speciálního objektu MQOT_Q_MGR 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 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 viz Příkazy skriptu (MQSC) .

Tabulka 572. Atributy správce front.
Seznam atributů správce front s odkazy a krátkým popisem

Atribut	Popis
AccountingConnOverride	Přepsat nastavení evidence.
AccountingInterval	Jak často zapisovat intermediační evidenční záznamy.
ActivityConnOverride	Přepsat nastavení aktivity.
ActivityTrace	Ovládá shromažďování trasování aktivity aplikace WebSphere 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
ChannelMonitoring	Online monitorovací data pro kanály
ChannelStatistics	Řídí shromažďování statistických dat pro kanály.
ChinitAdapters	Počet podúloh adaptéru pro zpracování volání produktu WebSphere MQ .
ChinitDispatchers	Počet dispečerů, který má být použit pro inicializátor kanálu.
	Rezervováno 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

Tabulka 572. Atributy správce front.

Seznam atributů správce front s odkazy a krátkým popisem

(pokračování)

Atribut	Popis
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
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

Tabulka 572. Atributy správce front.
Seznam atributů správce front s odkazy a krátkým popisem
(pokračování)

Atribut	Popis
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 784	WebSphere MQ Advanced 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 SSL.
SSLFIPSRequired	Pro šifrování používejte pouze certifikované algoritmy FIPS.
SSLKeyRepository 1	Umístění úložiště klíčů SSL.
PočetSSLKeyResetCount	Počet obnovení klíče SSL.
SSLTasks 1	Počet podúloh serveru pro zpracování volání SSL.
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.
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 UNIX, Linux a 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 v systémech IBM i, Unixových systémech a 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 v systémech IBM i, Windows, UNIXa Linux .

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 systémech IBM i, Unixových systémech a Windows.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_ACTIVITY_CONN_OVERRRIDE s voláním MQINQ .

ActivityTrace (MQLONG)

Tato volba určuje kolekci trasování aktivity aplikace produktu WebSphere MQ MQI.

Hodnota je jedna z následujících možností:

MQMON_ON
Shromažďovat trasování aktivity aplikace WebSphere MQ MQI.

MQMON_OFF
Neshromažďovat trasování aktivity aplikace WebSphere 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 systémech IBM i, Unixových systémech a Windows.

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.

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 v systému 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)

Tato volba určuje, zda mají být generovány události mostu IMS .

Hodnota je jedna z následujících možností:

POVOLENÝ MQEVR_
Vygenerujte 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 v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_BRIDGE_EVENT s voláním MQINQ.

ChannelAutoDef (MQLONG)

Tento atribut řídí automatickou definici kanálů typu MQCHT_RECEIVER a MQCHT_SVRCONN. 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.

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris a Windows.

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 .

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris a Windows.

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 999 .

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windowsa z/OS. V systému z/OSse vztahuje pouze na odesílací kanály klastru a kanály 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)

To 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.

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).

Tento atribut je podporován pouze v systémech IBM i, v systémech UNIX a Windows.

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 při zpracování volání produktu WebSphere 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 WebSphere MQ . To je důležité pro trvalé zprávy. Méně adaptérů může být lepší pro přechodné zprávy.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_ADAPTERS s voláním MQINQ.

ChinitDispatchers (MQLONG)

Jedná se o počet dispečerů, 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 v systému 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 v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_AUTO_START s voláním MQINQ.

ChinitTraceTableSize (MQLONG)

Jedná se o velikost datového prostoru trasování inicializátoru kanálu (v MB).

Hodnota musí být v rozsahu 0 až 2048, s výchozí hodnotou 2.

Poznámka: Při použití rozsáhlých datových prostorů systému z/OS se ujistěte, že máte v systému dostatek pomocné paměti pro podporu jakýchkoli souvisejících aktivit stránkování operačního systému z/OS . Pravděpodobně bude potřeba také zvýšit velikost datových sad SYS1.DUMP.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_CHINIT_TRACE_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 vypnuto

MQMON_LOW
Pokud *ChannelMonitoring* není MQMON_NONE, monitorování je zapnuto 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 přepnuto se střední rychlostí shromažďování dat s omezeným účinkem na výkon systému.

MQMON_HIGH
Pokud *ChannelMonitoring* není MQMON_NONE, monitorování je zapnuto s vysokou rychlostí 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
Přepnout na shromažďování statistických dat pro automaticky definované kanály odesílatele klastru s nízkým poměrem shromažďování dat.

MQMON_MEDIUM
Přepnout na shromažďování statistických dat pro automaticky definované kanály odesílatele klastru se středním poměrem shromažďování dat.

MQMON_HIGH
Přepnout na 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).

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.

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windows a z/OS.

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.

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windows a z/OS.

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* .

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windows a z/OS.

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.
- Na systémech UNIX je 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)

Značí úroveň příkazů řízení systému podporovaných správcem front. Může jít o jednu z následujících hodnot:

MQCMDL_LEVEL_1
Úroveň 1 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- MQSeries for AIX verze 2, vydání 2
- MQSeries for
  - Verze 1, vydání 1.1
  - Verze 1, vydání 1.2
  - Verze 1, vydání 1.3
- MQSeries for OS/400
  - Verze 2, vydání 3
  - Verze 3, vydání 1
  - Verze 3, vydání 6
- MQSeries for Windows , verze 2, vydání 0

MQCMDL_LEVEL_101
MQSeries for Windows verze 2, vydání 0.1.

MQCMDL_LEVEL_110
MQSeries for Windows , verze 2, vydání 1.

MQCMDL_LEVEL_114
MQSeries for Version 1, vydání 1.4.

MQCMDL_LEVEL_120
MQSeries for Version 1, vydání 2.0.

MQCMDL_LEVEL_200
MQSeries for Windows NT verze 2 vydání 0.

MQCMDL_LEVEL_210
MQSeries for OS/390 , verze 2, vydání 1.0.

MQCMDL_LEVEL_220
Úroveň 220 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- MQSeries for AT & T GIS UNIX , verze 2, vydání 2
- MQSeries for SINIX a DC/OSx, verze 2, vydání 2
- MQSeries for SunOS verze 2 vydání 2
- MQSeries for Tandem NonStop Kernel, verze 2, vydání 2

MQCMDL_LEVEL_221
Úroveň 221 řídicích příkazů systému.

Tato hodnota je vrácena produktem MQSeries for AIX verze 2, vydání 2.1

MQCMDL_LEVEL_320
Úroveň 320 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- MQSeries for OS/400
  - Verze 3, vydání 2
  - Verze 3, vydání 7

MQCMDL_LEVEL_420
Úroveň 420 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- MQSeries for IBM i
  - Verze 4, vydání 2.0
  - Verze 4, vydání 2.1

MQCMDL_LEVEL_500
Úroveň 500 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX verze 5, vydání 0
- MQSeries for HP-UX verze 5, vydání 0
- MQSeries for Solaris verze 5 vydání 0
- MQSeries for Windows NT verze 5 vydání 0

MQCMDL_LEVEL_510
Úroveň 510 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX verze 5, vydání 1
- MQSeries for AS/400 verze 5 vydání 1
- MQSeries for HP-UX verze 5 vydání 1
- IBM WebSphere MQ for HP Integrity NonStop Server verze 5, vydání 3
- MQSeries for Compaq Tru64 UNIX , verze 5, vydání 1
- MQSeries for Solaris verze 5 vydání 1
- MQSeries for Windows NT verze 5 vydání 1

MQCMDL_LEVEL_520
Úroveň 520 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- MQSeries for AIX ersion 5, vydání 2
- MQSeries for AS/400 verze 5, vydání 2
- MQSeries for HP-UX verze 5, vydání 2
- MQSeries for Linux verze 5, vydání 2
- MQSeries for OS/390 verze 5 vydání 2
- MQSeries for Sun Solaris verze 5, vydání 2
- MQSeries for Windows NT verze 5 vydání 2

MQCMDL_LEVEL_530
Úroveň 530 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX verze 5 vydání 3
- IBM WebSphere MQ for HP-UX verze 5 vydání 3
- IBM WebSphere MQ for i/Series verze 5, vydání 3
- IBM WebSphere MQ for Linux for Intel verze 5, vydání 3
- IBM WebSphere MQ for Linux pro zSeries verze 5, vydání 3
- IBM WebSphere MQ for Solaris verze 5 vydání 3
- IBM WebSphere MQ for Windows verze 5 vydání 3
- IBM WebSphere MQ for z/OS verze 5 vydání 3

MQCMDL_LEVEL_600
Úroveň 600 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 6.0
- IBM WebSphere MQ for HP-UX Version 6.0
- IBM WebSphere MQ for i/Series Version 6.0
- IBM WebSphere MQ for Linux Version 6.0
- IBM WebSphere MQ for Solaris Version 6.0
- IBM WebSphere MQ for Windows Version 6.0
- IBM WebSphere MQ for z/OS Version 6.0

MQCMDL_LEVEL_700
Úroveň 700 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 7.0
- IBM WebSphere MQ for HP-UX Version 7.0
- IBM WebSphere MQ for IBM i Version 7.0
- IBM WebSphere MQ for Linux Version 7.0
- IBM WebSphere MQ for Solaris Version 7.0
- IBM WebSphere MQ for Windows Version 7.0
- IBM WebSphere MQ for z/OS Version 7.0

MQCMDL_LEVEL_701
Úroveň 701 řídicích příkazů systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 7.0.1
- IBM WebSphere MQ for HP-UX Version 7.0.1

- IBM WebSphere MQ for IBM i Version 7.0.1
- IBM WebSphere MQ for Linux Version 7.0.1
- IBM WebSphere MQ for Solaris Version 7.0.1
- IBM WebSphere MQ for Windows Version 7.0.1
- IBM WebSphere MQ for z/OS Version 7.0.1

MQCMDL_LEVEL_710
Úroveň 710 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 7.1
- IBM WebSphere MQ for HP-UX Version 7.1
- IBM WebSphere MQ for IBM i Version 7.1
- IBM WebSphere MQ for Linux Version 7.1
- IBM WebSphere MQ for Solaris Version 7.1
- IBM WebSphere MQ for Windows Version 7.1
- IBM WebSphere MQ for z/OS Version 7.1

MQCMDL_LEVEL_750
Úroveň 750 příkazů pro řízení systému.

Tato hodnota je vrácena v následujících verzích produktu IBM WebSphere MQ:

- IBM WebSphere MQ for AIX Version 7.5
- IBM WebSphere MQ for HP-UX Version 7.5
- IBM WebSphere MQ for IBM i Version 7.5
- IBM WebSphere MQ for Linux Version 7.5
- IBM WebSphere MQ for Solaris Version 7.5
- IBM WebSphere MQ for Windows Version 7.5

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 následující:

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 podporován v systému z/OS.

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 následující:

MQEV_DISABLED
Vytváření sestav událostí je zakázáno.

POVOLENÝ MQEVR_
Vytváření sestav událostí je povoleno.

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 předali jejich dobu platnosti (viz MQMD-Expiry field), **nejsou** přeneseny 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 "Záhlaví MQDLH-Dead-letter" na stránce 319 .

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. Všechny správce front produktu WebSphere MQ podporují frontu nedoručených zpráv (nedoručená zpráva), avšak standardně 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 `DefClusterXmitQueueType` určuje, která přenosová fronta je standardně vybrána kanály odesílatele klastru k získání zpráv z kanálů příjemce klastru k odeslání zpráv do kanálů příjemce klastru.

Hodnoty atributu `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 Version 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.

Atribut není v produktu z/OSpodporován.

Je-li atribut správce front DefClusterXmitQueueTypenastaven na hodnotu CHANNEL, se výchozí konfigurace změní 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 662
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)

Jedná se o název skupiny pro modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, které se mají připojit při použití podpory služeb DNS (Dynamic Domain Name Services) správce pracovní zátěže. Maximální délka je 18 znaků. Ponecháte-li tento název prázdný, bude použit název skupiny sdílení front.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_DNS_GROUP s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_DNS_GROUP_NAME_LENGTH.

DNSWLM (MQLONG)

Určuje, zda se má modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registrovat ve správci pracovní zátěže pro služby DNS (Dynamic Domain Name Services)

Hodnota je jedna z následujících možností:

MQDNSWLM_YES
Modul listener se registruje se správcem pracovní zátěže.

MQDNSWLM_NO
Modul listener se neregistruje se správcem pracovní zátěže. Toto je výchozí hodnota.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DNS_WLM s voláním MQINQ.

ExpiryInterval (MQLONG)

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.

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.

Tento atribut je podporován pouze v systému z/OS.

IGQUserId (MQLONG)

Tento atribut lze použít pouze v případě, že je lokální správce front členem skupiny sdílení front. Určuje identifikátor uživatele, který je přidružen k lokálnímu 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.

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.

IntraGroupřazení do fronty (MQLONG)

Tento atribut se použije pouze v případě, že lokální správce front je členem skupiny sdílení front. Označuje, zda je pro skupinu sdílení front povoleno ukládání do front 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, je-li 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_QUEUE s voláním MQINQ.

Tento atribut je podporován pouze v systému z/OS.

IPAddressVersion (MQLONG)

Určuje, která verze adresy IP má být použita buď IPv4 , nebo IPv6.

Tento atribut je relevantní pouze pro systémy, které spouštějí jak IPv4 , tak IPv6 , a ovlivňují pouze kanály definované jako *TransportType* MQXPY_TCP, pokud je jedna z následujících podmínek pravdivá:

- *ConnectionName* kanálu je název hostitele, který se interpretuje jak na adresu IPv4 , tak na adresu IPv6 a jeho parametr *LocalAddress* není zadán.
- *ConnectionName* a *LocalAddress* kanálu jsou názvy hostitelů, které se interpretují jak na adresy IPv4 , tak i na adresy IPv6 .

Hodnota může být následující:

MQIPADDR_IPV4
Používá se IPv4 .

MQIPADDR_IPV6
Použije se IPv6 .

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 produktem WebSphere 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 v systému 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 systémech AIX, HP-UX, IBM i, Linux, Solaris a Windows.

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 v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU_GROUP_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_LU_NAME_LENGTH.

Název 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 v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_LU_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_LU_NAME_LENGTH.

LU62ARMSuffix (MQCHAR2)

Jedná se o příponu SYS1.PARMLIB člen APPCPMxx, který nominuje LUADD pro tento inicializátor kanálu. Příkaz z/OS SET APPC=xx se vydá, když ARM restartuje inicializátor kanálu. Ponecháte-li toto jméno prázdné, nebude vydáno žádné SET APPC=xx.

Tento atribut je podporován pouze v systému 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 v systému 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 operační systém z/OSmusí být hodnota v rozsahu od 1 do 9 999. Pro všechny ostatní platformy musí být hodnota v rozsahu od 1 do 65 535.

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 operační systém z/OSmusí být hodnota v rozsahu od 1 do 9 999, přičemž výchozí hodnota je 200. U všech ostatních platforem musí být hodnota v rozsahu od 1 do 65 535, s výchozí hodnotou 100. 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 platformy jiné než z/OSje hodnota parametru MaxChannels nastavena v souboru qm.ini příslušných správců front.

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 systému 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 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.

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.

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 *nejsou* počítány jako nepotvrzené zprávy:

- 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 systémech IBM i, v systémech UNIX a Windows.

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 systémech IBM i, UNIX and Linux a Windows.

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 v systému 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 v systému 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).

MQPL_MVS
z/OS (stejná hodnota jako MQPL_ZOS).

MQPL_NSK
HP Integrity NonStop Server.

MQPL_OS390
z/OS (stejná hodnota jako MQPL_ZOS).

MQPL_OS400
IBM i.

MQPL_UNIX
Systémy UNIX .

POČ MQPL_WINDOWS_NT
Systémy 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 produktu WebSphere MQ V6.0, počáteční hodnota tohoto atributu závisí na hodnotách DiscardNonPersistentResponse a DLQNonPersistentOdezva před migrací, jak je uvedeno 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 verzí produktu WebSphere Message Broker V6 nebo dřívějšími verzemi pomocí tohoto správce front, protože musí číst stejné fronty, z nichž 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, HP-UX, z/OS, IBM i, Solaris, Linux, Okna, plus klienti WebSphere MQ připojené 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 stejný jako název subsystému, který je omezen na 4 neprázdné znaky.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_Q_MGR_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_Q_MGR_NAME_LENGTH.

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 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.

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 v systému 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 v systému 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 v systému 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.

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windowsa z/OS.

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.

Tento atribut je podporován pouze v systémech AIX, HP-UX, IBM i, Linux, Solaris, Windowsa z/OS.

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)* .

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 ve skupině sdílení front.

Hodnota může být následující:

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 v systému z/OS.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SHARED_Q_MGR_NAME s voláním MQINQ.

SPLCAP
Určuje, zda jsou k dispozici funkce zabezpečení produktu WebSphere MQ Advanced Message Security pro správce front.

PODPOROVANÁ MQCAP_
Jedná se o výchozí hodnotu, pokud je komponenta WebSphere MQ AMS nainstalována pro instalaci, pod kterou je spuštěn správce front.

PODPOROVÁNO MQCAP_NOT_SUPPORTED

SSLEvent (MQLONG)
Uvádí, zda jsou generovány události SSL.

Je to jedna z následujících hodnot:

POVOLENÝ MQEVR_
Vygenerujte události SSL následujícím způsobem:

MQRC_CHANNEL_SSL_ERROR

MQEV_DISABLED
Negenerovat události SSL; jedná se o výchozí hodnotu.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_EVENT s voláním MQINQ.

Vyžadováno SSLFIPSRequired (MQLONG)

To umožňuje určit, že mají být použity pouze algoritmy certifikované podle standardu FIPS, pokud je šifrování prováděno v produktu WebSphere MQ, nikoli v kryptografickém hardwaru. Je-li konfigurován kryptografický hardware, použité šifrovací moduly jsou moduly poskytované hardwarovým produktem; tyto moduly mohou nebo nemusí být FIPS certifikovány na konkrétní úroveň v závislosti na použitém hardwarovém produktu.

Hodnota je jedna z následujících hodnot:

MQSSL_FIPS_NO
Použijte jakoukoli volbu CipherSpec podporovanou na platformě, kterou používáte. Tato hodnota je výchozí hodnotou.

MQSSL_FIPS_YES
Ve všech připojeních SSL mezi správcem front a tímto správcem front používejte pouze šifrovací algoritmy certifikované podle standardu FIPS v rámci CipherSpecs .

Tento parametr je platný pouze na platformách UNIX, Linux, Windowsa z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_SSL_FIPS_REQUIRED s voláním MQINQ.

Související úlohy
Určení, že pro běhové prostředí klienta MQI je použit pouze certifikovaný standard FIPS CipherSpecs
Související odkazy
Federální standardy zpracování informací (FIPS) pro UNIX, Linux a Windows

Počet SSLKeyResetCount (MQLONG)

Uvádí, kdy agenti kanálu zpráv kanálu SSL (MCS), které zahájí komunikaci, resetují tajný klíč, který se používá 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. Pokud zadáte počet obnovení tajných klíčů SSL/TLS v rozsahu od 1 bajtu do 32 KB, kanály SSL/TLS použijí počet obnovení tajných klíčů 32 kB. Tím se vyhnete nákladům na zpracování nadměrných resetů klíčů, které by se vyskytly u malých hodnot resetu tajného klíče protokolu SSL/TLS.

Tajný klíč je znovu vyjednán, když celkový počet nezašifrovaných bajtů odeslaných a přijatých inicializačním kanálem MCA překročí uvedenou hodnotu, nebo pokud jsou prezenční signály kanálu povoleny, než jsou data odeslána nebo přijata po prezenčním signálu kanálu, 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 systému 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 v systému 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 v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_TCP_KEEP_ALIVE s voláním MQINQ.

TCPName (MQCHAR8)

Jedná se o název buď jediného nebo výchozího systému TCP/IP, který používáte, v závislosti na hodnotě TCPStackType. Výchozí hodnota je TCPIP.

Tento atribut je podporován pouze v systému z/OS .

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQCA_TCP_NAME s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_TCP_NAME_LENGTH.

TCPStackType (MQLONG)

Uvádí, zda může inicializátor kanálu použít pouze adresní prostor TCP/IP uvedený v TCPName, nebo se může volitelně připojit k jakékoli vybrané adrese TCP/IP

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 v systému 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 WebSphere 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 WebSphere MQ Telemetry .

Hodnota je jedna z následujících možností:

PODPOROVANÁ MQCAP_
: Jsou podporovány komponenty produktu WebSphere MQ Telemetry a jsou podporovány příkazy Telemetry.

PODPOROVÁNO MQCAP_NOT_SUPPORTED
: Komponenta webSphere MQ Telemetry není nainstalována.

Tento atribut je podporován pouze v systémech IBM i, Unixových systémech a Windows.

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. V systému z/OS můžete z ní vytvořit 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. Fronta je známá jako soukromá fronta v systému z/OS.

Sdílená fronta (pouzez/OS)
Fronta se nachází ve sdíleném úložišti, které je přístupné všem správcům front patřícím ke skupině 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.

S výjimkou produktu z/OSmůže správce front ukládat zprávy pro ostatní správce front v klastru do více přenosových front. Správce front můžete nakonfigurovat tak, aby ukládal zprávy do více přenosových front klastru, dvěma různými způsoby. Když nastavíte atribut správce front `DEFCLXQ` na `CHANNEL`, vytvoří se jiná přenosová fronta klastru z `SYSTEM.CLUSTER.TRANSMIT.MODEL.QUEUE` automaticky pro každý odesílací kanál 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.

Fronta klastru může být fronta, kterou sdílí členové skupiny sdílení front v produktu IBM WebSphere 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.

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 573 na stránce 789 a v následných tabulkách.

Tabulka 573 na stránce 789 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 tématu Příkazy MQSC (Script) .

Tabulka 573. Atributy pro fronty. Sloupce se používají následujícím způsobem:

- 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.

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
AlterationDate	Datum, kdy byla definice naposledy změněna	✓		✓	✓	
AlterationTime	Čas, kdy byla definice naposledy změněna	✓		✓	✓	
BackoutRequeueQName	Nadměrný název fronty vrácených zpráv	✓	✓			
BackoutThreshold	Práh vrácení	✓	✓			
BaseQName	Název fronty, na kterou se rozlišuje alias			✓		
CFStrucName	Název struktury prostředku Coupling Facility	✓	✓			
CLCHNAME	Názvy odesílacích kanálů klastru	✓	✓			
ClusterName	Název klastru, do kterého fronta patří	✓		✓	✓	✓

Tabulka 573. Atributy pro fronty. Sloupce se používají následujícím způsobem:

- 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.

(pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
ClusterNamelist	Název objektu seznamu názvů obsahujícího názvy klastrů, do kterých fronta patří	✓		✓	✓	
CLWLQueuePriority	Priorita fronty pracovní zátěže klastru	✓		✓	✓	✓
CLWLQueueRank	Hodnocení fronty pracovní zátěže klastru	✓		✓	✓	✓
CLWLUseQ	Použít vzdálenou frontu	✓				
CreationDate	Datum, kdy byla fronta vytvořena	✓				
CreationTime	Čas, kdy byla fronta vytvořena	✓				
CurrentQDepth	Aktuální délka fronty	✓				
DefaultPutResponse	Odezva výchozího umístění	✓	✓	✓	✓	
DefBind	Výchozí vazba	✓		✓	✓	✓
DefinitionType attribute	Typ definice fronty	✓	✓			
DefInputOpenOption	Výchozí volba otevření pro vstup	✓	✓			
DefPersistence	Výchozí trvalost zpráv	✓	✓	✓	✓	✓
DefPriority	Výchozí priorita zpráv	✓	✓	✓	✓	✓
DefReadAhead	Výchozí dopředné čtení	✓	✓	✓		
DistLists	Podpora seznamu distribuce	✓	✓			
HardenGetBackout	Zda se má udržovat přesný počet vrácení	✓	✓			
IndexType	Typ indexu	✓	✓			
InhibitGet	Zda jsou povoleny operace získání pro frontu	✓	✓	✓		
InhibitPut	Zda jsou povoleny operace vložení pro frontu	✓	✓	✓	✓	✓
InitiationQName	Název inicializační fronty	✓	✓			
MaxMsgLength	Maximální délka zprávy v bajtech	✓	✓			
MaxQDepth	Maximální hloubka fronty	✓	✓			
MsgDeliverySequence attribute	Pořadí doručení zpráv	✓	✓			
NonPersistentMessage Class	Cíl spolehlivosti pro netrvalé zprávy	✓	✓			
OpenInputCount	Počet otevření pro vstup	✓				
OpenOutputCount	Počet otevření pro výstup	✓				
PropertyControl	Řízení vlastností	✓	✓	✓		

Tabulka 573. Atributy pro fronty. Sloupce se používají následujícím způsobem:

- 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.

(pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
ProcessName	Název procesu	✓	✓			
QDepthHighEvent attribute	Zda se generují události vysoké hloubky fronty	✓	✓			
QDepthHighLimit	Horní mez hloubky fronty	✓	✓			
QDepthLowEvent attribute	Zda se generují události nízké hloubky fronty	✓	✓			
QDepthLowLimit attribute	Dolní mez hloubky fronty	✓	✓			
QDepthMaxEvent	Zda se generují události zaplnění fronty	✓	✓			
QDesc	Popis fronty	✓	✓	✓	✓	✓
QName	Název fronty	✓		✓	✓	✓
QServiceInterval	Cíl pro interval služby fronty	✓	✓			
QServiceIntervalEvent attribute	Zda se generují události servisního intervalu vysoké nebo servisní interval OK	✓	✓			
QSGDisp attribute	Dispozice skupiny sdílení front	✓		✓	✓	
QueueAccounting	Shromažďování dat evidence front	✓	✓	✓	✓	✓
QueueMonitoring	Online monitorování dat pro fronty	✓	✓			
QueueStatistics	Kolekce statistických dat fronty	✓	✓	✓	✓	✓
QType	Typ fronty	✓		✓	✓	✓
RemoteQMgrName	Název vzdáleného správce front				✓	
RemoteQName	Název vzdálené fronty				✓	
RetentionInterval	Interval uchování	✓	✓			
Scope	Zda položka pro frontu také existuje v adresáři buňky	✓		✓	✓	
Shareability	Možnost sdílení front	✓	✓			
StorageClass	Paměťová třída pro frontu	✓	✓			
TriggerControl	Řízení spouštěče	✓	✓			
TriggerData	Data spouątěče	✓	✓			
TriggerDepth	Hloubka spouątěče	✓	✓			
TriggerMsgPriority	Prahová hodnota priority zpráv pro spouštěče	✓	✓			
TriggerType	Typ spouątěče	✓	✓			
Usage attribute	Použití fronty	✓	✓			

Tabulka 573. Atributy pro fronty. Sloupce se používají následujícím způsobem:

- 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.

(pokračování)

Atribut	Popis	Lokální	Model	Alias	Vzdálený	Klastr
XmitQName	Jméno přenosové fronty				✓	

Související pojmy

Fronty klastru

Lokální fronty

AlterationDate (MQCHAR12)

Datum, kdy byla definice naposledy změněna.

Lokální	Model	Alias	Vzdálený	Klastr
X		X	X	

Toto je datum, kdy byla definice naposledy změněna. Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například 1992-09-23¬¬, kde ¬¬ představuje dva prázdné znaky).

Hodnoty určitých atributů (například *CurrentQDepth*) se mění 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.

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/OSse jedná o 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.

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace spuštěné na serveru WebSphere Application Server a v těch, které používají produkt WebSphere MQ Application Server Facilities, používají tento atribut k určení toho, kam by měly jít zprávy, které byly vráceny. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu.

Třídy WebSphere MQ pro rozhraní JMS používají tento atribut k určení toho, kam má být přenesena zpráva, která již byla vrácena, maximální počet opakování, který je určen 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.

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Aplikace spuštěné na serveru WebSphere Application Server a ty, které používají služby WebSphere MQ Application Server Facilities, budou tento atribut používat k určení, zda by měla být zpráva vrácena. U všech ostatních aplikací neprovádí správce front žádnou akci založenou na hodnotě atributu.

Třídy WebSphere MQ pro platformu JMS používají tento atribut k určení, kolikrát mají být před přenosem zprávy do fronty zadané atributem *BackoutRequeueQName* zprávy, které mají být vráceny, vráceny 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.

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.

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.

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Chcete-li získat úplný název struktury ve spojovacím zařízení, zadejte hodnotu atributu správce front *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.

Tento atribut je podporován pouze v systému z/OS.

Název ClusterChannel(MQCHAR20)

`ClusterChannelName` je generický název odesílacích kanálů klastru, které tuto frontu používají jako přenosovou frontu. Atribut uvádí, které odesílací kanály klastru budou z této přenosové fronty klastru posílat zprávy do přijímacího kanálu klastru. Hodnota `ClusterChannelName` není v systému z/OSpodporována.

Lokální	Model	Alias	Vzdálený	Klastr
✓	✓			

Výchozí konfigurace správce front pro všechny odesílací kanály klastru je odesílat zprávy 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`.

Zadáte-li hvězdičku "`*`" do volby `ClusterChannelName`, můžete přidružit přenosovou frontu k sadě odesílacích kanálů klastru. Hvězdička může být na začátku, na konci nebo kdekoli ve středu řetězce názvu klastru. `ClusterChannelName` je v délce omezen na 20 znaků: `MQ_CHANNEL_NAME_LENGTH`.

ClusterName (MQCHAR48)

Jedná se o název klastru, do kterého fronta patří.

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.

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.

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.

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.

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 MQCA_CLWL_USEQ s voláním MQINQ. Délka tohoto atributu je dána hodnotou MQ_CLWL_USEQ_LENGTH.

CreationDate (MQCHAR12)

Toto je datum vytvoření fronty.

Lokální	Model	Alias	Vzdálený	Klastr
X				

Formát data je YYYY-MM-DD, vyplněný dvěma koncovými mezerami, aby byla délka 12 bajtů (například 2013-09-23¬¬, kde ¬¬ představuje 2 prázdné znaky).

- V systému IBM ise 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.

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/OSse jedná o 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.

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.

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.

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 Version 7.1nová, nesmí se používat, pokud se některé z aplikací otevírající tuto frontu připojují k produktu IBM WebSphere MQ Version 7.0.1 nebo starším správcům front.

Chcete-li určit hodnotu tohoto atributu, použijte selektor MQIA_DEF_BIND s voláním MQINQ.

DefinitionType (MQLONG)
Označuje, jak byla fronta definována.

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-Zavření objektu" na stránce 607.

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-Zavření objektu" na stránce 607.

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.

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.

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.

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.

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.

Lokální	Model	Alias	Vzdálený	Klastr
Ano	Ano	Ano	Ano	Ano

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.

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 podporován v systému z/OS.

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.

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 systémech IBM i, systémech UNIX a Windowsje počet vrácení 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.

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
Správce front udržuje index, který používá tokeny zpráv ve frontě zpráv ve frontě pro použití s funkcemi správce pracovní zátěže (WLM) v systému 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 operačního systému z/OS , ale načítá zprávy pomocí tokenu zprávy jako kritérium 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 CFLEVEL (4).

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 574 na stránce 802 a Tabulka 575 na stránce 803.

Tabulka 574. 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:		

Tabulka 574. 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 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
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 575. 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:		

Tabulka 575. Navržené nebo vyžadované hodnoty typu indexu fronty, je-li zadáno MQGMO_LOGICAL_ORDER (pokračování)		
Kritéria výběru při volání MQGET	**Typ indexu pro nesdílenou frontu**	**Typ indexu pro sdílenou frontu**
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.

Tento atribut je podporován pouze v systému z/OS.

InhibitGet (MQLONG)

Tento ovládací prvek určuje, zda jsou povoleny operace get pro tuto frontu.

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.

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 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.

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.

Lokální	Model	Alias	Vzdálený	Klastr
X	X			

Vzhledem k tomu, že atribut fronty produktu *MaxMsgLength* lze nastavit nezávisle na atributu správce front *MaxMsgLength* , je nižší z těchto dvou hodnot skutečný horní limit délky nejdelší fyzické zprávy, která může být umístěna ve frontě.

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.

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 schopnost načítání zpráv; *všechny* zprávy ve frontě lze načíst pomocí volání MQGET.

Hodnota tohoto atributu je nula nebo větší. Horní mez je určena prostředím:

- V systému AIX, HP-UX, z/OS, Solaris, Linuxa Windowsnemůže hodnota překročit 999 999 999.
- 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)

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).

- Pokud má v systému z/OSfronta *IndexType* MQIT_GROUP_ID, určuje atribut *MsgDeliverySequence* pořadí, ve kterém se budou skupiny zpráv vracet do aplikace. 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 systému z/OSprodukt *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.

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.

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.

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. Pokud se jedná o sdílenou frontu, tento počet nezahrnuje otevření pro výstup, který byl proveden pro frontu v jiných správcích front v rámci skupiny 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.

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.

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. Toto je výchozí hodnota. Umožňuje aplikacím, které očekávají, že se vlastnosti týkající se platformy JMS budou nacházet v záhlaví MQRFH2 dat zprávy, nadále fungovat 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.

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 systémem 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.

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 systémem 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.

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 systémem 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.

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 systémem 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.

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 systémem z/OS, ale volání MQINQ nelze použít k určení její hodnoty.

QDesc (MQCHAR64)
Toto pole použijte pro popisný komentář.

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.

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.

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 systémem 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.

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.

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 v rámci skupiny 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.

Tento atribut je podporován pouze v systému z/OS.

QueueAccounting (MQLONG)

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.

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)

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
Přepnout na shromažďování statistických dat pro tuto frontu.

QType (MQLONG)

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)

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 *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)

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* prázdná.

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.

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.

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 se nerozšiřuje za 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/OStento atribut není podporován.

Sdílitelnost (MQLONG)

Označuje, zda lze frontu otevřít pro vstup vícenásobně souběžně.

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.

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.

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.

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.

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)

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).

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.

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.

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 přenosových frontách najdete v tématu Definování přenosové fronty.

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 je formát dat zprávy správný (viz “MQXQH-záhlaví přenosové fronty” na stránce 576), 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.

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 *DefXmitQName* .

Tento atribut je ignorován, je-li definice používána 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 WebSphere MQ a v klientech WebSphere MQ MQI 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 viz Příkazy skriptu (MQSC) .

Tabulka 576. 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.

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 WebSphere 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 queue-manager: Definice objektu je známá pouze lokálnímu správci front; definice není ostatním správcům front ve skupině sdílení front známa.

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.

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 viz Příkazy skriptu (MQSC) .

Tabulka 577. 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 produktem WebSphere MQ vyžaduje, aby byl produktem *ApplId* název spustitelného programu. Níže uvedené poznámky se vztahují na uvedená prostředí:

- V systému z/OSmusí produkt *ApplId* být:
  - Identifikátor transakce systému CICS pro aplikace spuštěné pomocí transakce monitoru 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.
- Na systémech UNIX může 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 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 jakoukoli hodnotu, ale následující hodnoty jsou doporučeny pro standardní typy; omezují uživatelem definované typy aplikací na hodnoty v rozsahu MQAT_USER_FIRST přes MQAT_USER_LAST:

MQAT_AIX
Aplikace AIX (stejná hodnota jako MQAT_UNIX).

MQAT_BATCH
aplikace pro dávkové úlohy

MQAT_BROKER
Aplikace zprostředkovatele

MQAT_CICS
Transakce CICS .

MOST MQAT_CICS_BRIDGE
Aplikace mostu CICS .

MQAT_CICS_VSE
Transakce CICS/VSE .

MQAT_DOS
Aplikace klienta WebSphere MQ MQI v systému PC DOS.

MQAT_IMS
Aplikace IMS .

MOST MQAT_IMS_BRIDGE
Aplikace mostu IMS .

MQAT_JAVA
Aplikace Java.

MQAT_MVS
Aplikace MVS nebo TSO (stejná hodnota jako MQAT_ZOS).

MQAT_NOTES_AGENT
Aplikace agenta Lotus Notes .

MQAT_NSK
HP Integrity NonStop Server .

MQAT_OS390
Aplikace OS/390 (stejná hodnota jako MQAT_ZOS).

MQAT_OS400
IBM i .

MQAT_RRS_BATCH
Dávková aplikace RRS.

MQAT_UNIX
Aplikace UNIX .

MQAT_UNKNOWN
Aplikace neznámého typu.

UŽIVATEL MQAT_USER
Uživatelská aplikace.

MQAT_VOS
Aplikace Stratus VOS.

MQAT_WINDOWS
16bitová aplikace systému Windows .

POČ MQAT_WINDOWS_NT
32bitovou aplikaci systému Windows .

MQAT_WLM
Aplikace správce pracovní zátěže systému z/OS .

MQAT_XCF
XCF.

MQAT_ZOS
Aplikace 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 WebSphere MQ připojí produkt *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 WebSphere MQ.
  - Je-li ApplType MQAT_WLM, můžete zadat výchozí hodnoty do polí EnvData pro pole ServiceName a ServiceStep v záhlaví pracovních informací (MQWIH).
- Na systémech UNIX lze *EnvData* nastavit na znak & , aby se spustila spuštěná aplikace 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 queue-manager: Definice objektu je známá pouze lokálnímu správci front; definice není ostatním správcům front ve skupině sdílení front známa.

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.

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 WebSphere MQ předává produkt *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. V systému Microsoft Windowsnesmí znakový řetězec obsahovat uvozovky, pokud se definice procesu předá do 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 WebSphere MQ Message Queue Interface (MQI) a WebSphere MQ Administration Interface (MQAI) se správcem front nebo uživatelskou procedurou vrací kód **completion** a kód ukončení **reason** , 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.

Kontrola aplikací po úspěšném dokončení po volání rozhraní API produktu WebSphere MQ musí vždy kontrolovat 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 Kódy příčiny .

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 uvedené výše se vzájemně 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 možnosti:
  - 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 uvedené výše se vzájemně 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 pouze v kombinaci s MQSO_CREATE.

- 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 828
- “Zrušit registraci zprávy odběratele” na stránce 829
- “Publikovat zprávu” na stránce 833
- “Registrovat zprávu odběratele” na stránce 835
- “Požadavek na aktualizaci zprávy” na stránce 840

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 841
- “Nastavení MQMD pro publikování přeposlané správcem front” na stránce 847
- “Nastavení MQMD ve zprávách odezvy správce front” na stránce 848
- “Kódy příčiny publikování a odběru” na stránce 843

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 zprávy příkazu správci front, naleznete v tématu “Nastavení MQMD pro publikování přeposlané správcem front” na stránce 847 .

Vlastnosti

< Command> (*MQPSC_COMMAND*)
Hodnota je DeletePub(*MQPSC_DELETE_PUBLIKACE*).

Tato vlastnost musí být uvedena.

< Topic> (*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

< Command> (*MQPSC_COMMAND*)
Hodnota je `DeregSub` (*MQPSC_DEREGISTER_SUBSCRIBER*).

Tato vlastnost musí být uvedena.

< Topic> (*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 parametrFullResp , 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.

<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

< Command> (***MQPSC_COMMAND***)
Hodnota je `Publikovat`(*MQPSC_PUBLISH*).

Tato vlastnost musí být uvedena.

< Topic> (***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 V6je hodnota vlastnosti <SubPoint> hodnota atributu Bod odběru u uzlu publikování, který zpracovává publikování.

V produktu WebSphere MQ V7.0.1se 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 jako odezva na příkazovou zprávu `Register Subscriber` nebo `Request Update` . 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 846 .

Je-li odpověď na frontu dočasnou dynamickou frontou, správce front při zavření fronty automaticky zruší registraci odběru.

Vlastnosti

< Command> (*MQPSC_COMMAND*)
Hodnota je RegSub (*MQPSC_REGISTER_SUBSCRIBER*). Tato vlastnost musí být uvedena.

< Topic> (*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 V6se hodnota vlastnosti <SubPoint> musí shodovat s hodnotou atributu Bod odběru u odebíraných uzlů publikování.

V produktu WebSphere MQ V7.0.1se 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.

Pokud má parametr `SubIdentity` také jednu z voleb sloučení (`JoinExcl` nebo `JoinShared`), je identita přidána do sady identit bez ohledu na to, zda je zadán parametr `NoAlter` .

NONE
(*MQPSC_NONE*)

Všechny volby registrace mají své výchozí hodnoty.

Pokud je odběratel již registrován, jeho volby jsou resetovány na jejich výchozí hodnoty (všimněte si, že toto *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.

Pokud se jedná o dočasnou dynamickou frontu, musí být v rámci vlastnosti <RegOpt> uvedeno dočasné doručení publikací (`NonPers`).

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 s atributy, jsou-li zadány v nové zprávě příkazu `Register Subscriber` , jsou-li zadány. 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** .

Pokusíte-li se změnit existující pojmenovaný odběr pomocí jiného příkazu `Register Subscriber` , který určuje stejnou hodnotu **SubName**a hodnoty tématu, QMgrName, QName a CorrelId v novém příkazu odpovídají odlišnému stávajícímu odběru, s definovaným nebo bez definovaného 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 `Register Subscriber` , ve které je zadán parametr **SubIdentity** , je úspěšný pouze tehdy, pokud 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*.

<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

< Command> (*MQPSC_COMMAND*)
Hodnota je ReqUpdate (*MQPSC_REQUEST_UPDATE*). Tato vlastnost musí být uvedena.

< Topic> (*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 pokusí odeslat odpověď do fronty nedoručených zpráv produktu WebSphere 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

< Completion> (*MQPSCR_COMPLETION*)
Kód dokončení, který může mít jednu ze tří hodnot:

OK
Příkaz byl úspěšně dokončen

varování
Příkaz byl dokončen, ale s varováním

Chyba
Příkaz selhal

< Response> (*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.

< Reason> (*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 WebSphere MQ cmqc.h . Konstanty MQRCCF_ vyžadují soubor záhlaví produktu WebSphere 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 společnosti 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` je nastavena na hodnotu v rámci zprávy `Publish` 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, do kterého se tato publikace odesílá, jinak určeno 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
Prioritaje 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 zprávy” na stránce 383 .

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[3]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 MQMD, příslušné konstanty, které popisují požadovaná kódování, mohou být:

- Přidáno společně nebo
- Kombinované s použitím 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

[3] Institute of Electrical and Electronics Engineers

- MQENC_DECIMAL_MASK pro kódování dekadického celého čísla
- 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. Rozdělte hodnotu pole *Encoding* hodnotou A; zavoláte výsledek B.
3. Dělí se B o 16; volejte výsledek C.
4. Multiply C od 16 a odečítat od B; volejte výsledek D.
5. Multiply D by 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 578 na stránce 851.

Tabulka 578. 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
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 zprávy” na stránce 383.

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 výše.

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 hodnoty 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 testování 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 popisu uvedeného 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. Vydělte pole *Report* hodnotou A; volejte výsledek B.
3. Dělí B podle 8; volejte výsledek C.
4. Multiply C od 8 a odečtěte od B; volejte výsledek D.
5. Multiply D by 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 výše popsaného postupu, ale nahrazením hodnoty 8 v krocích uvedených výše 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 výše.

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 WebSphere MQ* na adrese https://www.ibm.com/support/docview.wss?uid=swg27005729.

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 856.

- Konvence zpracování použité správcem front při zpracování vestavěného formátu; tyto konvence se doporučují také pro uživatelské procedury zápisu. Viz “Konvence zpracování” na stránce 857.
- Speciální pokyny pro převod zpráv sestav viz “Převod zpráv sestav” na stránce 861.
- Parametry předané uživatelské proceduře pro převod dat; viz “MQ_DATA_CONV_EXIT-Ukončení převodu dat” na stránce 873.
- Volání, které lze použít z uživatelské procedury pro převod znakových dat mezi různými reprezentacemi; viz “MQXCNVC-Převod znaků” na stránce 867.
- Parametr datové struktury, který je specifický pro uživatelskou proceduru, viz “MQDXP-Data-konverze-výstupní parametr” na stránce 862.

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:

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 volba MQGMO_ACCEPT_TRUNCATED_MSG *nebyla* zadaná, bude vrácena nekonvertovaný kód dokončení s kódem dokončení MQCC_WARNING a kódem 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:

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 volba MQGMO_ACCEPT_TRUNCATED_MSG *nebyla* zadaná, bude vrácena nekonvertovaný kód dokončení s kódem dokončení MQCC_WARNING a kódem 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 (před nebo během převodu), délka vrácená pro parametr *DataLength* 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.
   - 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*, *DestEnvOffset*a *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 pro umístění dvojic název/hodnota 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 UCS-2 je příkladem 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 skutečně vrácených 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 požadován převod *je* , vyvolá správce front uživatelskou proceduru s názvem zadaným polem *Format* ve struktuře MQDLH nebo provede vlastní převod (pokud je *Format* 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 možno určit, které z výše uvedených požadavků se mají 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 za použití struktury MQXQH jsou dány poli *Format*, *CodedCharSetId*a *Encoding* ve struktuře MQMD obsažené v *uvnitř* 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 *nejsou* data zprávy aplikace; 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 výše popsaných 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 výše).

Je-li při načítání zprávy sestavy uvedena volba MQGMO_CONVERT, postupujte takto:

- Pro případ 1 výše se uživatelská procedura konverze dat nevyvolá (protože zpráva hlášení neobsahuje žádná data).
- Pro případ 3 výše může název formátu správně implikuje délku dat zprávy.
- Ale ve výše uvedeném případu je 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 musí uživatelská procedura konverze dat *ne* používat název formátu k odvození délky dat, která mu byla předána; místo toho musí být ukončena délka poskytnutých dat a musí být připravena převést *menší* data, 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 určen atributem správce front *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. Avšak pole *DataLength nelze* změnit, pokud je převáděná zpráva segmentem, 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, protože uživatelská procedura vrátila MQXDR_CONVERSION_FAILED v produktu *ExitResponse*; správce front však ignoruje hodnoty polí *CompCode* a *Reason* vrácené uživatelskou procedurou v tomto případě a použije místo toho hodnoty, které měla tato pole na vstupu *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 579. 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. Avšak, tato délka *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, uživatelská procedura *musí* vrátit MQXDR_CONVERSION_FAILED v poli *ExitResponse* , aby mohl správce front vrátit nekonvertované zprávy. 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 (pouzeIBM 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 WebSphere MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce produktu WebSphere 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 *Hconn*zadat 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. Je-li požadováno více než jedno, hodnoty mohou být:

- sečíst (žádnou konstantu nepřičítejte vícekrát než jednou) nebo
- Kombinované s použitím 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, HP-UX, IBM i, Solaris, 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 UCS-2), ale *TargetLength* uvádí délku, která je lichý počet 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, HP-UX, IBM i, Solaris, Linux, Windows.

Volby kódování: Popsané volby lze použít k uvedení celočíselných kódování zdrojového a cílového řetězce. Relevantní kódování se používá *pouze* , když 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 UCS-2).

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, naleznete v části "Analyzování kódování" na stránce 850 .
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 vícebajtovou znakovou sadou se znázorněním, která je závislá na kódování (například znaková sada UCS-2), volání selže s kódem příčiny MQRC_SOURCE_INTEGRER_ENC_ERROR nebo MQRC_TARGET_INTEGRER_ENC_ERROR.

Volby kódování jsou podporovány v následujících prostředích: AIX, HP-UX, z/OS, IBM i, Solaris, Linux, Windows.

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

Toto je délka vstupního řetězce v *SourceBuffer*v bajtech; musí být nula nebo větší.

SourceBuffer
Typ: MQCHARxSourceDélka-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: MQCHARxTargetLength-output

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.

reason
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 Kódy příčiny.

Vyvolání jazyka C

```
MQXCNVC (Hconn, Options, SourceCCSID, SourceLength, SourceBuffer,
        TargetCCSID, TargetLength, TargetBuffer, &DataLength,
        &CompCode, &Reason);
```

Deklarujte parametry následujícím způsobem:

```
MQHCONN  Hconn;            /* Connection handle */
MQLONG   Options;          /* Options that control the action of
                              MQXCNVC */
MQLONG   SourceCCSID;      /* Coded character set identifier of string
                              before conversion */
MQLONG   SourceLength;     /* Length of string before conversion */
MQCHAR   SourceBuffer[n];  /* String to be converted */
MQLONG   TargetCCSID;      /* Coded character set identifier of string
                              after conversion */
MQLONG   TargetLength;     /* Length of output buffer */
MQCHAR   TargetBuffer[n];  /* String after conversion */
MQLONG   DataLength;       /* Length of output string */
MQLONG   CompCode;         /* Completion code */
MQLONG   Reason;           /* Reason code qualifying CompCode */
```

Deklarace COBOL (pouzeIBM 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 assembler 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 WebSphere MQ Data Conversion Interface (DCI), který je jedním z rozhraní rámce WebSphere 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 862 .

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* předávaný uživatelské proceduře je vždy nejvíce aktuální 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 MQMD version-2 : AIX, HP-UX, IBM i, Solaris, 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*; pokud je tento parametr změněn, výsledky nejsou definované.

V programovacím jazyku C je tento parametr definován jako ukazatel na hodnotu typu void.

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* z následujících podmínek 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 sestav (viz "Převod zpráv sestav" na stránce 861 , kde získáte další podrobnosti).

     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 parametru *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 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 *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 zpráva rozbalila *během* převodu, na místo, kde 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. Pro aplikace 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. Avšak `typedef` je poskytnuto pro název MQ_DATA_CONV_EXIT v programovacím jazyce C a lze jej použít k deklarování uživatelské procedury, aby se zajistilo, že parametry jsou 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. Je-li v systému z/OSv platnosti také uživatelská procedura překřížení rozhraní API, zavolá 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)

```
     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.

Tato skutečnost zachovává kompatibilitu s předchozími verzemi klientů WebSphere MQ JMS a XMS . Tento oddíl popisuje, jak určit vlastnosti v záhlaví MQRFH2 .

Chcete-li jako vlastnosti použít prvky MQRFH2 , zadejte prvky podle popisu v tématu Použití tříd produktu WebSphere MQ pro jazyk Java. Tyto informace doplňují informace popsané v části “MQRFH2 -Pravidla a formátovací záhlaví 2” na stránce 487.

Mapování datových typů vlastností na datové typy MQRFH2

Toto téma poskytuje informace o typech vlastností zpráv mapovaných na odpovídající datové typy MQRFH2 .

Tabulka 580. Podporované datové typy MQRFH2

Typ vlastnosti zprávy	**Datový typ MQRFH2**
MQBYTE []	bin.hex
MQBOOL	typ boolean
MQINT8	i1
MQINT16	i2
MQINT32	i4
MQINT64	i8
MQFLOAT32	r4
MQFLOAT64	r8
MQCHAR []	řetězec

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 libovolných vlastností definovaných aplikací JMS, které jsou přidruženy ke zprávě. Ve složce `<usr>` nejsou povoleny skupiny.

Záhlaví MQRFH2 a rozhraní JMS podporují následující další složky:

- <mq>

  Tato složka se používá a je vyhrazena pro vlastnosti definované produktem MQ, které jsou používány produktem IBM WebSphere MQ.

- <mq_usr>

  Tuto složku lze použít k přenosu libovolných vlastností definovaných aplikací, které nejsou vystaveny jako uživatelem definované vlastnosti JMS, protože vlastnosti nemusí splňovat požadavky na vlastnost 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 WebSphere MQ .

Produkt IBM WebSphere 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 vystaveny jako vlastnosti JMS nebo jsou mapovány na vlastnosti JMS_IBM_ *, ale jsou vystaveny aplikacím WAS/SIB; tyto vlastnosti zahrnují vlastnosti dopředného a zpětného směrování cest.

  Alespoň některé 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 platformy 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 WebSphere 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 uživatelským aplikacím WAS/SIB nebo jako vlastnosti 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; rezervovány pro použití produktem IBM WebSphere MQ.
- Všechny složky s předponou sib; jsou vyhrazeny pro použití produktem WAS/SIB.
- Složky <Root> a <Body> , vyhrazené, ale nepoužité.

Následující složky nejsou rozpoznány jako obsahující vlastnosti zprávy:

- <psc>

Používáno produktem WebSphere Message Broker k přenosu zpráv příkazů publikování/odběru do zprostředkovatele.

- <pscr>

  Používá produkt WebSphere Message Broker k ukládání informací ze zprostředkovatele v odezvě na zprávy příkazů publikování/odběru.

- Jakákoli složka není definována produktem WebSphere Message Broker, která není označena atributem content='properties' .

Nezadávejte content='properties' ve složkách <psc> nebo <pscr> . Pokud tak učiníte, tyto složky budou považovány za vlastnosti a WebSphere Message Broker 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 s definovanými vlastnostmi IBM, 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 WebSphere MQdefinované 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 WebSphere MQ.

Generování záhlaví MQRFH2

Pokud produkt WebSphere MQ převede vlastnosti zprávy na jejich znázornění MQRFH2 , musí do zprávy přidat MQRFH2 . Přidává MQRFH2 buď jako samostatné záhlaví, nebo ho sloučí s existujícím záhlavím.

Generování nových záhlaví produktu MQRFH2 pomocí produktu WebSphere MQ by mohlo 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. Produkt WebSphere MQ se pokusí minimalizovat dopad přidání vlastností na zprávu sloučením vlastností zprávy do existujícího záhlaví produktu MQRFH2 , kde to může být možné. 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 .

Pravidla pro slučování generovaných MQRFH2

Následující pravidla se vztahují ke sloučení vygenerovaného MQRFH2 s existujícím MQRFH2. Vygenerované záhlaví MQRFH2 je sloučeno s existujícím záhlavím MQRFH2 , pokud:

1. Existující produkt MQRFH2 se nachází ve stejné pozici WebSphere MQ by umístil generovaný MQRFH2nebo dřívější řetězec 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 existujícím produktu 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; názvy vlastností jsou vyhrazeny pro použití službou JMS a nejsou platné pro uživatelem definované vlastnosti.

  Takový název prvku nezpůsobí selhání analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu WebSphere 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 nezpůsobí selhání analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu WebSphere MQ .
- 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 nezpůsobí selhání analýzy MQRFH2 , ale není přístupný pro rozhraní API vlastností zprávy produktu WebSphere MQ .

Obecně platí, že záhlaví MQRFH2 , která obsahují platná data ve formátu XML, lze analyzovat produktem WebSphere MQ bez selhání, ačkoli některé prvky MQRFH2 nejsou přístupné prostřednictvím rozhraní API vlastností zprávy WebSphere 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 produktu WebSphere 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), název vlastnosti nemusí být nutně názvem 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 WebSphere MQ classes for JMS jsou již dokumentovány v produktu Použití jazyka Java.

Název vlastnosti	Název složky MQRFH2	Název prvku MQRFH2
JMSDestination	jms	Dst
JMSType	mcd	Type, Set, Fmt
xxx (definovaný uživatelem, kde xxx nezačíná na JMS)	usr	xxx

Proto, když aplikace platformy JMS přistupuje k vlastnosti JMSDestination , je tato mapa mapována na prvek Dst ve složce <jms> .

Při určování vlastností jako prvků MQRFH2 definuje produkt IBM WebSphere MQ své prvky následujícím způsobem:

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 WebSphere MQ JMS pokusí o přístup k vlastnosti produktu Property1 , bude se mapovat 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 .

Záhlaví JMS a vlastnosti specifické pro poskytovatele, které jsou obsaženy ve struktuře MQRFH2 ve složkách produktu <mcd>, <jms>a <mqext> , jsou přístupné prostřednictvím aplikací IBM WebSphere MQ pomocí krátkých názvů definovaných v tématu Použití tříd produktu WebSphere MQ pro jazyk Java.

Vlastnosti definované uživatelem platformy JMS jsou přístupné ze složky <usr> . Aplikace IBM WebSphere MQ může použít složku <usr> pro své vlastnosti aplikace, je-li přijatelné, aby se vlastnost zobrazovala pro aplikace platformy 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 jiné vlastnosti než JMS.

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 878 , 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 produktu IBM WebSphere MQ classes for JMS nebo klienta XMS , které mohou mít přístup pouze ke složce produktu <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 736 podle potřeby automaticky přidá tento atribut. Tento atribut nesmí být přidán do žádné z definovaných složek IBM, například <jms> a <usr>. Pokud tak učiníte, bude zpráva odmítnuta klientem produktu IBM WebSphere MQ classes for JMS před verzí 7.0. s `MessageFormatException`.

Protože složka <usr> je výchozí umístění vlastností syntaxe <Property> , aplikace IBM WebSphere MQ a aplikace JMS pro 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.

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.

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"**

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**.
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 z voleb produktu WebSphere MQ verze 7, takže existující aplikace nebudou selhávat.
- 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 selže s kódem příčiny MQRC_RFH_FORMAT_ERROR, pokud lze určit, že aplikace obsahuje některé volby produktu WebSphere MQ verze 7, takže existující aplikace nebudou selhávat.
- 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 736nebo 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, převod systému z/OS , převod systému IBM i a podporu konverze Unicode.

V každé národní jazykové sekci jsou uvedeny následující informace:

- Podporované nativní CCSID
- Převody kódových stránek, které **nejsou** podporovány

V informacích jsou použity následující termíny:

-8
Označuje pro HP-UX , že CCSID je pro HP-UX definovanou kódovou sadu *roman8*

AIX
Označuje produkt WebSphere MQ pro systém AIX

HP-UX
Označuje WebSphere MQ pro HP-UX

Linux
Označuje produkt WebSphere MQ for Linux for Intel a WebSphere MQ for Linux for zSeries

HP Integrity NonStop Server
Označuje WebSphere MQ pro HP Integrity NonStop Server

OS/400
Označuje WebSphere MQ pro IBM i

Solaris
Označuje WebSphere MQ pro Solaris

Windows
Označuje produkt WebSphere MQ for Windows

z/OS
Označuje WebSphere MQ pro 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. Převod informací o klientovi WebSphere MQ MQI se provádí na serveru, takže server musí podporovat převod z CCSID klienta na CCSID serveru.
2. Převod může zahrnovat podporu přidanou CSD/PTF na nejnovější verzi produktu WebSphere MQ. Zkontrolujte obsah nejnovější úrovně služeb a zjistěte, zda je třeba instalovat CSD/PTF pro povolení tohoto převodu.

Viz Tabulka 581 na stránce 886 , kde je křížový odkaz mezi některými čísly CCSID a některými názvy odvětvových kódových sad.

Názvy kódových sad a CCSID

Produkt WebSphere MQ for z/OS poskytuje více převodů, než je uvedeno v tabulkách specifických jazyků.

Tabulka 581. Názvy kódových sad a CCSID

Názvy kódové sady	CCSID
ISO 8859-1	819

Tabulka 581. Názvy kódových sad a CCSID (pokračování)	
Názvy kódové sady	**CCSID**
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 WebSphere MQ.

Jazyky podporované produktem WebSphere MQ jsou:

- Americká angličtina-viz téma “americká angličtina” na stránce 888
- Němčina-viz téma “Němčina” na stránce 888
- Dánština a norština-viz téma “Dánština a nor” na stránce 889
- Finština a švédština-viz téma “Finnish a” na stránce 890
- Italština-viz téma “italština” na stránce 891
- Španělština-viz téma “Španělština” na stránce 891
- Britská angličtina/Gaelština-viz téma “Britská angličtina /gaelština” na stránce 892
- Francouzština-viz téma “Francouzština” na stránce 892
- Vícejazyčná-viz téma “Vícejazyčné” na stránce 893
- Portugalština-viz téma “Portugalština” na stránce 893
- Islandština-viz téma “Islandština” na stránce 894
- Východní evropské jazyky-viz téma “Jazyky východní Evropy” na stránce 895
- Cyrilice-viz téma “Cyrilice” na stránce 896
- Estonština-viz téma “Estonština” na stránce 897
- Lotyšština a litevština-viz téma “Lotyšské a litevské” na stránce 898
- Ukranian-viz téma “Ukrajinština” na stránce 899

- Řečtina-viz téma “řečtina” na stránce 899
- Turečtina-viz téma “Turečtina” na stránce 900
- Hebrejština-viz téma “Hebrejský” na stránce 901
- Farsi-viz téma “Perština” na stránce 902
- Urdu-viz téma “Urdština” na stránce 903
- Thajština-viz téma “Thajština” na stránce 903
- Laosky-viz téma “Laoština” na stránce 903
- Vietnamština-viz téma “Vietnamština” na stránce 903
- Japonština Latin SBCS-viz téma “Japonština Latin” na stránce 904
- Japonská Katakana SBCS-viz téma “Japonská Katakana SBCS” na stránce 905
- Japonština-Kanji/Latin Mixed-viz téma “Japonština-Kanji/Latin” na stránce 907
- Japonština-Kanji/Katakana Smíšený-viz téma “Smíšené červené, fialové a růžové květy” na stránce 908
- Korejština-viz téma “Korejština” na stránce 910
- Zjednodušená čínština-viz téma “Zjednodušená čínština” na stránce 910
- Tradiční čínština-viz téma “Tradiční čínština” na stránce 911

americká angličtina

Podrobnosti o CCSID a konverzi CCSID pro americkou angličtinu.

Následující tabulka ukazuje nativní CCSID pro americkou angličtinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	37, 924, 1140
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 1252, 5348, 858
HP Integrity NonStop Server, Solaris, Linux	819, 923
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.

Následující tabulka uvádí nativní CCSID pro němčinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	273, 924, 1141
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro dánštinu a norštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	277, 924, 1142
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 865, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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:

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

Kódová stránka:

819
Nepřevede na kódovou stránku 865

HP-UX

Kódová stránka:

1051
Nepřevede na kódovou stránku 865

Windows

Kódová stránka:

865
Nepřevádět na kódové stránky 1051, 1275

Finnish a

Podrobnosti o CCSID a konverzi CCSID pro finštinu a švédštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro finštinu a švédštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	278, 924, 1143
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 865, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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:

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

Kódová stránka:

819
Nepřevede na kódovou stránku 865

850
Nepřevede na kódovou stránku 865

HP-UX

Kódová stránka:

1051
Nepřevede na kódovou stránku 865

Windows

Kódová stránka:

865
Nepřevádět na kódové stránky 1051, 1275

italština

Podrobnosti o CCSID a konverzi CCSID pro italštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro italštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	280, 924, 1144
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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:

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.

Následující tabulka ukazuje nativní CCSID pro španělštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	284, 924, 1145
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923

Platforma	Nativní CCSID
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:

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.

Následující tabulka ukazuje nativní CCSID pro anglickou anglickou verzi/Gaelin na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	285, 924, 1146
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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:

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.

Následující tabulka ukazuje nativní identifikátory CCSID pro francouzštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	297, 924, 1147
AIX	819, 923, 5348

Platforma	Nativní CCSID
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, Solaris, 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.

Následující tabulka zobrazuje nativní CCSID pro vícejazyčný převod na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	500, 924, 1148
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	437, 850, 858, 1252, 5348
HP Integrity NonStop Server, SINIX, Solaris, Linux	819, 923
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.

Následující tabulka zobrazuje nativní CCSID pro portugalštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i	37, 500, 924, 1140
z/OS	500, 924, 1140
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 860, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923
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
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

HP-UX

Kódová stránka:

1051
Nepřevede na kódovou stránku 860

Windows

Kódová stránka:

860
Nepřevádět na kódové stránky 1051, 1275

Islandština

Podrobnosti o CCSID a převodu CCSID pro islandštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro islandštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	871, 924, 1149
AIX	819, 923, 5348
HP-UX	819, 923, 1051
Windows	850, 858, 861, 1252, 5348
HP Integrity NonStop Server, Solaris, Linux	819, 923

Platforma	Nativní CCSID
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

HP-UX

Kódová stránka:

1051
Nepřevede na kódovou stránku 861

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.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro východoevropské jazyky na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	870, 1153
Windows	852, 1250, 5346, 9044
AIX, HP-UX, HP Integrity NonStop Server, Solaris, 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

HP-UX, Solaris, Linux

Kódová stránka:

912
Nepřevádět na kódové stránky 1284, 1285

HP Integrity NonStop Server

Kódová stránka:

912
Nepřevede na kódové stránky 1153, 1284, 1285, 9044

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.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro cyrilici na podporovaných platformách:

Platforma	Nativní CCSID
z/OS	1025
IBM i	880, 1025
Windows	855, 866, 1131, 1251, 5347
Solaris	878, 915
AIX, HP-UX, Linux, HP Integrity NonStop Server	915
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.

Následující tabulka ukazuje nativní CCSID pro estonštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1122, 1157
Windows	902, 922, 1257, 5353, 9449
AIX, HP-UX, Solaris, Linux	902, 922
HP Integrity NonStop Server	922

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

HP-UX, Solaris, 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

HP Integrity NonStop Server

Kódová stránka:

922
Nepřevádí se na kódové stránky 902, 1157, 9449

Lotyšské a litevské

Podrobnosti o CCSID a konverze CCSID pro lotyštinu a litevštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro lotyšštinu a litevštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1112, 1156
Windows	901, 921, 1257, 5353, 9449
AIX, HP-UX, Solaris, Linux	901, 921
HP Integrity NonStop Server	921

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

HP-UX, Solaris, 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

HP Integrity NonStop Server

Kódová stránka:

921
Nepřevádí se na kódové stránky 901, 1156, 9449

Ukrajinština

Podrobnosti o CCSID a konverzi CCSID pro ukrajinštinu.

Následující tabulka ukazuje nativní CCSID pro Ukranian na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1123
Windows	1124, 1125, 1251, 5347
AIX, HP-UX, HP Integrity NonStop Server, Solaris, 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

HP-UX

Kódová stránka:

1124
Nepřevede na kódovou stránku 5347

Windows

Kódová stránka:

1125
Nepřevádí na kódovou stránku 1123

řečtina

Podrobnosti o CCSID a konverzi CCSID pro řečtinu.

Následující tabulka ukazuje nativní CCSID pro řečtinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	875
HP-UX	813 (viz poznámka)
Windows	869, 1253, 5349
AIX, NCR, HP Integrity NonStop Server, Solaris, Linux	813
klient Apple	1280
Klient systému DOS	737
Poznámka: V systému HP-UXje podporována pouze kódová sada ISO. Kódová sada greek8 proprietárního souboru HP-UX nemá žádné registrované CCSID a není podporována.	

Všechny jiné než klientské platformy podporují převod mezi svými nativními CCSID, nativní CCSID ostatních platforem s následujícími výjimkami.

IBM i

Kódová stránka:

875
Nepřevede na kódovou stránku 5349

Windows

Kódová stránka:

1253
Nepřevede na kódovou stránku 737

5349
Nepřevede na kódovou stránku 737

Turečtina

Podrobnosti o CCSID a konverzi CCSID pro turečtinu.

Následující tabulka zobrazuje nativní CCSID pro turečtinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1026
HP-UX	920 (viz poznámka)
Windows	857, 1254, 5350
AIX, HP Integrity NonStop Server, Solaris, Linux	920
klient Apple	1281
Poznámka: V systému HP-UXje podporována pouze kódová sada ISO. Kódová sada turkish8 vlastněný systémem HP-UX nemá žádné registrované CCSID a není podporována.	

Všechny platformy, které nejsou klienty, podporují převod mezi svými nativními CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

1026
Nepřevede na kódovou stránku 5350

Hebrejský

Podrobnosti o CCSID a převodu CCSID pro hebrejštinu.

Následující tabulka zobrazuje nativní CCSID pro hebrejštinu na podporovaných platformách:

Platforma	Nativní CCSID
z/OS	424, 803, 4899, 12712
IBM i	424
AIX	916, 9048
HP-UX	916 (viz poznámka)
Windows	1255, 5351
HP Integrity NonStop Server, Solaris, Linux	916
Poznámka: V systému HP-UXje podporována pouze kódová sada ISO. Kódová sada greek8 proprietárního souboru HP-UX nemá žádné registrované CCSID a není podporována.	

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

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

Následující tabulka obsahuje nativní identifikátory CCSID pro arabštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	420
AIX	1046, 1089
HP-UX	1089 (viz poznámka)
Windows	720, 864, 1256, 5352
HP Integrity NonStop Server, Solaris, Linux	1089
Poznámka: V systému HP-UXje podporována pouze kódová sada ISO. The HP-UX proprietary arabic8 codeset has no registered CCSID and is not supported.	

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

HP-UX, Solaris, Linux, HP Integrity NonStop Server, 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.

Následující tabulka ukazuje nativní CCSID pro Farsi na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1097
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux Windows	1098 (viz poznámka)

Platforma	Nativní CCSID
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.

Následující tabulka uvádí nativní identifikátory CCSID pro aplikaci Urdu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	918
Windows	868
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	1006

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

918
Nepřevede na kódovou stránku 1006

Thajština

Podrobnosti o CCSID a převodu CCSID pro thajštinu.

Následující tabulka obsahuje nativní identifikátory CCSID pro thajštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	838
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux 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.

Následující tabulka zobrazuje nativní CCSID pro Lao na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1132
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux Windows	1133

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem.

Vietnamština

Podrobnosti o CCSID a převodu CCSID pro vietnamštinu.

Následující tabulka zobrazuje nativní CCSID pro vietnamštinu na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	1130
Windows	1258, 5354
AIX, HP-UX, HP Integrity NonStop Server, Solaris, Linux	1129

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

IBM i

Kódová stránka:

1130
Nepřevádět na kódové stránky 1129, 5354

Japonština Latin

Podrobnosti o CCSID a konverzi CCSID pro japonštinu Latin SBCS.

Následující tabulka ukazuje 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 poznámky 2 a 3)
Linux, HP Integrity NonStop Server, Solaris	943, 5050
HP-UX	Není známo

Poznámka:

1. 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na systému AIX. CCSID hlášený operačním systémem je 33722.
2. Windows NT používá kódovou stránku 932, ale to je nejlépe reprezentované CCSID 943. Tento CCSID však nepodporuje všechny platformy WebSphere MQ .

   V systému WebSphere MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943.
3. Produkt WebSphere MQ nepodporuje kódové stránky založené na standardu JIS X 0213 (JIS2004).

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

1027
Nepřevádí se na kódové stránky 932, 942, 943, 954, 5050, 33722

IBM i

Kódová stránka:

1027
Nepřevede na kódovou stránku 932

AIX

Kódová stránka:

932
Nepřevede na kódovou stránku 1027

5050
Nepřevede na kódovou stránku 1027

33722
Nepřevede na kódovou stránku 1027

Linux

Kódová stránka:

943
Nepřevede na kódovou stránku 1027

5050
Nepřevede na kódovou stránku 1027

Solaris

Kódová stránka:

943
Nepřevede na kódovou stránku 1027

5050
Nepřevede na kódovou stránku 1027

HP Integrity NonStop Server

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.

Následující tabulka ukazuje nativní identifikátory CCSID pro japonské znaky Katakana SBCS na podporovaných platformách:

Platforma	Nativní CCSID
IBM i, z/OS	290
HP-UX	897
AIX	932, 5050, 33722 (viz poznámka 1)
Windows	932, 943 (viz poznámky 2 a 3)
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Platforma	Nativní CCSID
Poznámka: 1. 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na systému AIX. CCSID hlášený operačním systémem je 33722. 2. Windows NT používá kódovou stránku 932, ale to je nejlépe reprezentované CCSID 943. Tento CCSID však nepodporuje všechny platformy WebSphere MQ . V systému WebSphere MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943. 3. Produkt WebSphere MQ nepodporuje kódové stránky založené na standardu JIS X 0213 (JIS2004). 4. Kromě výše uvedených převodů podporují produkty WebSphere MQ v systému AIX, HP-UX, Solaris, Linux a Tru64 konverzi z CCSID 897 na CCSID 37, 273, 277, 278, 280, 284, 285, 290, 297, 437, 500, 819, 850, 1027 a 1252.	

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

290
Nepřevádí se na kódové stránky 932, 943, 954, 5050, 33722

IBM i

Kódová stránka:

290
Nepřevede na kódovou stránku 932

AIX

Kódová stránka:

932
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

33722
Nepřevádí na kódové stránky 290, 897

HP-UX

Kódová stránka:

897
Nepřevádí se na kódové stránky 932, 943, 954, 5050, 33722

Linux

Kódová stránka:

943
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

Solaris

Kódová stránka:

943
Nepřevádí na kódové stránky 290, 897

5050
Nepřevádí na kódové stránky 290, 897

HP Integrity NonStop Server

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.

Následující tabulka ukazuje 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)
HP-UX	932, 954, 5039 (viz poznámka 3)
Windows	932, 943 (viz poznámky 4 a 5)
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Poznámka:

1. 5035 je CCSID související s kódovou stránkou 939
2. 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na systému AIX. CCSID hlášený operačním systémem je 33722.
3. Kódové sady `japan15` a SJIS v systému HP-UX jsou reprezentovány CCSID 932. Tyto znaky mají několik znaků DBCS, které mají v systému SJIS odlišné znázornění, takže hodnota 932 může být nesprávně převedena, pokud konverze není provedena na systému HP-UX . WebSphere MQ for HP-UX podporuje 5039, správný CCSID pro HP SJIS. Změnu souboru `/var/mqm/conv/ccsid.tbl` lze provést, chcete-li změnit CCSID použité z 932 na 5039.
4. Windows NT používá kódovou stránku 932, ale to je nejlépe reprezentované CCSID 943. Tento CCSID však nepodporuje všechny platformy WebSphere MQ .

   V systému WebSphere MQ for Windows CCSID 932 se používá ke znázornění kódové stránky 932, ale změna souboru `../conv/table/ccsid.tbl` může být provedena, což změní CCSID použité na 943.
5. Produkt WebSphere MQ nepodporuje kódové stránky založené na standardu JIS X 0213 (JIS2004).

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

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

Kódová stránka:

1399
Nepřevede na kódovou stránku 5039

5035
Nepřevede na kódovou stránku 5039

HP-UX

Kódová stránka:

932
Nepřevádí na kódové stránky 942, 943, 1399

954
Nepřevádí na kódové stránky 942, 943, 1399

5039
Nepřevádí na kódové stránky 942, 943, 1399

HP Integrity NonStop Server

Kódová stránka:

943
Nepřevede na kódovou stránku 1399

5050
Nepřevede na kódovou stránku 1399

Smíšené červené, fialové a růžové květy

Podrobnosti o CCSID a konverzi CCSID pro japonskou směs Kanji/Katakana.

Tabulka 582. Nativní CCSID pro japonštinu Kanji/Katakana Smíšený na podporovaných platformách

Platforma	Nativní CCSID
z/OS	1390, 5026 (viz poznámka 1)
IBM i	5026 (viz poznámka 1)
AIX	932, 5050, 33722 (viz poznámka 2)
HP-UX	932, 954, 5039 (viz poznámka 3)
Windows	932, 943 (viz poznámky 4 a 5)
Linux, HP Integrity NonStop Server, Solaris	943, 5050

Tabulka 582. Nativní CCSID pro japonštinu Kanji/Katakana Smíšený na podporovaných platformách (pokračování)

Platforma	Nativní CCSID

Poznámka:

1. CCSID 1390 nepřijímá malá písmena. 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. 5050 a 33722 jsou CCSID související se základní kódovou stránkou 954 na systému AIX. CCSID ohlášený operačním systémem je 33722.
3. Kódové sady `japan15` a SJIS v systému HP-UX jsou reprezentovány CCSID 932. Ty mají několik znaků DBCS, které mají různé reprezentace v SJIS, takže 932 může být nesprávně převedeno, pokud se konverze neprovádí v systému HP-UX . WebSphere MQ pro HP-UX podporuje 5039, správný CCSID pro HP SJIS. Změna souboru `/var/mqm/conv/ccsid.tbl` může být provedena pro změnu CCSID použitého z 932 na 5039.
4. Windows NT používá kódovou stránku 932, ale toto je nejlépe reprezentováno CCSID 943. Avšak ne všechny platformy produktu WebSphere MQ podporují tento CCSID.

   V systému WebSphere MQ pro systém Windowsse k reprezentaci kódové stránky 932 používá CCSID 932, ale lze provést změnu souboru `../conv/table/ccsid.tbl` , která změní CCSID použitý na 943.
5. WebSphere MQ nepodporuje kódové stránky založené na standardu JIS X 0213 (JIS2004).

Všechny platformy podporují převod mezi jejich nativními identifikátory CCSID a nativními identifikátory CCSID jiných platforem, s následujícími výjimkami.

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

Kódová stránka:

5026
Nepřevádí na kódové stránky 1390, 5039

HP-UX

Kódová stránka:

932
Nepřevádí se na kódové stránky 942, 943, 1390

954
Nepřevádí se na kódové stránky 942, 943, 1390

5039
Nepřevádí se na kódové stránky 942, 943, 1390

HP Integrity NonStop Server

Kódová stránka:

943
Nepřevede na kódovou stránku 1390

5050
Nepřevede na kódovou stránku 1390

Korejština

Podrobnosti o CCSID a konverzi CCSID pro korejštinu.

Následující tabulka ukazuje nativní CCSID pro korejštinu na podporovaných platformách:

Platforma	Nativní CCSID
z/OS, IBM i	933, 1364
AIX, HP-UX, Linux, HP Integrity NonStop Server, Solaris	970
Windows	949, 1363

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

933
Nepřevede na kódovou stránku 970

1364
Nepřevede na kódovou stránku 970

HP-UX

Kódová stránka:

970
Nepřevádí na kódové stránky 949, 1363, 1364

Zjednodušená čínština

Podrobnosti o CCSID a konverzi CCSID pro zjednodušenou čínštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro zjednodušenou čínštinu na podporovaných platformách:

Platforma	Nativní CCSID
z/OS	935, 1388
IBM i	935, 1388
AIX	1383, 1386
HP-UX	1381 (viz poznámka 1)
Windows	1381, 1386 (viz poznámka 2)
Linux, HP Integrity NonStop Server, Solaris	1383

Platforma	Nativní CCSID
Poznámka: 1. Kódové sady `prc15` a `hp15CN` v systému HP-UX jsou reprezentovány CCSID 1381. 2. Systém Windows používá kódovou stránku 936, ale to je nejlépe reprezentováno hodnotou CCSID 1386. Tento CCSID však nepodporuje všechny platformy WebSphere MQ . V systému WebSphere MQ for Windows CCSID 1381 se používá ke znázornění kódové stránky 936, ale lze provést změnu do souboru `../conv/table/ccsid.tbl` , který změní identifikátor CCSID použitý na 1386. 3. Produkt WebSphere MQ podporuje fázi jedna z čínských standardů čínské GB18030 . V systému z/OS, Linux, Windowsa Solaris je podpora konverze poskytována mezi Unicode (UTF-8 a UCS-2) a s CCSID 1388 (EBCDIC s příponou GB18030), Unicode (UTF-8 a UCS-2) a CCSID 5488 (GB18030 jedna) a mezi CCSID 1388 a CCSID 5488. **Poznámka:** V systému IBM ije podpora poskytována operačním systémem pro převod mezi kódovou stránkou Unicode (UTF-8 a UCS-2) a kódovou sadou CCSID 1388 (EBCDIC s příponou GB18030). V systému HP-UX v současné době není v operačním systému HP11 pro operační systém GB18030k dispozici žádná podpora. V systému HP11iposkytuje oprava PHCO_26456 podporu převodu mezi GB18030 (CCSID 5488) a Unicode. Podpora není poskytována pro převod mezi GB18030 a 1388 (EBCDIC).	

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

935
Nepřevede na kódovou stránku 1383

1388
Nepřevede na kódovou stránku 1383

HP-UX

Kódová stránka:

1381
Nepřevádí se na kódové stránky 1383, 1386, 1388

Tradiční čínština

Podrobnosti o CCSID a konverzi CCSID pro tradiční čínštinu.

V následující tabulce jsou uvedeny nativní identifikátory CCSID pro tradiční čínštinu na podporovaných platformách:

Platforma	Nativní CCSID
z/OS, IBM i	937
HP-UX	938, 950, 964 (viz poznámka)
Windows	950
AIX, HP Integrity NonStop Server, Solaris, Linux	950, 964

Platforma	Nativní CCSID
Poznámka: Kódová sada `roc15` v systému HP-UX je reprezentována identifikátorem CCSID 938.	

Všechny platformy podporují převod mezi svými nativními CCSID a nativními CCSID jiných platforem, s následujícími výjimkami.

z/OS

Kódová stránka:

937
Nepřevede na kódovou stránku 964

1388
Nepřevede na kódovou stránku 1383

HP-UX

Kódová stránka:

938
Nepřevede na kódovou stránku 948

950
Nepřevede na kódovou stránku 948

964
Nepřevede na kódovou stránku 948

Linux, Solaris

Kódová stránka:

964
Nepřevede na kódovou stránku 938

Podpora konverze Unicode

Některé platformy podporují převod uživatelských dat na kódování Unicode nebo z kódování Unicode. Podporovány jsou dvě formy kódování Unicode: UCS-2 (CCSID 1200, 13488 a 17584) a UTF-8 (CCSID 1208).

Výraz *UCS-2* je často používán zaměnitelně, ale nesprávně s *UTF-16*. UCS-2 je kódování pevné šířky, kde každý znak zabírá 2 bajty. UTF-16 je kódování s proměnnou šířkou, které je nadřazenou sadou UCS-2. Kromě 2bajtových znaků UCS-2 obsahuje znaky UTF-16 znaky, známé jako náhradní páry, které mají délku 4 bajtů. Produkt WebSphere MQ nepodporuje náhradní páry. Podpora pro UTF-16 a UTF-8 v produktu WebSphere MQ je tedy omezena na znaky Unicode, které lze zakódovat v UCS-2.

Poznámka: Produkt WebSphere MQ nepodporuje identifikátory CCSID správce front UCS-2 , takže data záhlaví zprávy nemohou být zakódována v UCS-2.

Podpora produktu WebSphere MQ AIX pro kódování Unicode

V produktu WebSphere MQ for AIX je pro identifikátory CCSID v následující tabulce podporován převod na kódování CCSID Unicode a z něj.

037	273	278	280	284	285
297	423	437	500	813	819
850	852	856	857	858	860
861	865	867	869	875	878

880	901	902	912	915	916
920	923	924	932	933	935
937	938	939	942	943	948
949	950	954	964	970	1026
1046	1089	1129	1130	1131	1132
1133	1140	1141	1142	1143	1144
1145	1146	1147	1148	1149	1200
1153	1156	1157	1208	1250	1251
1253	1254	1258	1280	1281	1282
1283	1284	1285	1363	1364	1381
1383	1386	1388	4899	5026	5035
5050	5346	5347	5348	5349	5350
5351	5352	5353	5354	5488	9044
9048	9449	12712	13488	17584	33722

Podpora kódování Unicode produktu WebSphere MQ HP-UX

V systému WebSphere MQ for HP-UX je pro identifikátory CCSID uvedené v následující tabulce podporována konverze na a z CCSID Unicode.

437	737	813	819	850	852
855	857	861	864	865	866
869	874	912	915	916	920
932	938	950	954	964	970
1051	1089	1140	1141	1142	1143
1144	1145	1146	1147	1148	1149
1200	1208	1250	1251	1252	1253
1254	1255	1256	1257	1258	1381
5050	5488	13488	33722		

Podpora kódování Unicode v systému WebSphere MQ for Windows, Solaris a Linux

Na systémech WebSphere MQ for Windows, WebSphere MQ for Solaris a WebSphere MQ pro převod Linux a z CCSID Unicode jsou podporovány identifikátory CCSID v následující tabulce.

037	277	278	280	284	285
290	297	300	301	420	424
437	500	813	819	833	835
836	837	838	850	852	855
856	857	858	860	861	862
863	864	865	866	867	868
869	870	871	874	875	878
880	891	897	901	902	903

904	912	913 (5)	915	916	918
920	921	922	923	924	927
928	930	931 (1)	932 (2)	933	935
937	938 (3)	939	941	942	943
947	948	949	950	951	954 (4)
964	970	1006	1025	1026	1027
1040	1041	1042	1043	1046	1047
1051	1088	1089	1097	1098	1112
1114	1115	1122	1123	1124	1129
1130	1132	1133	1140	1141	1142
1143	1144	1145	1146	1147	1148
1149	1153	1156	1157	1200	1208
1250	1251	1252	1253	1254	1255
1256	1257	1258	1275	1280	1281
1282	1283	1363	1364	1380	1381
1383	1386	1388	4899	5050	5346
5347	5348	5349	5350	5351	5352
5353	5354	5488 (5)	9044	9048	9449
12712	13488	17584	33722 (4)		

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, Linuxa Solaris.

Podpora produktu IBM i pro kódování Unicode

Podrobné informace o podpoře UNICODE najdete v příslušné publikaci IBM i týkající se vašeho operačního systému.

Podpora kódování Unicode v produktu WebSphere MQ for z/OS

Převod na produkt WebSphere MQ for z/OS na základě CCSID Unicode a z něj je podporován pro následující identifikátory CCSID:

37	256	259	273	275	277
278	280	282	284	285	290
293	297	300	301	367	420
423	424	437	500	720	737
775	803	806	808	813	819
833	834	835	836	837	838

848	849	850	851	852	855
856	857	858	859	860	861
862	863	864	865	866	867
868	869	870	871	872	874
875	878	880	891	895	896
897	901	902	903	904	905
912	914	915	916	918	920
921	922	923	924	927	928
930	932	933	935	937	939
941	942	943	944	946	947
948	949	950	951	1004	1006
1008	1009	1010	1011	1012	1013
1014	1015	1016	1017	1018	1019
1025	1026	1027	1040	1041	1042
1043	1046	1047	1051	1088	1089
1097	1098	1112	1114	1115	1122
1123	1124	1125	1126	1129	1130
1131	1132	1133	1137	1140	1141
1142	1143	1144	1145	1146	1147
1148	1149	1153	1154	1155	1156
1157	1158	1159	1160	1161	1162
1164	1200	1208	1250	1251	1252
1253	1254	1255	1256	1257	1258
1275	1276	1277	1280	1281	1282
1283	1284	1285	1351	1362	1363
1364	1370	1371	1380	1381	1385
1386	1388	1390	1399	4899	4909
4930	4933	4948	4951	4952	4960
4971	5012	5039	5104	5123	5142
5210	5346	5347	5348	5349	5350
5351	5352	5353	5354	5488	8482
8612	9027	9030	9044	9048	9049
9056	9061	9066	9238	9449	12712
13121	13218	13488	16684	16804	17248
17584	21427	28709			

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 32bitovém i 64bitovém systému WebSphere MQ :

Název	Délka
MQLONG	4 bajty
MQULONG	4 bajty
MQINT32	4 bajty
MQUINT32	4 bajty
MQINT64	8 bajtů
MQUINT64	8 bajtů

Standardní datové typy

Informace o standardních datových typech v 32bitových aplikacích UNIX, 64bitových systémech UNIXa 64bitových systémech Windows .

32bitové aplikace UNIX

Tato sekce je zahrnuta pro porovnání a je založena na systému Solaris. Všechny rozdíly s jinými platformami UNIX jsou zaznamenány:

Název	Délka
ZNAK	1 bajt
short	2 bajty
celé číslo	4 bajty
long	4 bajty
float	4 bajty
dvojitý	8 bajtů
long double	16 bajtů Všimněte si, že na systému AIX a Linux PPC je dlouhé dvojité délky 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 Všimněte si, že v systému AIX je wchar_t 2 bajty.

64bitové aplikace systému UNIX

Tato sekce je založena na systému Solaris. Všechny rozdíly s jinými platformami UNIX jsou zaznamenány:

Název	Délka
ZNAK	1 bajt
short	2 bajty

Název	Délka
celé číslo	4 bajty
long	8 bajtů
float	4 bajty
dvojitý	8 bajtů
long double	16 bajtů
	Všimněte si, že na systému AIX a Linux PPC je dlouhé dvojité délky 8 bajtů.
Ukazatel	8 bajtů
ptrdiff_t	8 bajtů
velikost_t	8 bajtů
time_t	8 bajtů
hodin_hodin	8 bajtů
	Všimněte si, že na jiných platformách UNIX je clock_t 4 bajty.
wchar_t	4 bajty
	Všimněte si, že v systému AIX je wchar_t 2 bajty.

64bitové aplikace Windows

Název	Délka
ZNAK	1 bajt
short	2 bajty
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

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 WebSphere MQ :

Odkaz SOAP

Přenos produktu WebSphere MQ pro referenční informace SOAP uspořádané abecedně.

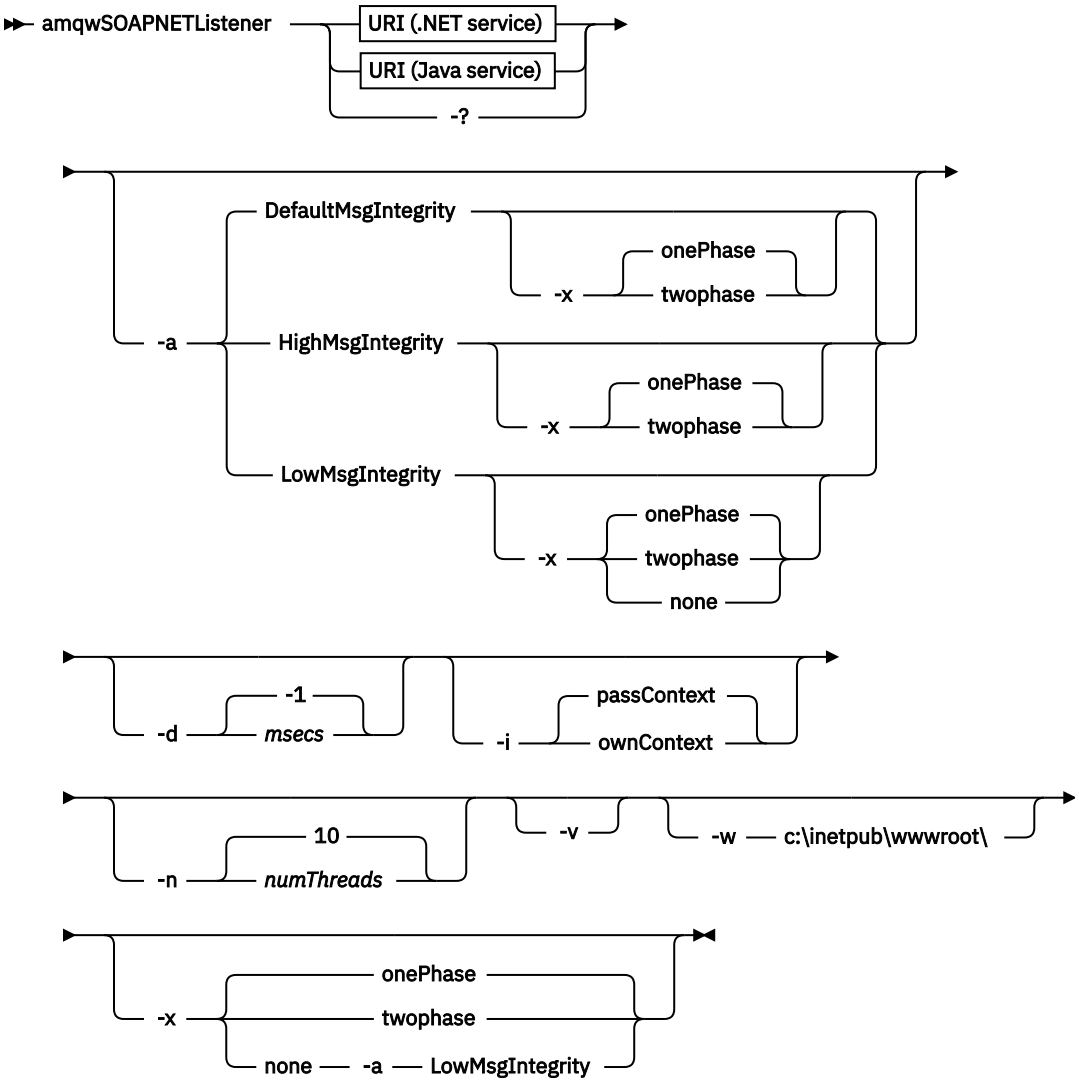
amqwSOAPNETListener: IBM WebSphere MQ modul listener SOAP pro prostředí .NET Framework 1 nebo 2

Syntaxe a parametry pro modul listener protokolu SOAP produktu WebSphere MQ pro prostředí .NET Framework 1 nebo 2.

Účel

Spustí modul listener protokolu SOAP produktu IBM WebSphere MQ pro prostředí .NET Framework 1 nebo 2.

.NET



Povinné parametry

URI ***platforma***

Viz “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

-?

Vytisknout text nápovědy popisující způsob použití příkazu.

Nepovinné parametry

-a ***integrityOption***

Volba *integrityOption* určuje chování modulů listener protokolu SOAP produktu WebSphere MQ , pokud nelze odeslat zprávu s nezdarem do fronty nedoručených zpráv. *integrityOption* může mít jednu z následujících hodnot:

DefaultMsgIntegrity

Pro netrvalé zprávy modul listener zobrazí varovnou zprávu a pokračuje ve zpracování s vyřazenou původní zprávou. V případě trvalých zpráv zobrazí chybovou zprávu a zazálohuje zprávu

s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena. DefaultMsgIntegrita se použije, pokud je vynechána volba -a , nebo není-li zadána volba *integrityOption* .

LowMsgIntegrity
U trvalých i dočasných zpráv modul listener zobrazí varování a pokračuje v provádění a zahození zprávy.

HighMsgIntegrity
V případě trvalých i dočasných zpráv modul listener zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Je-li zadána hodnota -x none , musí být zadána hodnota -a LowMsgIntegrity . Pokud jsou příznaky nekompatibilní, obslužný program implementace se ukončí s chybovou zprávou a bez kroků implementace, které byly provedeny.

-d ***ms***
Hodnota *ms* určuje počet milisekund pro modul listener protokolu SOAP produktu WebSphere MQ , který má být udržen v případě, že byly zprávy požadavků přijaty v libovolném podprocesu. Je-li parametr *ms* nastaven na hodnotu -1, modul listener zůstane naživu neomezeně dlouho.

-i ***Kontext***
Kontext určuje, zda listenery předávají kontext identity. *Kontext* má následující hodnoty:

passContext
Nastavte kontext identity původní zprávy požadavku do zprávy odpovědi. Modul listener SOAP kontroluje, zda má oprávnění uložit kontext z fronty požadavků a předat jej do fronty odpovědí. Kontextové kontroly provádějí při otevírání fronty požadavků do kontextu ukládání kontextu a ve frontě odpovědí pro předávání kontextu. Pokud nemá požadované oprávnění, nebo se volání MQOPEN nezdaří a zpráva odpovědi se nezpracuje. Zpráva odpovědi je vložena do fronty nedoručených zpráv s hlavičkou nedoručených zpráv, která obsahuje návratový kód z nezdaru MQOPEN. Modul listener poté pokračuje ve zpracování následných příchozích zpráv jako obvykle.

ownContext
Modul listener SOAP nepředává kontext. Vrácený kontext odráží ID uživatele, pod kterým je modul listener spuštěný, spíše než ID uživatele, který vytvořil původní zprávu požadavku.

Pole v kontextu původu jsou nastavena prostřednictvím správce front a nikoli modulem listener protokolu SOAP.

-n ***numThreads***
Volba *numThreads* určuje počet podprocesů v generovaných spouštěcích skriptech pro modul listener protokolu SOAP produktu WebSphere MQ . Výchozí hodnota je 10. Zvažte zvýšení tohoto čísla, pokud máte vysokou propustnost zpráv.

-v
-v nastaví podrobný výstup z externích příkazů. Chybové zprávy jsou vždy zobrazeny. Pomocí volby -v můžete vytvářet výstupní příkazy, které lze upravit, a vytvořit tak přizpůsobené skripty implementace.

-w ***serviceDirectory***
serviceDirectory je adresář obsahující webovou službu.

-x ***transakcionalita***
transactnost určuje typ transakčního řízení pro modul listener. *transakcionalitu* lze nastavit na jednu z následujících hodnot:

onePhase
IBM WebSphere MQ je použita jednofázová podpora. Pokud systém selže během zpracování, zpráva požadavku se znovu doručí do aplikace. Transakce WebSphere MQ zajistí, aby zprávy odezvy byly napsány přesně jednou.

twoPhase
Je použita dvoufázová podpora. Je-li služba zapsána správně, zpráva se doručí přesně jednou, koordinovanou s jinými prostředky, v rámci jediného potvrzeného provedení služby. Tato volba se vztahuje pouze na připojení vazeb serveru.

none
Žádná transakční podpora. Pokud dojde k selhání systému během zpracování, může dojít ke ztrátě zprávy požadavku i v případě, že je trvalá. Je možné, že služba byla nebo nemusela být provedena a že zprávy odezvy, sestavy nebo zprávy s deadutem mohou nebo nemusí být zapsány.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Podrobnosti najdete v popisu příznaku -a .

Příklad .NET

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

amqswsdl: generování služby WSDL pro službu .NET Framework 1 nebo 2

amqswsdl vezme webovou službu napsanou pro prostředí .NET Framework 1 nebo 2 a vygeneruje kód WSDL pro třídu, vloží identifikátor URI, který poskytnete pro přenos WebSphere MQ pro protokol SOAP do generovaného WSDL.

Účel

Pomocí produktu **amqswsdl** vygenerujte WSDL obsahující identifikátor URI služby implementované v produktu WebSphere MQ. Použijte WSDL ke generování proxy klienta.

Parametry

***escapedUri* (Input)**
Identifikátor URI služby, se všemi "&" unikl do "&amp.". Příklad:

```
"jms:/queue?destination=REQUESTDOTNET
&amp.initialContextFactory=com.ibm.mq.jms.Nojndi
&amp.connectionFactory=(connectQueueManager(QM1)binding(server))
&amp.targetService=Quote.asmx"
```

***className*.asmx (Input)**
Třída služeb.

***className*.wsdl (Výstup)**
WSDL služby.

Popis

Je-li třída implementována pomocí programovacího modelu kódu, musíte sestavit *className*.dll a uložit jej do ./bin.

amqwclientconfig: vytvořte deskriptor implementace klienta webových služeb 1.4 pro protokol WebSphere MQ pro protokol SOAP

Produkt **amqwclientconfig** vytvoří soubor deskriptoru implementace klienta client-config.wsdd Axis 1.4 .

Účel

Přidává transport jms:/ do deskriptoru a registruje java:com.ibm.mq.soap.transport.jms.WMQSender jako třídu pro obsluhu požadavků SOAP pro přenos jms: .

Syntaxe

Popis

amqwclientconfig zavolá příkaz **amqwsetcp** k nastavení proměnné CLASSPATH a spustí tento příkaz:

```
java org.apache.axis.utils.Admin client "%WMQSOAP_HOME%\bin\amqwclientTransport.wsdd"
```

amqwdeployWMQService: implementace obslužného programu webové služby

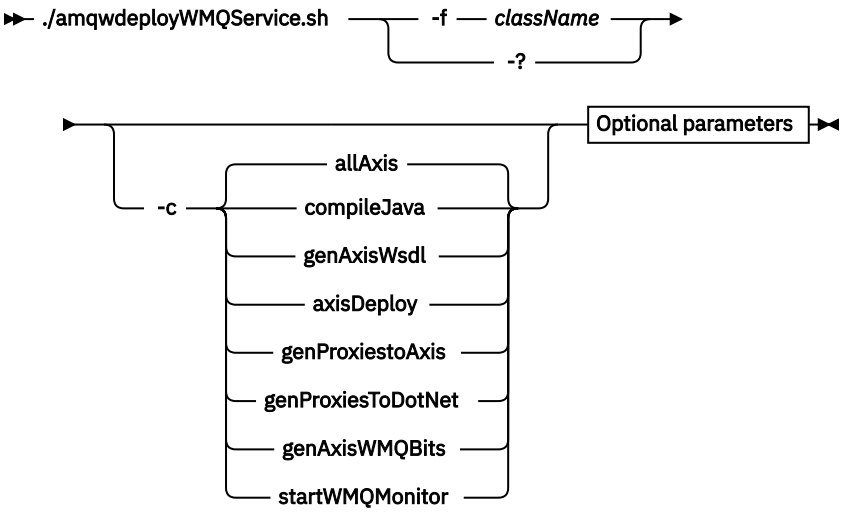
Obslužný program implementace připraví třídu služeb pro použití jako webovou službu pomocí produktu WebSphere MQ jako přenosu.

Účel

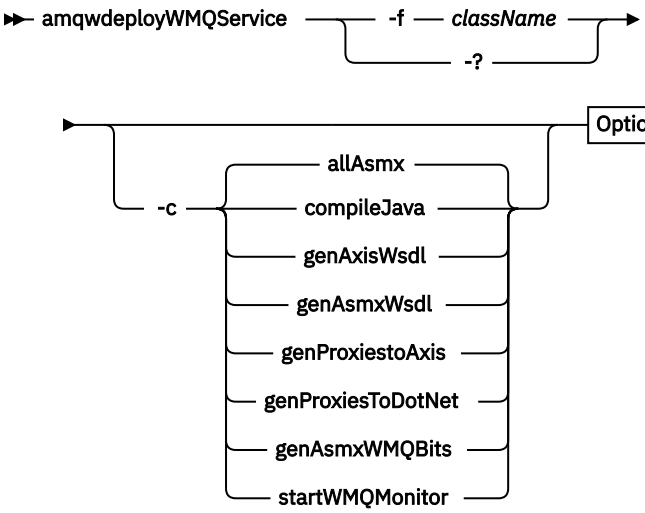
Pomocí obslužného programu implementace vygenerujte soubory, které jsou potřebné k implementaci služby Axis 1.4, .NET Framework 1 nebo .NET Framework 2. Použijte tyto soubory k implementaci služby vyvolané produktem IBM WebSphere MQ. Soubory vygenerované produktem **amqwdeployWMQService** jsou zobrazeny v “Výstupní soubory z amqwdeployWMQService” na stránce 928.

Syntax diagram

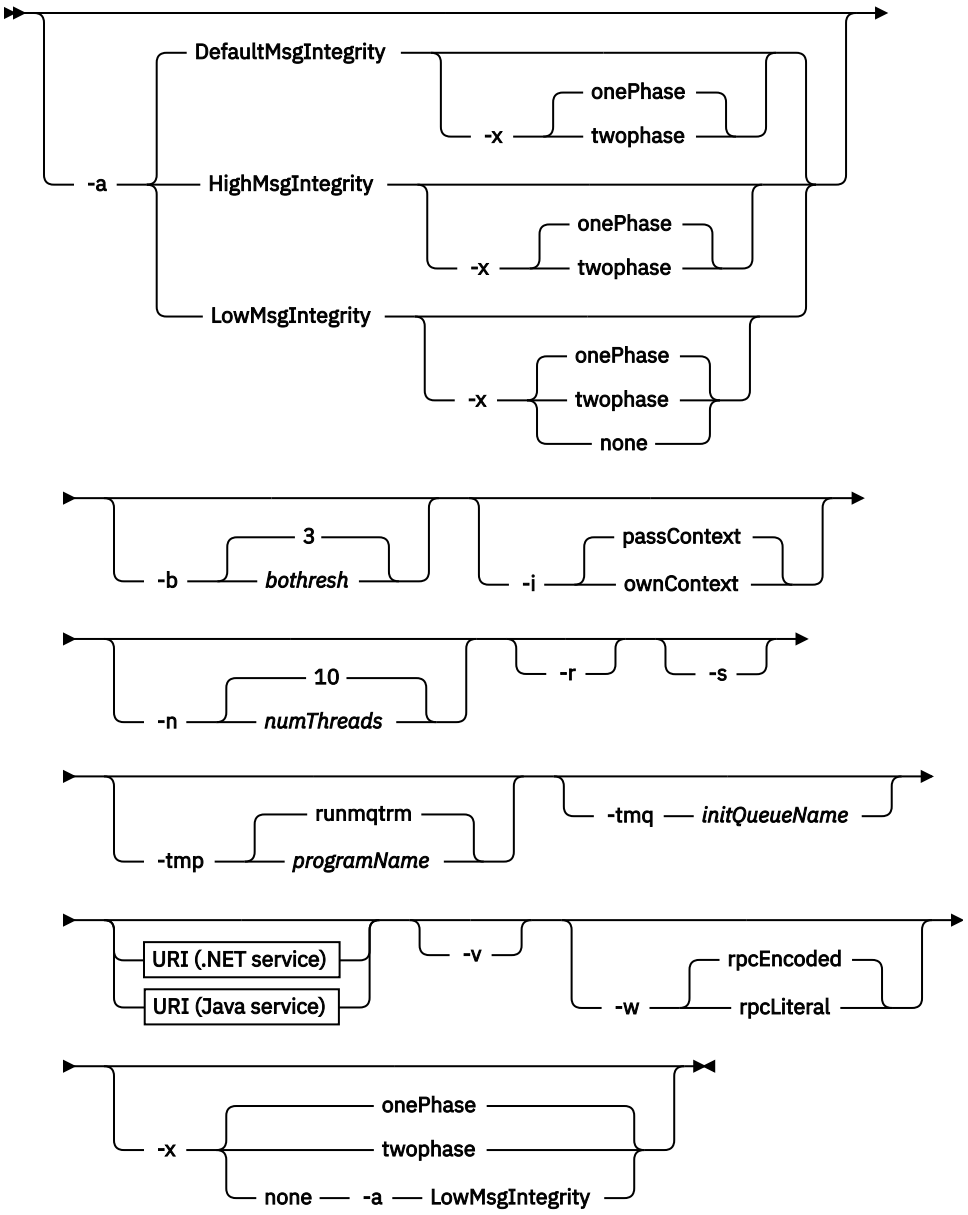
UNIX and Linux systems



Windows



Optional parameters



Povinné parametry

-f ***className***
className je název třídy, která se má implementovat. Pro služby Axis *className* je zdrojový soubor Java a pro služby .NET, soubor `.asmx` . Obrázek 11 na stránce 925 ilustruje implementaci služby Axis a Obrázek 12 na stránce 925 služby .NET.

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
```

Obrázek 11. Příklad implementace služby Axis

```
amqwdeployWMQService -f StockQuoteDotNet.asmx
```

Obrázek 12. Příklad implementace služby .NET

Pro prostředí Java musí být název balíku *className* plně kvalifikovaný. Může být uvedeno jako název cesty s oddělovači adresáře nebo jako název třídy s oddělovači období. Vygenerovaná třída se nachází v `./generated/client/remote/`*`path name`*. Pro službu .NET, ačkoli lze tento adresář zadat, jsou generované proxy Java vždy umístěny v `./generated/client/remote/dotNetService`.

Zadáte-li identifikátor URI s volbou -u a v identifikátoru URI uvedete *targetService*, obslužný program implementace zkontroluje *className*. *className* musí odpovídat *targetService*. Pokud se třída a služba neshodují, obslužný program implementace zobrazí chybovou zprávu a ukončí se.

-?
Vytisknout text nápovědy popisující způsob použití příkazu.

Nepovinné parametry

-a ***integrityOption***
Volba *integrityOption* určuje chování modulů listener protokolu SOAP produktu WebSphere MQ , pokud nelze odeslat zprávu s nezdarem do fronty nedoručených zpráv. *integrityOption* může mít jednu z následujících hodnot:

DefaultMsgIntegrity
Pro netrvalé zprávy modul listener zobrazí varovnou zprávu a pokračuje ve zpracování s vyřazenou původní zprávou. V případě trvalých zpráv zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena. DefaultMsgIntegrita se použije, pokud je vynechána volba -a , nebo není-li zadána volba *integrityOption* .

LowMsgIntegrity
U trvalých i dočasných zpráv modul listener zobrazí varování a pokračuje v provádění a zahození zprávy.

HighMsgIntegrity
V případě trvalých i dočasných zpráv modul listener zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Je-li zadána hodnota -x none , musí být zadána hodnota -a LowMsgIntegrity . Pokud jsou příznaky nekompatibilní, obslužný program implementace se ukončí s chybovou zprávou a bez kroků implementace, které byly provedeny.

-b ***threshosh***
Parametr *bothresh* určuje nastavení prahové hodnoty pro vrácení pro frontu požadavků. Výchozí hodnota je 3.

-c ***operace***
Parametr *operation* určuje, která část procesu implementace se má provést. *operace* je jedna z následujících možností:

allAxis
Provést všechny kroky kompilace a nastavení pro službu Axis nebo Java[4].

compileJava
Zkompilujte službu Java: `.java` na `.class`.

genAxisWsdl
Generovat WSDL: `.class` do `.wsdl`.

axisDeploy
Nasaďte soubor třídy: `.wsdl` na `.wsdd`, použijte `.wsdd`.

genProxiestoAxis
Generujte servery proxy: `.wsdl` pro `.java` a `.class`.

genAxisWMQBits
Nastavte IBM WebSphere MQ front, IBM WebSphere MQ modulů listener SOAP a spouštěče pro službu Axis.

allAsmx
Provést všechny kroky nastavení pro službu .NET[5].

genAsmxWsdl
Generovat WSDL: `.asmx` do `.wsdl`.

genProxiesToDotNet
Generovat servery proxy: `.wsdl` do `.java`, `.class`, `.cs` a `.vb`.

genAsmxWMQBits
Nastavení front IBM WebSphere MQ , IBM WebSphere MQ modulů listener a spouštěčů SOAP

startWMQMonitor
Spusťte monitor spouštěčů pro služby SOAP produktu WebSphere MQ .

Poznámka: runmqtrm běží pod ID uživatele `mqm` . Je-li zabezpečení problémem, musíte se ujistit, že jsou listenery spuštěny pod odpovídajícími ID uživatele.

-i ***Kontext***
Kontext určuje, zda listenery předávají kontext identity. *Kontext* má následující hodnoty:

passContext
Nastavte kontext identity původní zprávy požadavku do zprávy odpovědi. Modul listener SOAP kontroluje, zda má oprávnění uložit kontext z fronty požadavků a předat jej do fronty odpovědí. Kontextové kontroly provádějí při otevírání fronty požadavků do kontextu ukládání kontextu a ve frontě odpovědí pro předávání kontextu. Pokud nemá požadované oprávnění, nebo se volání `MQOPEN` nezdaří a zpráva odpovědi se nezpracuje. Zpráva odpovědi je vložena do fronty nedoručených zpráv s hlavičkou nedoručených zpráv, která obsahuje návratový kód z nezdaru `MQOPEN`. Modul listener poté pokračuje ve zpracování následných příchozích zpráv jako obvykle.

ownContext
Modul listener SOAP nepředává kontext. Vrácený kontext odráží ID uživatele, pod kterým je modul listener spuštěný, spíše než ID uživatele, který vytvořil původní zprávu požadavku.

Pole v kontextu původu jsou nastavena prostřednictvím správce front a nikoli modulem listener protokolu SOAP.

-n ***numThreads***
Volba *numThreads* určuje počet podprocesů v generovaných spouštěcích skriptech pro modul listener protokolu SOAP produktu WebSphere MQ . Výchozí hodnota je 10. Zvažte zvýšení tohoto čísla, pokud máte vysokou propustnost zpráv.

-r
Volba *-r* určuje, že budou nahrazeny všechny existující požadavky nebo definice fronty monitoru spouštěčů. Fronty monitoru spouštěčů jsou nahrazeny pouze tehdy, je-li zadán také parametr -tmq . Fronty jsou znovu vytvořeny se standardními výchozími atributy a stávající zprávy ve frontách se vymažou. Není-li použita volba -r , nebudou existující definice front změněny a existující zprávy nebudou odstraněny. Neuvedete-li -r, ujistěte se, že jsou zachovány všechny upravené atributy fronty.

---

[4] Výchozí, pokud je *className* má příponu .java
[5] Předvolba, pokud má *className* příponu .asmx.

-s
Nakonfigurujte modul listener, aby se spouštěl jako služba produktu WebSphere MQ . Pokud jsou zadány oba parametry -s a -tmq , obslužný program implementace zobrazí chybovou zprávu a ukončí se.

-tmp ***programName***
programName uvádí název programu monitoru spouštěčů. Použijte `-tmp` *programName* v prostředí UNIX nebo Linux jako alternativu k použití produktu **runmqtrm**. Programy, které iniciuje spuštění, běží pod oprávněním `mqm` .

Příklad:

```
amqwdeployWMQService -f javaDemos/service/StockQuoteAxis.java
-tmq trigger.monitor.queue -tmp trigmon
```

-tmq ***queueName***
Položka *queueName* určuje název fronty monitoru spouštěčů. Jsou vytvořeny definice procesů produktu IBM WebSphere MQ pro konfiguraci automatického spouštění modulů listener protokolu SOAP produktu WebSphere MQ s přidruženým názvem fronty monitoru spouštěčů. Není-li tato volba zadána, obslužný program implementace nedefinuje žádnou spouštěcí konfiguraci. Pokud jsou zadány oba parametry -s a -tmq , obslužný program implementace zobrazí chybovou zprávu a ukončí se.

URI ***platforma***
Viz “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

-v
-v nastaví podrobný výstup z externích příkazů. Chybové zprávy jsou vždy zobrazeny. Pomocí volby -v můžete vytvářet výstupní příkazy, které lze upravit, a vytvořit tak přizpůsobené skripty implementace.

-w
Volba -w určuje styl kódu WSDL, který má být vygenerován. Výchozí hodnota je rpcEnclodedpro kompatibilitu s předchozími vydáními přenosu produktu WebSphere MQ pro protokol SOAP. Chcete-li vytvořit kód WSDL kompatibilní s generováním proxy klienta Axis2 , použijte příkaz rpcLiteral . rpcEncoded není kompatibilní s doporučeními WS-I.

-x ***transakcionalita***
transactnost určuje typ transakčního řízení pro modul listener. *transakcionalitu* lze nastavit na jednu z následujících hodnot:

onePhase
IBM WebSphere MQ je použita jednofázová podpora. Pokud systém selže během zpracování, zpráva požadavku se znovu doručí do aplikace. Transakce WebSphere MQ zajistí, aby zprávy odezvy byly napsány přesně jednou.

twoPhase
Je použita dvoufázová podpora. Je-li služba zapsána správně, zpráva se doručí přesně jednou, koordinovanou s jinými prostředky, v rámci jediného potvrzeného provedení služby. Tato volba se vztahuje pouze na připojení vazeb serveru.

none
Žádná transakční podpora. Pokud dojde k selhání systému během zpracování, může dojít ke ztrátě zprávy požadavku i v případě, že je trvalá. Je možné, že služba byla nebo nemusela být provedena a že zprávy odezvy, sestavy nebo zprávy s deadutem mohou nebo nemusí být zapsány.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Podrobnosti najdete v popisu příznaku -a .

Chyby

Pokud jsou v systému Windowshlášeny chyby z **amqswsdl**, zkuste zaregistrovat soubory `.asmx` jako služby zadáním následujícího příkazu.

```
%windir%/Microsoft.NET/Framework/version number/aspnet_regiis.exe -ir
```

Problém se obvykle vyskytuje na systémech, kde IIS není instalován, nebo IIS byla nainstalována po NET. Problém je zjištěn, když **amqswsdl** generuje soubory `.wsdl` .

Poznámka: Klíče registru jsou také nezbytné k povolení modulu listener k vyvolání služeb. Používáte-li vlastní upravené procedury implementace, nemusíte se s problémem setkat, dokud nebude spuštěna běhová komponenta.

Výstupní soubory z amqwdeployWMQService

Seznam adresářů a výstupních souborů z **amqwdeployWMQService**

*Tabulka 583. Výstupní soubory z **amqwdeployWMQService***

Výstupy	Popis	Výstupní adresář	Název souboru
`.class`	Kompilovaný zdrojový soubor Java	`./generated/server/`*`server package`*	*`classname`*`.class`
`.wsdl`	popis služby	`./generated`	*`className`*`Axis_Wmq.wsdl` *`className`*`DotNet_Wmq.wsdl`
`.wsdd`	Klient Axis a soubory implementace služby	`./`	`client-config.wsdd` `server-config.wsdd`
		`./generated/server/`*`server package`*	*`className`*`_deploy.wsdd` *`className`*`_undeploy.wsdd`
Zdroj klienta (`.vb`, `.cs`, `.java`)	stuby klienta .Net do služby Axis	`./generated/client`	*`classname`*`AxisService.cs` *`classname`*`AxisService.vb`
	síťové stuby .Net do služby. Net	`./generated/client`	*`classname`*`DotNet.cs` *`classname`*`DotNet.vb`

*Tabulka 583. Výstupní soubory z **amqwdeployWMQService** (pokračování)*			
Výstupy	**Popis**	**Výstupní adresář**	**Název souboru**
Pomocník klienta (.java a .class)	proxy klienta Java pro službu. Net	./generated/server/soap/client/remote/dotnetService	*className*DotNet.class *className*DotNet.java *className*DotNetLocator.class *className*DotNetLocator.java *className*DotNetSoap12Stub.class *className*DotNetSoap12Stub.java *className*DotNetSoap_BindingStub.class *className*DotNetSoap_BindingStub.java *className*DotNetSoap_PortType.class *className*DotNetSoap_PortType.java
	proxy klienta Java na službu Axis	./generated/server/soap/client/remote/*client package*	SoapServer*className*AxisBindingSoapStub.class SoapServer*className*AxisBindingSoapStub.java *className*Axis.class *className*Axis.java *className*AxisService.class *className*AxisService.java *className*AxisServiceLocator.class *className*AxisServiceLocator.java
Skripty (.cmd a .sh)	Skripty modulu listener	/generated/server	startWMQJListener.cmd startWMQJListener.sh startWMQNListener.cmd endWMQJListener.cmd endWMQJListener.sh endWMQNListener.cmd

Poznámky k použití pro amqwdeployWMQService

Popisuje úlohy prováděné produktem **amqwdeployWMQService**.

Obslužný program implementace provádí následující akce.

1. Kontroluje cesty k následujícím souborům:
   - axis.jar.
   - *WMQSOAP_HOME*/java/lib/com.ibm.mq.soap.jar.
   - V systému Windows: csc.exe
2. V systému Windowspoužívá produkt %SystemRoot%\Microsoft.NET\Framework\v1.1.432 nebo, je-li kompilátor C# nainstalován, cestu k produktu csc.exe jako cestu k prostředí .NET Framework.

   Poznámka: Máte-li nainstalovaný produkt Microsoft Visual Studio 2008 (verze 9), portál wsdl.exe se nenachází v cestě k produktu csc.exe. Musíte přidat cestu k rozhraní .NET framework do proměnné cesty; například:

   ```
   Set Path=C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727;%Path%
   ```
3. Vytvoří adresář ./generated a požadované podadresáře, pokud neexistují.
4. Pro služby Java kompiluje zdroj do souboru *className*.class.

5. Generuje WSDL.
6. Pro služby Java vytvoří soubory deskriptoru implementace *className*_deploy.wsdd a *className*_undeploy.wsdd
7. Pro služby Java vytvoří nebo aktualizuje soubor deskriptoru implementace Axis, server-config.wsdd.
8. Generuje proxy klienta pro Java, C# a Visual Basic z WSDL.

   Poznámka: V systému Windowsobslužný program implementace generuje proxy pro Visual Basic a C# bez ohledu na jazyk, v němž je služba zapsána. Soubor WSDL a servery proxy vygenerované z něj zahrnují příslušný identifikátor URI pro volání služby:

   a.
   ```
   jms:/queue?destination=SOAPN.demos@WMQSOAP.DEMO.QM
   &connectionFactory=(connectQueueManager(WMQSOAP.DEMO.QM))
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &targetService=StockQuoteDotNet.asmx
   &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
   ```

   Obrázek 13. Příklad identifikátoru URI ve vygenerovaném klientu .NET k volání služby .NET

   b.
   ```
   jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM
   &connectionFactory=(connectQueueManager(WMQSOAP.DEMO.QM))
   &initialContextFactory=com.ibm.mq.jms.Nojndi
   &targetService=soap.server.StockQuoteAxis.java
   &replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
   ```

   Obrázek 14. Příklad identifikátoru URI v generovaném klientu .NET k volání služby Axis 1
9. Kompiluje proxy Java.
10. Vytvoří frontu WebSphere MQ , *requestQueue* pro zadržení požadavků na službu. Výchozí název fronty je ve tvaru SOAPJ.*directory*, nebo můžete zadat *requestQueue* ve volbě URI -u .
11. Vytvoří příkaz a skriptové soubory shellu ke spuštění modulů listener protokolu SOAP produktu WebSphere MQ , které zpracovávají frontu požadavků.
12. Pokud byla použita volba -tmq , obslužný program implementace vytvoří definice produktu WebSphere MQ pro automatické spuštění procesů modulu listener protokolu SOAP produktu WebSphere MQ .
    - Obslužný program implementace používá ke spuštění modulu listener atribut APPLICID příkazu **runmqsc** DEFINE PROCESS , který obsahuje příkaz ke spuštění modulu listener. Příkaz má název adresáře implementace, který je součástí tohoto adresáře. Pole APPLICID má maximální délku 256, což omezuje maximální délku adresáře implementace. Limit adresáře pro služby Java je následující:
      – Systémy UNIX and Linux : 218
      – Systém Windows: 197 mínus délka názvu fronty požadavků.

      Pro služby .NET je limit adresáře následující:
      – Windows: 209 minus délka názvu služby, minus rozšíření .asmx .
    - Obslužný program implementace kontroluje, zda je překročena mezní hodnota pro APPLICID . Je-li limit překročen, obslužný program se nepokusí definovat spouštěcí proces. Zobrazí chybovou zprávu a proces implementace selže bez provedení jakýchkoli kroků implementace.

    Následující příklady zobrazují příkazy konfigurace a spuštění vygenerované obslužným programem implementace ke spuštění modulu listener protokolu SOAP produktu WebSphere MQ .

    ```
    DEFINE PROCESS(requestQueue) APPLICID(applicIDStr) REPLACE
    ALTER  QLOCAL (requestQueue) TRIGTYPE(FIRST) TRIGGER
           PROCESS(requestQueue) INITQ(initQueueName) TRIGMPRI(0)
    ```

    Obrázek 15. Konfigurační příkazy produktu WebSphere MQ pro spuštění modulu listener SOAP.

```
applicIDStr = start "Java WMQSoapListener -requestQueue"
                    /min .\generated\server\startWMQJListener.cmd;
```

Obrázek 16. Spouštění modulu listener SOAP Axis v systému Windows

```
applicIDStr = start "WMQAsmxListener -className\
                    /min .\generated\server\startWMQNListener.cmd;
```

Obrázek 17. Spuštění modulu listener SOAP SOAP na systému Windows

```
applicIDStr = xterm -iconic -T \"Java WMQSoapListener_requestQueue\"
                   -e ./generated/server/startWMQJListener.sh & #
```

Obrázek 18. Spouštění modulu listener SOAP Axis na systémech UNIX and Linux

amqwRegisterdotNet: register IBM WebSphere MQ transport for SOAP to .NET

Registrujte přenos IBM WebSphere MQ pro protokol SOAP do globální mezipaměti sestavení v prostředí .NET.

Účel

Produkt **amqwRegisterdotNet** zaregistruje odesilatele WebSphere MQ SOAP, modul listener SOAP a procesor WSDL s prostředím .NET Framework 1 nebo 2.

Syntaxe

Popis

amqwRegisterdotNet se spouští automaticky během instalace. Není-li prostředí .NET Framework, které používáte, nainstalováno před přenosem WebSphere MQ pro SOAP, není třeba jej znovu spustit. Můžete ho spustit tolikrát, kolikrát chcete. Použijte jej k opětovné registraci přenosu WebSphere MQ pro protokol SOAP s různými verzemi produktu .NET Framework.

Poznámka: Na serveru Windows 2003 Server musíte také spustit obslužný program **aspnet_regiis** , a to i v případě, že neimplementujete na Internet Information Server (IIS). Umístění obslužného programu `aspnet_regiis.exe` se může lišit podle různých verzí produktu Microsoft .NET Framework, ale je obvykle umístěn v: `%SystemRoot%/Microsoft.NET/Framework/version number/aspnet_regiis`. Je-li nainstalováno více verzí, použijte produkt **aspnet_regiis** pro verzi .NET Framework, kterou používáte.

Licence na software Apache

Apache License Version 2.0, January 2004 http://www.apache.org/licenses/

http://www.apache.org/licenses/

Licence Apache
Verze 2.0, leden 2004
http://www.apache.org/licenses/

USTANOVENÍ A PODMÍNKY PRO POUŽITÍ, REPRODUKCI A DISTRIBUCI

1. Definice.

'Licencí' se rozumí podmínky pro použití, rozmnožování,
a distribuce, jak je definováno oddíly 1 až 9 tohoto dokumentu.

"Poskytovatelem licence" se rozumí vlastník nebo subjekt pověřený autorským právem.
vlastníkem autorských práv, který uděluje licenci.

'Právnická osoba' se rozumí unie jednajícího subjektu a všech
jiné subjekty, které kontrolují, jsou kontrolovány nebo jsou pod společným
ovládání s tímto subjektem. Pro účely této definice se:
"ovládací prvek" znamená (i) sílu, přímé nebo nepřímé, aby došlo k
směr nebo správa takového subjektu, ať již podle smlouvy nebo
jinak, nebo (ii) vlastnictví padesát procent (50%) nebo více z
nesplacené akcie, nebo (iii) prospěšné vlastnictví tohoto subjektu.

"Vy" (nebo "Vaše") se rozumí fyzická osoba nebo právnická osoba
výkonu oprávnění udělených touto licencí.

Formulář "Zdroj" znamená doporučený formulář pro provádění změn,
včetně, ale ne pouze na zdrojový kód softwaru, dokumentace
zdrojové soubory a konfigurační soubory.

Forma "Objekt" znamená jakoukoli formu vyplývající z mechanického
transformace nebo překlad zdrojového formuláře, včetně
nejsou omezeny na kompilovaný kód objektu, generovanou dokumentaci,
a převody na jiné typy médií.

'Práce' znamená práci autorství, ať již ve zdroji, nebo
Formulář objektu, který je dostupný na základě licence, jak je uvedeno v
upozornění na autorská práva, která jsou zahrnuta do práce nebo k ní připojena
(příklad je uveden v dodatku).

"Odvozené dílo" znamená veškerou práci, ať již ve zdroji nebo objektu
formulář, který je založen na (nebo odvozeném z) práce a pro které
redakční revize, anotace, rozpracování nebo jiné úpravy
představují jako celek původní práci autorství. Pro účely
z této licence, Derivative Works nesmí obsahovat díla, která zůstávají
oddělitelné od, nebo pouze odkaz (nebo vazba podle názvu) na rozhraní,
práce a odvození díla z nich.

"Příspěvek" se rozumí jakákoli práce autorství, včetně
původní verzi práce a jakýchkoli úprav nebo dodatků
za účelem práce nebo odvozování z nich, které je úmyslně
předáno poskytovateli licence k zahrnutí do práce vlastníkem autorských práv
nebo osobou nebo právnickou osobou oprávněnou k odeslání jménem
vlastníka autorských práv. Pro účely této definice byla "předložena"
znamená jakoukoli formu elektronické, verbální nebo písemné komunikace.
na poskytovatele licence nebo jeho zástupce, včetně-a nikoli však pouze-
komunikace v elektronických seznamech elektronické pošty, systémy řízení zdrojového kódu,
a zadejte systémy sledování, které jsou spravovány správou nebo jejím jménem.
Poskytovatel licence za účelem projednání a zlepšení práce, ale
vyloučení komunikace, která je nápadně označena nebo jinak
je označen jako "Not a Contribution" jako "Not a Contribution".

"Přispěvatel" znamená Poskytovatele licence a jakýkoli jednotlivec nebo právní subjekt
jménem uživatele, který obdržel příspěvek od poskytovatele licence, a
následně začleněny do práce.

2. Udělení licence pro autorská práva. S výhradou ustanovení a podmínek
Tato licence, každý přispěvatel tímto uděluje Perpetual,
po celém světě, bez-výhradní, bezplatné, bez poplatku, neodvolatelný, neodvolatelný

autorská práva k reprodukci, příprava derivátů, stavební práce
veřejné zobrazení, veřejné provádění, sublicence a distribuce
Práce a tato odvozená díla ve formuláři Zdroj nebo Objekt.

3. Udělení patentové licence. S výhradou ustanovení a podmínek
Tato licence, každý přispěvatel tímto uděluje Perpetual,
po celém světě, bez-výhradní, bezplatné, bez poplatku, neodvolatelný, neodvolatelný
(kromě výjimek uvedených v tomto oddíle) patentová licence, která má být provedena,
použití, nabízení k prodeji, prodeji, dovozu a jiné převedení práce,
pokud se taková licence vztahuje pouze na tyto patentové nároky licencovatelné
tímto přispěvatel, který je nezbytně porušen jejich
Pouze příspěvek (příspěvky) nebo kombinace jejich příspěvků
s prací, na které byl tento příspěvek (y) odeslán. Používáte-li
zavést patentový spor proti jakémukoli subjektu (včetně
křížový nárok nebo protinávrh v rámci žaloby) o tom, že práce
nebo příspěvek začleněn do práce je přímý
nebo příspěvkové porušení patentu, pak jakékoliv patentové licence
udělené Vám na základě této licence za tuto práci zaniknou
ode dne podání tohoto soudního sporu.

4. Redistribuce. Můžete kopírovat a distribuovat kopie těchto kopií
Práce nebo odvozená díla z nich v jakémkoliv médiu, s nebo bez
úpravy, a ve formuláři Zdroj nebo Objekt, za předpokladu, že jste
splňují následující podmínky:

(a) Musíte poskytnout jakékoliv jiné příjemce práce nebo
Derivative Works a kopie této licence; a

(b) Musíte způsobit, že všechny změněné soubory budou mít významné poznámky.
který uvádí, že jste změnili soubory; a

(c) Je třeba zachovat, ve zdrojové podobě všech odvozených děl
distribuci, veškerá autorská práva, patenty, ochranné známky a
oznámení o započtení ze zdrojové formy práce,
vyloučení těchto upozornění, která se netýkají žádné části
Derivativní práce a

d) Pokud práce zahrnuje textový soubor "OZNÁMENÍ" jako součást jeho
distribuce, pak všechny odvozené práce, které distribuujete, musí být
obsahovat čitelnou kopii obsažených oznámení o přiřazení
v takovém souboru NOTICES, kromě těch, které si nevšimnete
patří do kterékoli části derivátů, alespoň v jedné
z následujících míst: v rámci distribuovaného textového souboru NOTICE
v rámci Derivetive Works; v rámci formuláře Zdroj nebo
dokumentace, je-li poskytována spolu s odvození díla, nebo
v rámci zobrazení generovaného odvozeným odbytím, pokud a
Kdykoli se taková oznámení třetích stran obvykle objevují. Obsah
souboru NOTICE je pouze pro informativní účely a
neupravujte licenci. Můžete přidat vlastní přisouzení
upozornění v rámci Odvozených děl, které distribuujete, spolu
nebo jako dodatek k textu OZNÁMENÍ z práce, za předpokladu, že
že taková dodatečná oznámení o započtení nemohou být vykládána
jako úprava licence.

Můžete přidat své vlastní copyrightové prohlášení na své změny a
mohou poskytovat další nebo různé licenční podmínky
pro použití, reprodukci nebo distribuci Vašich úprav, nebo
veškeré takové stavební práce jako celku, za předpokladu, že používáte Vaše použití,

Výcvik a distribuce práce, která je jinak v souladu s
podmínky uvedené v této licenci.

5. Odeslání příspěvků. Není-li výslovně uvedeno jinak,
jakýkoli příspěvek záměrně předložený k zařazení do práce
Poskytovateli licence je za podmínek a podmínek stanovených
této licence bez jakýchkoli dodatečných podmínek.
Bez ohledu na výše uvedené skutečnosti zde nic nenahrazuje ani neupravuje.
podmínky jakékoli samostatné licenční smlouvy, kterou jste mohli provést
s Poskytovatelem licence týkající se takových příspěvků.

6. Ochranné známky. Tato licence neuděluje oprávnění pro použití obchodu
Jména, ochranné známky, servisní značky nebo názvy produktů Poskytovatele licence,
s výjimkou případů, kdy je to nezbytné pro přiměřené a obvyklé použití při popisování
původ práce a reprodukce obsahu souboru NOTICE.

7. Vyloučení záruky. Není-li to vyžadováno příslušnými právními předpisy nebo
souhlas s písemnou prací, Poskytovatelem licence je práce (a každý
Přispěvatel poskytuje své příspěvky) na bázi "JAK JE" (AS-IS),
BEZ ZÁRUK NEBO PODMÍNEK JAKÉHOKOLIV DRUHU, buď expresní,
odvozené, včetně, bez omezení, jakýchkoli záruk nebo podmínek
TITLE, NEPORUŠOVÁNÍ PRÁV TŘETÍCH STRAN, ZÁRUKY PRODEJNOSTI NEBO VHODNOSTI PRO
URČITÝ
ZVLÁŠTNÍ ÚČEL. Jste výhradně odpovědni za určení
vhodnost použití nebo opětovného distribuce práce a převzetí
rizika spojená s Vaším výkonem oprávnění na základě této licence.

8. Omezení odpovědnosti. V žádném případě a podle žádné právní teorie,
též ve stavu tort (včetně nedbalosti), smlouvy nebo jinak,
pokud to není vyžadováno příslušnými právními předpisy (například záměrné a hrubě
negligé akty) nebo se písemně dohodnou na tom, že se jedná o přispěvatele.
odpovědnost za škody, včetně všech přímých, nepřímých, zvláštních,
náhodné nebo následné škody vzniklé jakýmkoliv znakem, který vzniká jako
výsledek této licence nebo z použití nebo nemožnosti použít
Práce (včetně škod na ztrátě goodwillu, ale ne omezena)
pracovní zastávka, selhání počítače nebo chybná funkce, nebo všechny a všechny
jiné obchodní škody nebo ztráty), i když takový přispěvatel
byla upozorněna na možnost těchto škod.

9. Přijetí záruky nebo další odpovědnosti. Během redistribuce
dílo nebo odvození díla z nich, můžete se rozhodnout, že nabídku,
a účtovat poplatek za přijetí podpory, záruky, příslibu odškodnění,
nebo jiné povinnosti odpovědnosti a/nebo práva, která jsou v souladu s tímto
Licence. Při přijímání takových závazků však můžete jednat pouze
jménem vaší osoby a za výhradní odpovědnost, ne za jménem
jakéhokoli jiného přispěvatele, a to pouze tehdy, pokud souhlasíte s odškodněním,
obhájit a držet každého přispěvatele neškodného pro jakoukoli odpovědnost
vznikly žalobci nebo proti nim byly uplatněné námitky z důvodu
přijetí jakékoli takové záruky nebo dodatečné odpovědnosti.

KONEC PODMÍNEK

DODATEK: Jak použít licenci na produkt Apache pro vaši práci.

Chcete-li použít licenci na produkt Apache pro svou práci, připojte následující
upozornění na štítek s držáky, s poli ohraničenými hranatými závorkami "[]"
nahrazuje vašimi vlastními identifikačními informacemi. (Nezahrnujte
držáky!) Text by měl být ohraničen vhodným způsobem.

Syntaxe komentáře pro formát souboru. Také doporučujeme, aby
název souboru nebo třídy a popis účelu je obsažen v
Stejné "tištěné stránky" jako copyrightové výhrady pro usnadnění
identifikaci v archivech třetích stran.

Nastavení protokolu SOAP MQMD

Odesílatel protokolu SOAP IBM WebSphere MQ a modul listener protokolu SOAP produktu IBM WebSphere MQ SOAP vytvářejí deskriptor zprávy (**MQMD**). Toto téma popisuje pole, která musíte nastavit v deskriptoru MQMD, pokud vytváříte vlastní odesilatele SOAP nebo modul listener.

Účel

Hodnoty nastavené v produktu **MQMD** řídí výměnu zpráv mezi odesilatelem SOAP IBM WebSphere MQ , modulem listener SOAP IBM WebSphere MQ a klientským programem SOAP. Pokud vytváříte vlastní odesilatele SOAP nebo modul listener, postupujte podle pravidel v části Tabulka 584 na stránce 936.

Popis

Tabulka 584 na stránce 936 popisuje, jak jsou pole **MQMD** nastavena pomocí odesilatele IBM WebSphere MQ SOAP a modulu listener SOAP IBM WebSphere MQ . Pokud zapisujete svého vlastního odesílatele nebo modulu listener, musíte tato pole nastavit v souladu s pravidly pro výměnu zpráv. Modul listener protokolu SOAP IBM WebSphere MQ odpovídá typickým protokolům výměny zpráv produktu IBM WebSphere MQ . Pokud zapíšete vlastního odesílatele pro práci s moduly listener protokolu SOAP produktu IBM WebSphere MQ , můžete nastavit různé hodnoty **MQMD** .

V produktu Tabulka 584 na stránce 936jsou hodnoty ve sloupci Nastavení uspořádány následujícím způsobem:

Požadavek, jednosměrný
Nastavení provedená odesilatelem SOAP IBM WebSphere MQ .

Odpověď, zpráva
Nastavení provedená modulem listener protokolu SOAP produktu IBM WebSphere MQ v odezvě na požadavek odesilatele SOAP IBM WebSphere MQ .

ALL
Nastavení provedená odesílatelem SOAP IBM WebSphere MQ a modulem listener SOAP produktu IBM WebSphere MQ .

Vlastní odesílatel
Můžete napsat vlastního odesílatele. Přizpůsobený odesilatel obvykle přepíše standardní volby sestavy.

Tabulka 584. Nastavení SOAP MQMD		
Název pole	**Nastavení**	**Hodnoty**
StrucId	**ALL** MQMD_STRUC_ID	'MD¬¬'[1]
Version	**ALL** MQMD_VERSION_2	2
Report	**ALL** MQRO_NONE + MQRO_NEW_MSG_ID + MQRO_COPY_MSG_ID_TO_CORREL_ID + MQRO_EXCEPTION + MQRO_EXPIRY + MQRO_DISCARD **Vlastní odesílatel** Viz "Volby upravené sestavy" na stránce 940.	52428800
MsgType	**Požadavek** MQMT_REQUEST **Odezva** MQMT_REPLY **Sestava** MQMT_REPORT **Jednosměrné** MQMT_DATAGRAM	**MQMT_REQUEST** 1 **MQMT_REPLY** 2 **MQMT_REPORT** 4 **MQMT_DATAGRAM** 8
Expiry	**Požadavek, jednosměrný** Zadává se volbou Vypršení platnosti v identifikátoru URI. Výchozí hodnota je MQEI_UNLIMITED. **Odezva** Hodnota volby Vypršení platnosti ve zprávě požadavku **Sestava** MQEI_UNLIMITED	**MQEI_UNLIMITED** -1

Tabulka 584. Nastavení SOAP MQMD (pokračování)

Název pole	Nastavení	Hodnoty
Feedback	**Požadavek, odezva, jednosměrný** MQFB_NONE. **Sestava** • Generováno podle správce front- hodnota je nastavena podle běžných pravidel. • Generováno modulem listener protokolu SOAP produktu IBM WebSphere MQ : **MQRC_BACKOUT_THRESHOLD_REACHED** Byla překročena prahová hodnota vrácení pro více pokusů. **MQRCCF_MD_FORMAT_ERROR** Zpráva není rozpoznána jako záhlaví v záhlaví MQRFH2 . **MQRC_RFH_PARM_MISSING** Povinný parametr, například, SoapAction, v produktu **MQRFH2** chybí. **MQRC_RFH_FORMAT_ERROR** Kontrola základní integrity produktu **MQRFH2** se nezdařila, například vnitřní délky jsou poškozené. **MQRC_RFH_ERROR** Produkt **MQRFH2** prošel kontrolou integrity, ale tělo zprávy není nastaveno na MQFMT_NONE.	**MQFB_NONE** 0 **MQRC_BACKOUT_THRESHOLD_REACHED** 2362 **MQRCCF_MD_FORMAT_ERROR** 3023 **MQRC_RFH_PARM_MISSING** 2339 **MQRC_RFH_FORMAT_ERROR** 2421 **MQRC_RFH_ERROR** 2334
Encoding	**ALL** MQENC_NATIVE	Závisí na prostředí
CodedCharSetId	**ALL** Nastavit na UTF-8	1208
Format	**Požadavek, odezva, jednosměrný** MQFMT_RF_HEADER_2 **Sestava** **Sestavy správce front** Sleduje pravidla IBM WebSphere MQ **Sestavy modulu listener SOAP produktu IBM WebSphere MQ** Formát původní zprávy požadavku.	**MQFMT_RF_HEADER_2** "MQRFH2 "

Tabulka 584. Nastavení SOAP MQMD (pokračování)		
Název pole	**Nastavení**	**Hodnoty**
Priority	**Požadavek, jednosměrný** Určeno volbou Priorita v identifikátoru URI. Výchozí hodnota je MQPRI_PRIORITY_AS_Q_DEF. **Odpověď, zpráva** Hodnota Priorita ve zprávě požadavku.	**MQPRI_PRIORITY_AS_Q_DEF** -1
Persistence	**Požadavek, jednosměrný** MQPER_PERSISTENCE_AS_Q_DEF. **Odpověď, zpráva** Hodnota parametru Perzistence ve zprávě požadavku.	**MQPER_PERSISTENCE_AS_Q_DEF** 2
MsgId	**Požadavek, jednosměrný** Generováno správcem front. **Odpověď, zpráva** Vygeneruje se sady odesilatele SOAP IBM WebSphere MQ MQRO_NEW_MSG_ID a *MsgId* .	**Generováno** Jedinečná hodnota vygenerovaná správcem front
CorrelId	**Žádost, jednosměrný, Sestava** MQCI_NONE **Odpověď, zpráva** Odesilatel SOAP produktu IBM WebSphere MQ MQRO_COPY_MSG_ID_TO_CORREL_ID a modul listener zkopíruje ze zprávy požadavku *MsgId* .	**MQCI_NONE** 0
BackoutCount	**ALL** Nepoužito	0
ReplyToQ	**Požadavek** Určen volbou replyDestination v identifikátoru URI. Výchozí hodnota je SYSTEM.SOAP.RESPONSE.QUEUE. **Odezva, jednosměrný, Sestava** Vlevo prázdné	
ReplyToQMgr	**ALL** Pole ponecháno prázdné	Generovaný správcem front; viz téma Fronta pro odpovědi a správce front .
UserIdentifier	**Žádost, jednosměrný, Sestava** Vlevo prázdné **Odezva** Závisí na volbě -i *passContext* dodaném modulu listener a na oprávnění, pod kterým běží modul listener.	**Žádost, jednosměrný, Sestava** Generovaný správcem front; viz “UserIdentifier (MQCHAR12)” na stránce 425 **Odezva** *Proměnná*

Tabulka 584. Nastavení SOAP MQMD (pokračování)

Název pole	Nastavení	Hodnoty
AccountingToken	**ALL** MQACT_NONE	**MQACT_NONE** Nulový řetězec nebo prázdné znaky Nastaveno správcem front; viz “AccountingToken (MQBYTE32)” na stránce 386
ApplIdentityData	**ALL** Není	Řetězec s hodnotou null nebo prázdné znaky[2]
PutApplType	**ALL** MQAT_NO_CONTEXT	**MQAT_NO_CONTEXT** 0 Hodnota vygenerovaná správcem front; viz “Typ PutAppl(MQLONG)” na stránce 412.
PutApplName	**ALL** Není	Hodnota vygenerovaná správcem front; viz “Název funkce PutAppl(MQCHAR28)” na stránce 411.
PutDate	**ALL** Není	Hodnota vygenerovaná správcem front; viz “PutDate (MQCHAR8)” na stránce 414.
PutTime	**ALL** Není	Hodnota vygenerovaná správcem front; viz “PutTime (MQCHAR8)” na stránce 414.
ApplOriginData	**ALL** Není	Řetězec s hodnotou null nebo prázdné znaky[2]
GroupId	**Žádost, jednosměrný, Sestava** MQGI_NONE **Odezva** Pole je zkopírováno ze zprávy požadavku	Hodnoty null
MsgSeqNumber	**Žádost, jednosměrný, Sestava** Nepoužito **Odezva** Pole je zkopírováno ze zprávy požadavku	Generovaný správcem front; viz Fyzické pořadí ve frontě.
Offset	**Žádost, jednosměrný, Sestava** Nepoužito **Odezva** Pole je zkopírováno ze zprávy požadavku	0
MsgFlags	**Žádost, jednosměrný, Sestava** MQMF_NONE **Odezva** Pole je zkopírováno ze zprávy požadavku	**MQMF_NONE** 0 Viz “MsgFlags (MQLONG)” na stránce 402.

Tabulka 584. Nastavení SOAP MQMD (pokračování)		
Název pole	**Nastavení**	**Hodnoty**
OriginalLength	**Požadavek, jednosměrný, Odezva** MQOL_UNDEFINED **Sestava** Délka původní zprávy požadavku	**MQOL_UNDEFINED** -1

Notes:

1. Symbol ¬ představuje jeden prázdný znak.
2. Hodnota Nulový řetězec nebo prázdné znaky označuje řetězec s hodnotou null v jazyce C a prázdné znaky v jiných programovacích jazycích.

Volby upravené sestavy

Můžete napsat vlastní odesilatele SOAP a použít jej s dodanými moduly listener. Odesilatel obvykle můžete napsat, aby změnil volbu voleb sestavy. Moduly listener protokolu SOAP produktu IBM WebSphere MQ podporují většinu kombinací voleb sestav, jak je popsáno v následujících seznamech.

- Volby sestavy podporované moduly listener produktu IBM WebSphere MQ SOAP:
  - MQRO_EXCEPTION
  - MQRO_EXCEPTION_WITH_DATA
  - MQRO_EXCEPTION_WITH_FULL_DATA
  - MQRO_DEAD_LETTER_Q
  - MQRO_DISCARD_MSG
  - MQRO_NONE
  - MQRO_NEW_MSG_ID
  - MQRO_PASS_MSG_ID
  - MQRO_COPY_MSG_ID_TO_CORREL_ID
  - MQRO_PASS_CORREL_ID
- Volby sestavy podporované správcem front:
  - MQRO_COA
  - MQRO_COA_WITH_DATA
  - MQRO_COA_WITH_FULL_DATA
  - MQRO_COD
  - MQRO_COD_WITH_DATA
  - MQRO_COD_WITH_FULL_DATA
  - MQRO_EXPIRATION
  - MQRO_EXPIRATION_WITH_DATA
  - MQRO_EXPIRATION_WITH_FULL_DATA
- The following report options are not supported by the IBM WebSphere MQ SOAP listeners.
  - MQRO_PAN
  - MQRO_NAN

Chování modulů listener protokolu SOAP IBM WebSphere MQ v odezvě na kombinace produktů MQRO_EXCEPTION_* a MQRO_DISCARD je popsáno v tématu Tabulka 585 na stránce 941.

Notace MQRO_EXCEPTION_* označuje použití buď MQRO_EXCEPTION, MQRO_EXCEPTION_WITH_DATA nebo MQRO_EXCEPTION_WITH_FULL_DATA.

Tabulka 585. Chování modulu listener, které je výsledkem nastavení MQRO_EXCEPTION_* a MQRO_DISCARD		
	MQRO_DISCARD povoleno	**MQRO_DISCARD Nepovoleno**
MQRO_EXCEPTION_* povoleno	Výchozí chování. Zprávy sestavy se automaticky vygenerují, pokud je to nezbytné, a původní požadavek je vyřazen. Pokud se zpráva s hlášením nemůže vrátit do fronty odezev, zpráva se odešle do fronty nedoručených zpráv.	Zprávy sestavy se automaticky vygenerují, pokud je to nezbytné, a původní zpráva se odešle do fronty nedoručených zpráv. Pokud se zpráva sestavy nemohla vrátit do fronty odezvy, odešle se také do fronty nedoručených zpráv. V tomto případě jsou zde dva záznamy fronty nedoručených zpráv pro selhaný požadavek.
MQRO_EXCEPTION_* Nepovoleno	Zprávy sestavy se automaticky nevygenerují, když není rozpoznán příchozí formát nebo je překročen počet pokusů o vrácení. Zpráva se neodešle do fronty nedoručených zpráv. Nevrátí se žádné oznámení, které klient může zkontrolovat, a původní zpráva požadavku je ztracena.	Zprávy sestavy se automaticky nevygenerují, když není rozpoznán příchozí formát nebo je překročen počet pokusů o vrácení. Původní zpráva požadavku se však zapíše do fronty nedoručených zpráv, pokud by jinak byla generována sestava.

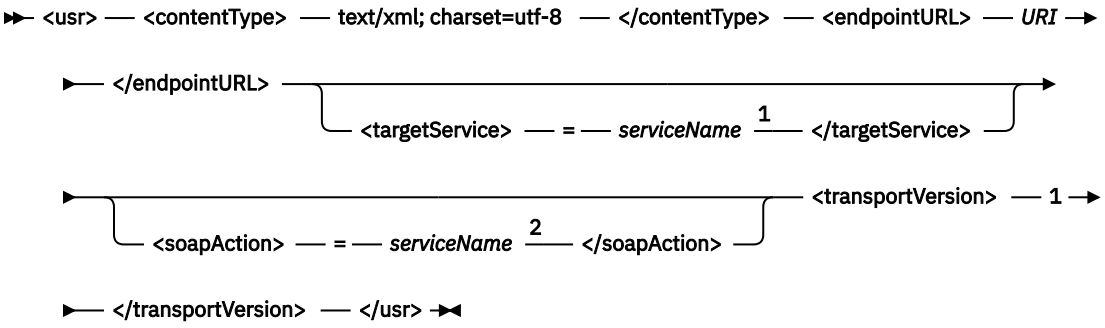
Nastavení protokolu SOAP MQRFH2

The IBM WebSphere MQ SOAP senders and listeners create or expect to receive an MQRFH2 with the following settings.

Účel

Produkt WebSphere MQ SOAP odesilatelé přidávají vlastnosti do složky produktu <usr> vytvořené produktem WebSphere MQ JMS. Vlastnosti obsahují informace požadované kontejnerem SOAP v cílovém prostředí. “Syntaxe vlastnosti” na stránce 941 popisuje syntaxi vlastností, když jsou přidány do MQRFH2. Popis záhlaví MQRFH2 viz MQRFH2 -Pravidla a formátovací záhlaví 2.

Syntaxe vlastnosti



Poznámky:

[1] Parametr targetService je vyžadován pro prostředí .NET Framework 1 nebo 2 a nepoužívá se v rámci osy 1.4.

[2] Položka soapAction je volitelná pro prostředí .NET Framework 1 nebo 2 a nepoužívá se u Axis 1.4.

Parametry

contentType
contentType vždy obsahuje řetězec `text/xml; charset=utf-8`.

endpointURL
Viz “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

targetService
[6]Na ose Axis je *serviceName* úplný název služby produktu Java , například: `targetService=javaDemos.service.StockQuoteAxis`. Není-li parametr targetService zadán, bude služba načtena s použitím výchozího mechanizmu Axis.

[7]V prostředí .NET je názvem *serviceName* název služby .NET umístěný v adresáři implementace, například: `targetService=myService.asmx`. V prostředí .NET umožňuje parametr targetService , aby jeden modul listener protokolu SOAP produktu WebSphere MQ mohl zpracovávat požadavky na více služeb. Tyto služby musí být implementovány ze stejného adresáře.

soapAction

transportVersion
Hodnota transportVersion je vždy nastavena na hodnotu 1.

Příklad

Příklad ukazuje MQRFH2 a následující zprávu SOAP. Délky složek jsou zobrazeny v desítkovém tvaru.

Poznámka: & v identifikátoru URI je kódováno jako &

```
52464820 00000002 000002B0 00000001 RFH 0002 1208 0001
000004B8 20202020 20202020 00000000 1208 0000
000004B8                           1208
32  <mcd>
        <Msd>jms_bytes</Msd>
    </mcd>?
208 <jms>
        <Dst>queue://queue://SOAPJ.demos</Dst>
        <Rto>queue://WMQSOAP.DEMO.QM/SYSTEM.SOAP.RESPONSE.QUEUE</Rto>
        <Tms>1157388516465</Tms>
        <Cid>ID:000000000000000000000000000000000000000000000000</Cid>
        <Dlv>1</Dlv>
    </jms>
400 <usr>
        <contentType>text/xml; charset=utf-8</contentType>
        <transportVersion>1</transportVersion>
        <endpointURL>
            jms:/queue?destination=SOAPJ.demos@WMQSOAP.DEMO.QM
            &amp;connectionFactory=connectQueueManager(WMQSOAP.DEMO.QM)
                clientConnection(localhost%25289414%2529)
                clientChannel(TESTCHANNEL)
            &amp;replyDestination=SYSTEM.SOAP.RESPONSE.QUEUE
            &amp;initialContextFactory=com.ibm.mq.jms.Nojndi
        </endpointURL>
    </usr>
<?xml version="1.0" encoding="UTF-8"?>
    <soapenv:Envelope
            xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:xsd="https://www.w3.org/2001/XMLSchema"
            xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">
        <soapenv:Body>
            <ns1:getQuote
                    soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    xmlns:ns1="soap.server.StockQuoteAxis_Wmq">
                <in0 xsi:type="xsd:string">XXX</in0>
            </ns1:getQuote>
        </soapenv:Body>
    </soapenv:Envelope>
```

[6] Pouze služba Java
[7] Pouze služba .NET

runivt: přenos WebSphere MQ pro ověřovací test instalace protokolu SOAP

Testovací sada verifikace instalace (IVT) je poskytována s přenosem IBM WebSphere MQ pro SOAP. Produkt **runivt** spouští několik demonstračních aplikací a zajišťuje, aby prostředí bylo po instalaci správně nastaveno.

Účel

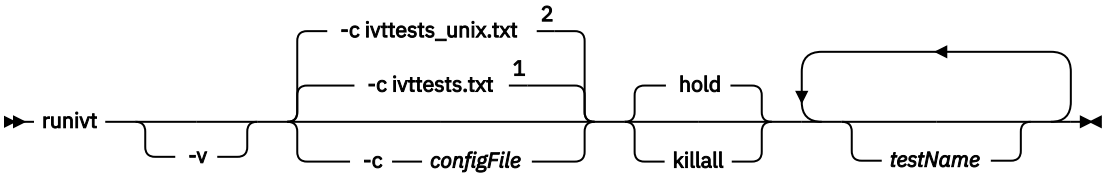
Příkaz **runivt** používá vzorové programy dodávané s přenosem WebSphere MQ pro protokol SOAP k odesílání požadavků webových služeb z klientů na služby. Spustí testy pro Axis 1.4, .NET Framework 1, a .NET Framework 2. Testy jsou nakonfigurovány v souboru testovacího skriptu. Soubor výchozího testovacího skriptu pro systém Windows spouští kombinaci testů mezi klienty Java a .NET a službami.

Popis

runivt musí být spuštěn ze svého vlastního adresáře.

Příkaz spustí listenery v jiném okně s příkazovým řádkem. Z tohoto důvodu musíte spustit příkaz z relace systému X Window System na systémech UNIX and Linux .

runivt syntax



Poznámky:

[1] Default on Windows
[2] Default on UNIX and Linux systems

runivt parametry

-v
Režim s komentářem. Zapište si podrobnější chybové zprávy na konzolu.

-c ***configFile***
Konfigurační soubor definující testy, které mají být spuštěny. Standardně je použit výchozí konfigurační soubor dodávaný s Windows, UNIX nebo Linux .

hold
Ponechat modul listener spuštěný po dokončení testů

killall
Ukončit modul listener po dokončení testů

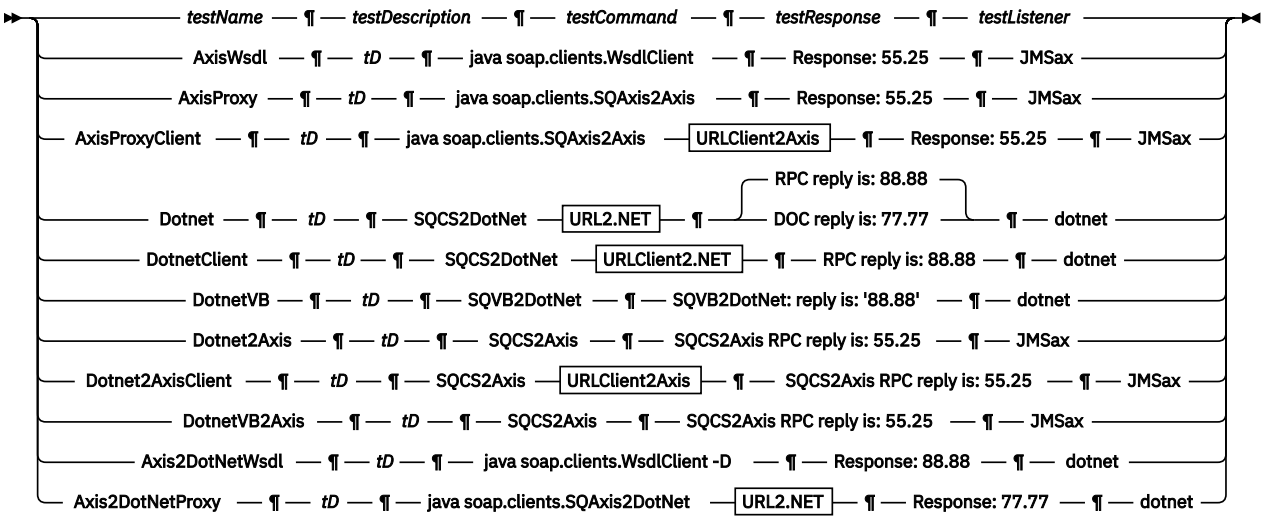
testName
Seznam testů, které mají být spuštěny, oddělený mezerami. Názvy testů jsou vybrány z konfiguračního souboru. Nejsou-li zadány žádné názvy, spustí se všechny testy v konfiguračním souboru.

Configuration file

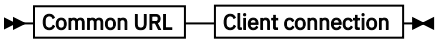
Each configuration file parameter is a separate line of the file. Leave a blank line between each group of parameters.

The parameters in the `ivttests.txt` parameter file are listed.

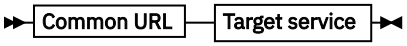
***configFile* syntax**



URLClient2Axis

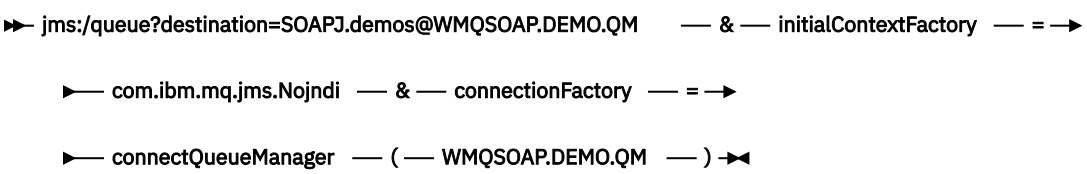


URL2.NET

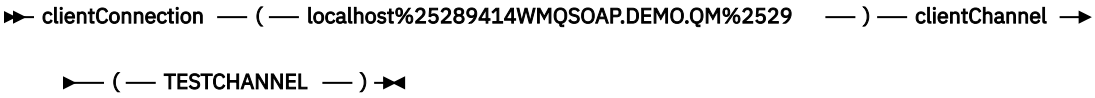


URLClient2.NET

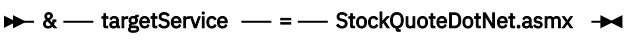
Common URL



Client connection



Target service



Parametry příkazu *configFile*

testName
Název testu. Použijte *testName* v příkazu **runivt**

testDescription
Documentation k testu

testCommand
Příkaz spuštěný příkazem **runivt** , aby provedl požadavek klienta.

testResponse
Přesný řetězec odpovědi vrácený požadavkem klienta na konzolu. Aby byl test úspěšný, *testResponse* se musí shodovat se skutečnou odezvou.

testListener
Název modulu listener protokolu SOAP produktu WebSphere MQ , který je spuštěn produktem **runivt** ke zpracování požadavku SOAP. dotnet a JMSax jsou synonyma pro dodané listenery, **amqwSOAPNETlistener** a **SimpleJavaListener**.

Příklady

```
runivt
```

Obrázek 19. spuštění všech výchozích testů

```
runivt dotnet
```

Obrázek 20. spustit specifický test z výchozích testů

```
runivt -c mytests.txt
```

Obrázek 21. spustit sadu vlastních testů

Související informace
Ověření transportu produktu WebSphere MQ pro protokol SOAP

Zabezpečit webové služby prostřednictvím transportu produktu IBM WebSphere MQ pro protokol SOAP

Můžete zabezpečit webové služby, které používají přenos produktu IBM WebSphere MQ pro protokol SOAP jedním ze dvou způsobů. Buď vytvořte kanál SSL mezi klientem a serverem, nebo použijte zabezpečení webových služeb.

SSL a přenos WebSphere MQ pro SOAP

Přenos protokolu SOAP produktu WebSphere MQ poskytuje několik voleb zabezpečení SSL, které lze zadat pro použití s kanálem klienta nakonfigurovaným pro spuštění v režimu SSL. Volby se liší mezi prostředími .NET a Java. Modul WebSphere MQ SOAP odesilatelů a listenerů zpracovává pouze volby zabezpečení SSL, které lze použít pro jejich konkrétní prostředí. Ignorují volby, které nejsou použitelné.

Přítomnost nebo nepřítomnost volby `sslCipherSpec` pro klienty .NET a volby `sslCipherSuite` pro klienty Java určuje, zda je zabezpečení SSL používáno či nikoli. Není-li tato volba uvedena v identifikátoru URI, pak se nepoužije výchozí SSL a všechny ostatní volby zabezpečení SSL se ignorují. Všechny volby zabezpečení SSL jsou volitelné, není-li uvedeno jinak.

Pro klienty WebSphere MQ nastavte atributy zabezpečení SSL v tabulce URI nebo v tabulce definic kanálů. Na serveru nastavte atributy pomocí zařízení produktu WebSphere MQ.

Standardně je při povolení zabezpečení SSL na kanálu nastavena standardní volba WebSphere MQ SSL, SKALCAUTH. Klienti se musí ověřit dříve, než může začít komunikace SSL. Není-li parametr SSLCAUTH nastaven, komunikace SSL se zavádí bez ověření klienta.

Chcete-li provést ověření sami, musí mít klienti certifikát přiřazený v úložišti klíčů, který je pro správce front přijatelný. Pro další zabezpečení je možné kanály produktu WebSphere MQ konfigurovat tak, aby přijímaly pouze certifikáty ze seznamu zakázaných položek. Tento seznam je omezen kontrolou rozlišujícího názvu certifikátu s atributem názvu partnera kanálu.

Pokud používáte jazyk Java, první připojení SSL z klienta WebSphere MQ SOAP způsobí, že budou opraveny následující parametry zabezpečení SSL. Stejné hodnoty jsou použity v následných připojeních pomocí stejného procesu klienta:

- `sslKeyStore`
- `sslKeyStorePassword`

- Úložiště`sslTrust`
- `sslTrustStorePassword`
- `SSLFIPSREQUIRED`
- `sslLDAPCRLservers`

Efekt změny těchto parametrů u následujících připojení z tohoto klienta není definován.

Pokud používáte prostředí .NET, první připojení SSL z klienta WebSphere MQ SOAP způsobí, že budou opraveny následující parametry zabezpečení SSL. Stejné hodnoty jsou použity v následných připojeních pomocí stejného procesu klienta:

- `SSLKeyRepository`
- `SSLCryptoHardware`
- `SSLFIPSREQUIRED`
- `sslLDAPCRLservers`

Efekt změny těchto parametrů u následujících připojení z tohoto klienta není definován. Tyto parametry se resetují, pokud se všechna připojení SSL stanou neaktivními a vytvoří se nové připojení SSL.

Jako systémové vlastnosti lze zadat také následující vlastnosti:

- `sslKeyStore`
- `sslKeyStorePassword`
- Úložiště`sslTrust`
- `sslTrustStorePassword`

Jsou-li zadány jako systémové vlastnosti i v identifikátoru URI a hodnoty se liší, zobrazí obslužný program implementace varování. Hodnoty identifikátorů URI mají přednost.

Související úlohy

Určení, že pro běhové prostředí klienta MQI je použit pouze certifikovaný standard FIPS CipherSpecs

Související odkazy

Parametry továrny na připojení zabezpečení SSL v identifikátoru URI webových služeb produktu WebSphere MQ

Přidejte volby zabezpečení SSL do seznamu voleb továrny připojení v identifikátoru URI webových služeb produktu IBM WebSphere MQ .

Federální standardy zpracování informací (FIPS) pro UNIX, Linux a Windows

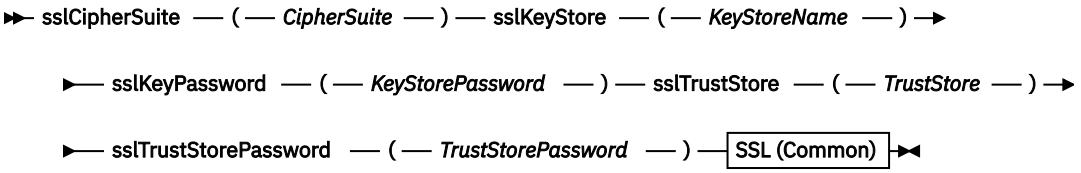
Parametry továrny na připojení zabezpečení SSL v identifikátoru URI webových služeb produktu WebSphere MQ

Přidejte volby zabezpečení SSL do seznamu voleb továrny připojení v identifikátoru URI webových služeb produktu IBM WebSphere MQ .

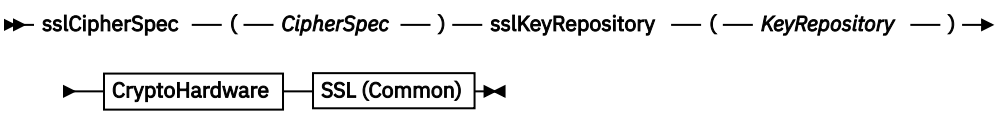
Účel

Zabezpečené připojení můžete použít mezi klientem webových služeb IBM WebSphere MQ a správcem front, který je hostitelem webové služby. Volby zabezpečení SSL řídí způsob konfigurace SSL na připojení ke kanálu klienta IBM WebSphere MQ MQI MQI.

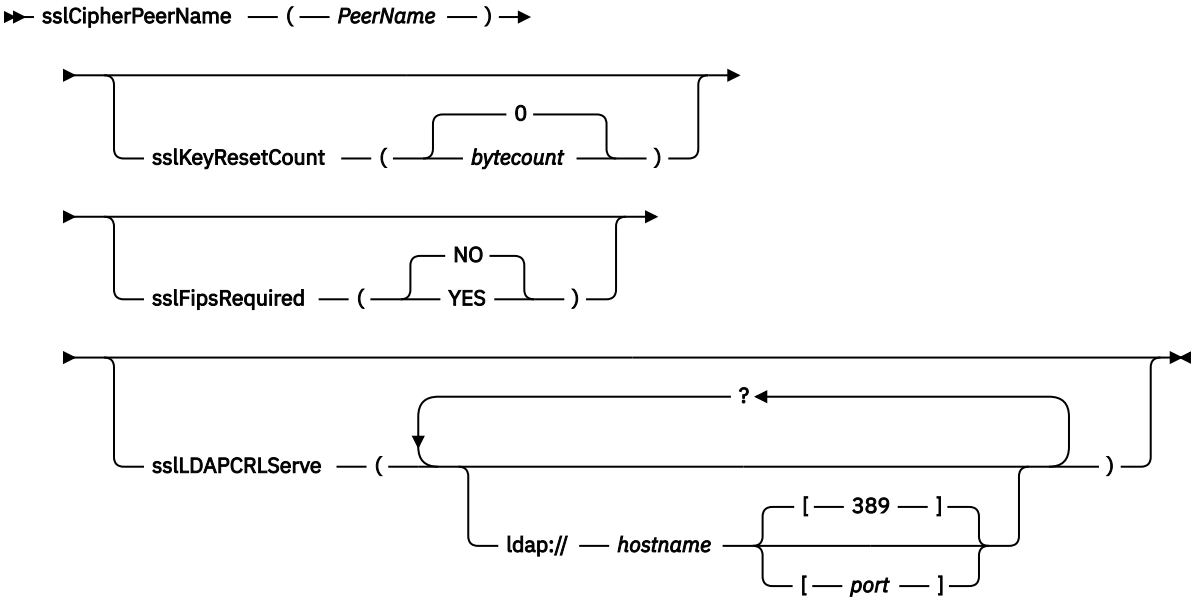
Syntax diagram
SSL (Java)



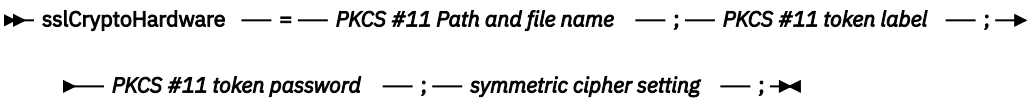
SSL (.NET)



SSL (Common)



CryptoHardware



Požadované parametry SSL (Common)

sslPeerName(*peerName*)
peerName uvádí sslPeerName použité na kanálu.

Požadované parametry SSL (Java)

sslCipherSuite(*CipherSuite*)
Volba *CipherSuite* uvádí sslCipherSuite použitou na kanálu. Volba CipherSuite určená klientem musí odpovídat sadě CipherSuite určené pro kanál připojení k serveru.

sslKeyStore(*KeyStoreName*)
KeyStoreName uvádí sslKeyStoreName použité na kanálu. Úložiště klíčů obsahuje soukromý klíč klienta použitého k ověření klienta na serveru. Úložiště klíčů je volitelné, pokud je připojení SSL konfigurováno tak, aby přijímaly anonymní připojení klienta.

sslKeyStorePassword(*KeyStoreHeslo*)
KeyStoreHeslo uvádí sslKeyStorePassword použité na kanálu.

sslTrustStore(*TrustStoreNázev*)
Hodnota *TrustStoreName* uvádí sslTrustStoreName použitou na kanálu. Úložiště údajů o důvěryhodnosti uchovává veřejný certifikát serveru nebo jeho klíčový řetězec k ověření serveru pro klienta. Úložiště údajů o důvěryhodnosti je volitelné, pokud se ke ověření serveru používá kořenový certifikát certifikační autority. V jazyce Java jsou certifikáty kořenových certifikátů uchovávány v úložišti certifikátů prostředí JRE, `cacerts`.

sslTrustStorePassword(*TrustStoreHeslo*)
Hodnota *TrustStorePassword* uvádí sslTrustStorePassword použité na kanálu.

Požadované parametry SSL (.NET)

sslCipherSpec(*CipherSpec*)
CipherSpec uvádí sslCipherSpec použité na kanálu. Je-li zadána volba, je na kanálu klienta použito zabezpečení SSL.

sslKeyRepository(*KeyRepository*)
KeyRepository uvádí sslCipherSpec použité na kanálu, kde jsou uloženy klíče SSL a certifikáty. *KeyRepository* je uveden ve stokovém formátu, tj. úplnou cestu se jménem souboru, ale s vynecháním přípony souboru. Vliv nastavení úložiště sslKeyRepository je stejný jako nastavení pole KeyRepository ve struktuře **MQSCO** na volání MQCONNX .

Volitelné parametry SSL (.NET)

sslCryptoHardware(*CryptoHardware*)
CryptoHardware uvádí sslCryptoHardware použitý na kanálu. Možné hodnoty pro toto pole a jeho nastavení jsou stejné jako u pole CryptoHardware struktury **MQSCO** na serveru MQCONNX.

Volitelné parametry SSL (Common)

sslKeyResetCount(*bytecount*)
Parametr *bytecount* určuje počet bajtů předávaných přes kanál SSL před opětovným vyjednáním tajného klíče zabezpečení SSL. Chcete-li zakázat opětovné vyjednávání klíčů SSL, vynechte toto pole nebo nastavte hodnotu nula. Nula je jediná hodnota podporovaná v některých prostředích, viz Opětovné dohadování tajného klíče ve třídách WebSphere MQ pro Javu. Efekt nastavení sslKeyResetCount je stejný jako nastavení pole KeyResetCount ve struktuře **MQSCO** na volání MQCONNX .

sslFipsRequired(*fipsCertified*)
fipsCertified uvádí, zda *CipherSpec* nebo *CipherSuite* musí v kanálu IBM WebSphere MQ na kanálu používat šifrovací mechanizmus certifikovaný FIPS. Efekt nastavení *fipsCertified* je stejný jako nastavení pole FipsRequired struktury **MQSCO** na volání MQCONNX .

sslLDAPCRLServers(*LDAPServerList*)

Položka *LDAPServerList* určuje seznam serverů LDAP, které mají být použity pro kontrolu seznamu odvolaných certifikátů (CRL).

U připojení klienta s povoleným SSL je *LDAPServerList* seznam serverů LDAP, které se mají použít pro kontrolu seznamu odvolaných certifikátů (CRL). Certifikát poskytnutý správcem front je kontrolován na jednom z vypsaných serverů LDAP CRL; pokud je nalezen, připojení selže. Každý server LDAP se vyzkouší, dokud není ustanoveno připojení k jednomu z nich. Pokud se nelze připojit k žádnému ze serverů, certifikát se odmítne. Po úspěšném navázání spojení s jedním z nich je certifikát přijat nebo zamítnut v závislosti na seznamech CRL přítomných na daném serveru LDAP.

Je-li parametr *LDAPServerList* prázdný, nebude certifikát náležící ke správci front kontrolován na základě seznamu odvolaných certifikátů. Je-li zadaný seznam identifikátorů URI protokolu LDAP neplatný, zobrazí se chybová zpráva. Efekt nastavení tohoto pole je stejný, jako je zahrnutí záznamů MQAIR a přístupu k nim ze struktury **MQSCO** na MQCONNX.

Související úlohy
Určení, že pro běhové prostředí klienta MQI je použit pouze certifikovaný standard FIPS CipherSpecs
Související odkazy
SSL a přenos WebSphere MQ pro SOAP
Přenos protokolu SOAP produktu WebSphere MQ poskytuje několik voleb zabezpečení SSL, které lze zadat pro použití s kanálem klienta nakonfigurovaným pro spuštění v režimu SSL. Volby se liší mezi prostředími .NET a Java. Modul WebSphere MQ SOAP odesilatelů a listenerů zpracovává pouze volby zabezpečení SSL, které lze použít pro jejich konkrétní prostředí. Ignorují volby, které nejsou použitelné.

Federální standardy zpracování informací (FIPS) pro UNIX, Linux a Windows

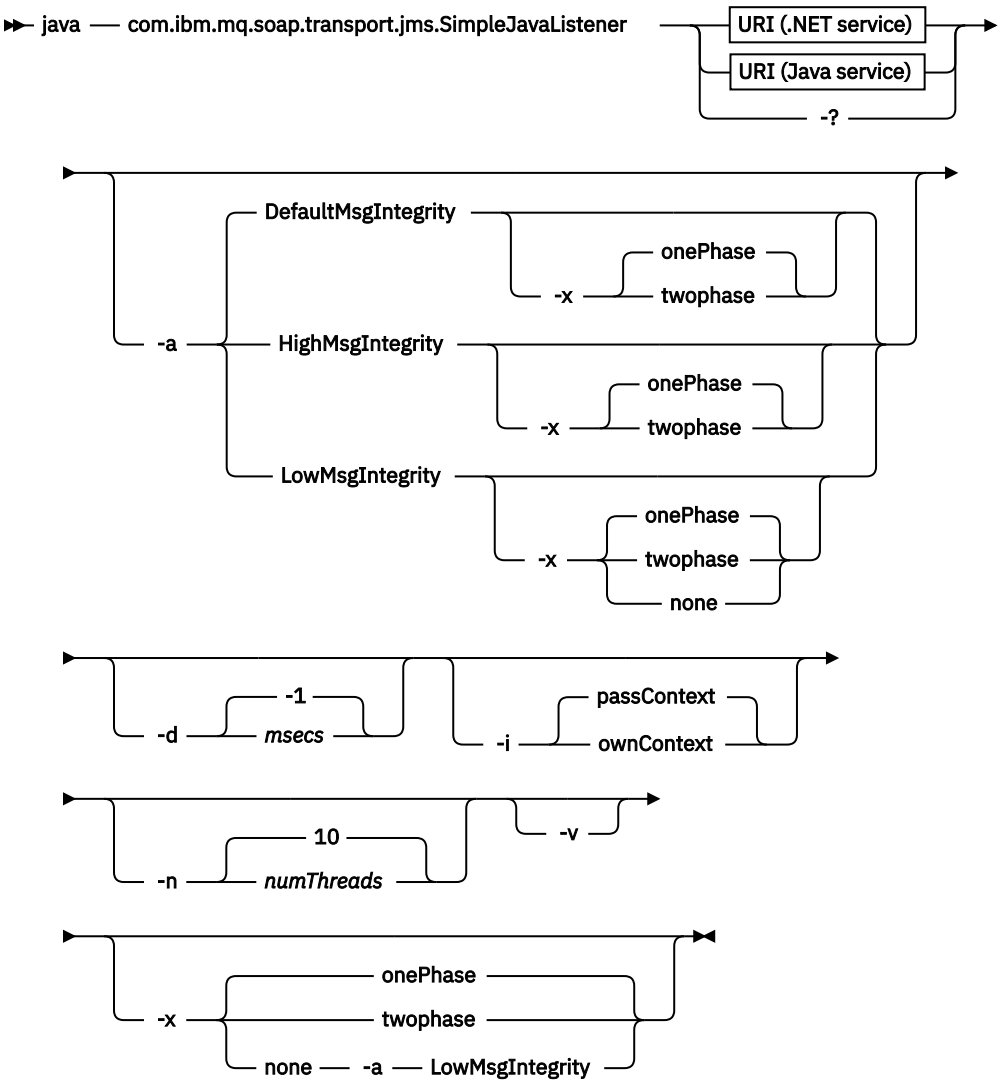
`SimpleJavaListener`: IBM WebSphere MQ Listener SOAP pro osu 1.4

Syntaxe a parametry pro modul listener protokolu SOAP produktu IBM WebSphere MQ pro osu 1.4.

Účel

Spustí modul listener SOAP produktu IBM WebSphere MQ pro osu 1.4.

Java



Povinné parametry

`URI` ***platforma***
Viz “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

`-?`
Vytisknout text nápovědy popisující způsob použití příkazu.

Nepovinné parametry

`-a` ***integrityOption***
Volba *integrityOption* určuje chování modulů listener protokolu SOAP produktu WebSphere MQ , pokud nelze odeslat zprávu s nezdarem do fronty nedoručených zpráv. *integrityOption* může mít jednu z následujících hodnot:

DefaultMsgIntegrity
Pro netrvalé zprávy modul listener zobrazí varovnou zprávu a pokračuje ve zpracování s vyřazenou původní zprávou. V případě trvalých zpráv zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena. DefaultMsgIntegrita se použije, pokud je vynechána volba -a , nebo není-li zadána volba *integrityOption* .

LowMsgIntegrity
U trvalých i dočasných zpráv modul listener zobrazí varování a pokračuje v provádění a zahození zprávy.

HighMsgIntegrity
V případě trvalých i dočasných zpráv modul listener zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Je-li zadána hodnota -x none , musí být zadána hodnota -a LowMsgIntegrity . Pokud jsou příznaky nekompatibilní, obslužný program implementace se ukončí s chybovou zprávou a bez kroků implementace, které byly provedeny.

-d ***ms***
Hodnota *ms* určuje počet milisekund pro modul listener protokolu SOAP produktu WebSphere MQ , který má být udržen v případě, že byly zprávy požadavků přijaty v libovolném podprocesu. Je-li parametr *ms* nastaven na hodnotu -1, modul listener zůstane naživu neomezeně dlouho.

-i ***Kontext***
Kontext určuje, zda listenery předávají kontext identity. *Kontext* má následující hodnoty:

passContext
Nastavte kontext identity původní zprávy požadavku do zprávy odpovědi. Modul listener SOAP kontroluje, zda má oprávnění uložit kontext z fronty požadavků a předat jej do fronty odpovědí. Kontextové kontroly provádějí při otevírání fronty požadavků do kontextu ukládání kontextu a ve frontě odpovědí pro předávání kontextu. Pokud nemá požadované oprávnění, nebo se volání MQOPEN nezdaří a zpráva odpovědi se nezpracuje. Zpráva odpovědi je vložena do fronty nedoručených zpráv s hlavičkou nedoručených zpráv, která obsahuje návratový kód z nezdaru MQOPEN. Modul listener poté pokračuje ve zpracování následných příchozích zpráv jako obvykle.

ownContext
Modul listener SOAP nepředává kontext. Vrácený kontext odráží ID uživatele, pod kterým je modul listener spuštěný, spíše než ID uživatele, který vytvořil původní zprávu požadavku.

Pole v kontextu původu jsou nastavena prostřednictvím správce front a nikoli modulem listener protokolu SOAP.

-n ***numThreads***
Volba *numThreads* určuje počet podprocesů v generovaných spouštěcích skriptech pro modul listener protokolu SOAP produktu WebSphere MQ . Výchozí hodnota je 10. Zvažte zvýšení tohoto čísla, pokud máte vysokou propustnost zpráv.

-v
-v nastaví podrobný výstup z externích příkazů. Chybové zprávy jsou vždy zobrazeny. Pomocí volby -v můžete vytvářet výstupní příkazy, které lze upravit, a vytvořit tak přizpůsobené skripty implementace.

-w ***serviceDirectory***
serviceDirectory je adresář obsahující webovou službu.

-x ***transakcionalita***
transactnost určuje typ transakčního řízení pro modul listener. *transakcionalitu* lze nastavit na jednu z následujících hodnot:

onePhase
IBM WebSphere MQ je použita jednofázová podpora. Pokud systém selže během zpracování, zpráva požadavku se znovu doručí do aplikace. Transakce WebSphere MQ zajistí, aby zprávy odezvy byly napsány přesně jednou.

twoPhase
Je použita dvoufázová podpora. Je-li služba zapsána správně, zpráva se doručí přesně jednou, koordinovanou s jinými prostředky, v rámci jediného potvrzeného provedení služby. Tato volba se vztahuje pouze na připojení vazeb serveru.

none
Žádná transakční podpora. Pokud dojde k selhání systému během zpracování, může dojít ke ztrátě zprávy požadavku i v případě, že je trvalá. Je možné, že služba byla nebo nemusela být provedena a že zprávy odezvy, sestavy nebo zprávy s deadutem mohou nebo nemusí být zapsány.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Podrobnosti najdete v popisu příznaku -a .

Příklad jazyka Java

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

Moduly listener protokolu SOAP produktu WebSphere MQ

Modul listener protokolu SOAP produktu WebSphere MQ čte příchozí požadavek SOAP od fronty určené jako místo určení v identifikátoru URI. Kontroluje formát zprávy požadavku a poté vyvolá webovou službu pomocí infrastruktury webových služeb. Modul listener protokolu SOAP produktu WebSphere MQ vrátí jakoukoli odezvu nebo chybu z webové služby pomocí fronty místa určení odpovědí v identifikátoru URI. Vrátíte sestavy produktu WebSphere MQ do fronty odpovědí.

Termín listener se používá zde ve standardním smyslu webových služeb. Je odlišený od standardního modulu listener produktu WebSphere MQ vyvolaného příkazem **runmqlsr** .

Popis

Modul listener Java SOAP je implementován jako třída Java a spouští služby pomocí Axis 1.4. Modul listener .NET je konzolou aplikací a spouští služby .NET Framework 1 nebo .NET Framework 2. V případě služeb .NET Framework 3 použijte vlastní kanál produktu WebSphere MQ pro Microsoft Windows Communication Foundation (WCF).

Obslužný program implementace vytváří skripty pro automatické spouštění modulů listener protokolu Java nebo .NET SOAP. Modul listener SOAP lze spustit ručně buď pomocí příkazu **amqSOAPNETListener** , nebo voláním třídy `SimpleJavaListener` . Modul listener protokolu SOAP produktu WebSphere MQ , který má být spuštěn jako služba WebSphere MQ , lze konfigurovat nastavením volby -s v obslužném programu implementace. Případně spusťte listenery pomocí spouštěče, nebo použijte spouštěcí a koncové skripty modulu listener generované obslužným programem implementace. Můžete nakonfigurovat spuštění spouštěče ručně, nebo můžete použít volby implementace -tmq a -tmp , chcete-li nakonfigurovat spouštění automaticky. Modul listener můžete ukončit tak, že nastavíte frontu požadavků na `GET(DISABLED)`.

Tabulka 586. Příkazové skripty generované obslužným programem implementace

Infrastruktura webových služeb	Systémy UNIX and Linux	Windows Java	Windows .NET
Spustit listener	**startWMQJListener.sh**	**startWMQJListener.cmd**	**startWMQNListener.cmd**
Ukončit listener	**endWMQJListener.sh**	**endWMQJListener.cmd**	**endWMQNListener.cmd**
Definování služby modulu listener	**defineWMQJListener.sh**	**defineWMQJListener.cmd**	**defineWMQNListener.cmd**

Modul listener protokolu SOAP produktu WebSphere MQ předá pole endpointURL a soapAction ze zprávy SOAP do infrastruktury SOAP. Modul listener vyvolá službu prostřednictvím infrastruktury webových služeb a vyčká na odezvu. Listener nevaliduje endpointURL a soapAction. Pole jsou nastavena odesilatelem SOAP produktu WebSphere MQ z dat, která jsou uvedena v identifikátoru URI nastavovaného klientem SOAP.

Modul listener vytvoří zprávu s odpovědí a odešle ji do místa určení odpovědí dodaném v identifikátoru URI zprávy požadavku. Kromě toho modul listener nastaví ID korelace ve zprávě odpovědi na základě volby sestavy ve zprávě požadavku. Vrací nastavení vypršení platnosti, perzistence a priority ze zprávy vzniklé při zpracování požadavku. Modul listener také za určitých okolností odesílá zprávy sestav zpět klientům.

Pokud v požadavku SOAP existují chyby formátu, modul listener vrátí klientovi zprávu s použitím fronty místa určení odpovědí. Správce front také vrací zprávy sestav klientovi pomocí fronty místa určení odpovědí, pokud byla požadována sestava. Úplné zprávy sestav jsou zapsány do fronty odpovědí jako odezva na určitý počet událostí:

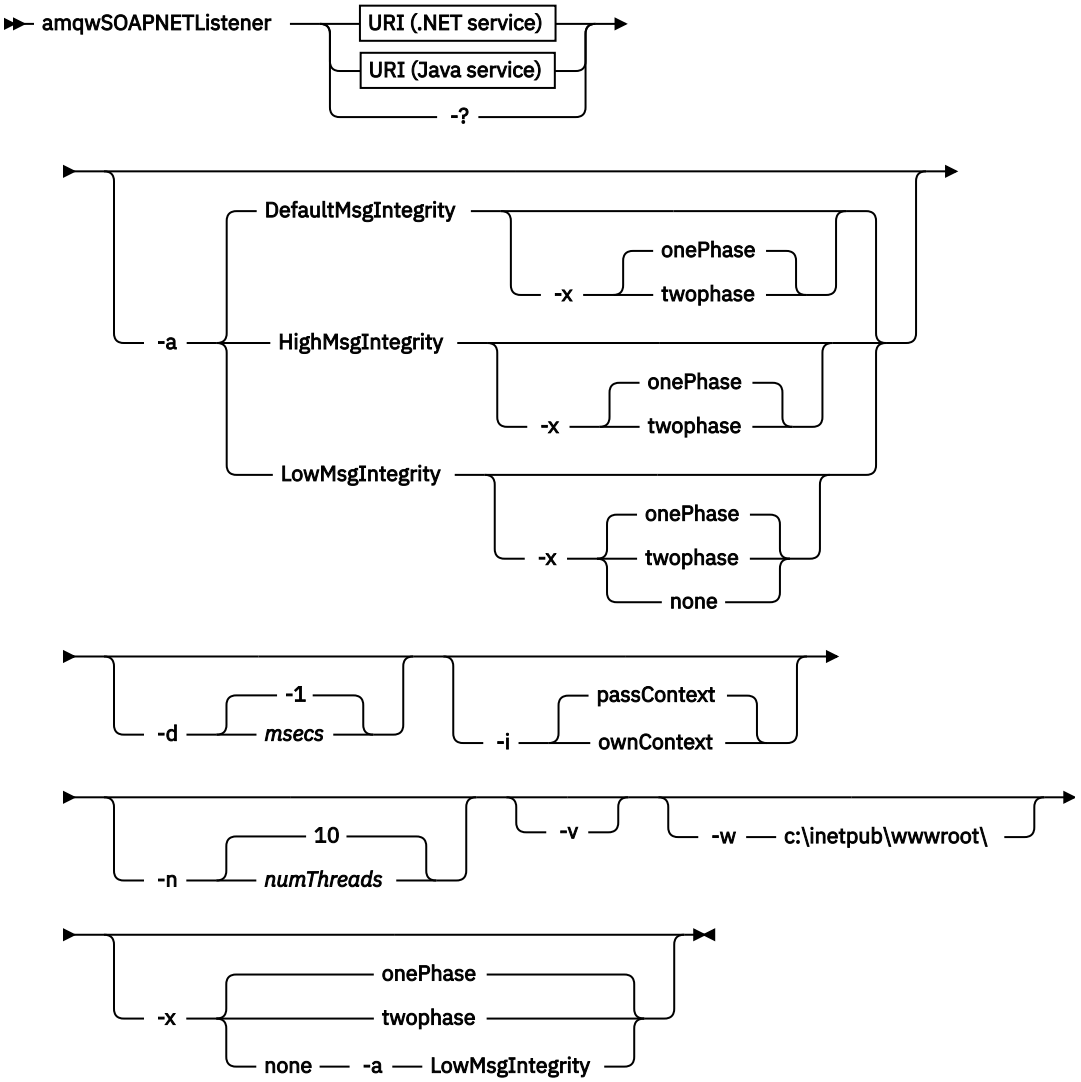
- Výjimka.
- Vypršení zprávy.
- Formát zprávy požadavku nebyl rozpoznán.
- Selhání kontroly integrity záhlaví **MQRFH2** .
- Formát těla hlavní zprávy není MQFMT_NONE.
- Prahová hodnota počtu odvolání je překročena, zatímco modul listener protokolu SOAP produktu WebSphere MQ zpracovává požadavek.

Odesilatel SOAP produktu WebSphere MQ nastavuje volby sestav MQRO_EXCEPTION_WITH_FULL_DATA a MQRO_EXPIRATION_WITH_FULL_DATA . Jako výsledek voleb sestavy nastavených odesilatelem protokolu SOAP produktu WebSphere MQ zpráva obsahuje celou zprávu o původním požadavku. Odesilatel protokolu SOAP produktu WebSphere MQ také nastaví volbu MQRO_DISCARD , která způsobí, že zpráva bude vyřazena po vrácení zprávy sestavy. Pokud volby sestavy nesplňují vaše požadavky, napište své vlastní odesilatele, aby mohli používat různé volby sestavy MQRO_EXCEPTION a MQRO_DISCARD . Je-li požadavek protokolu SOAP odeslán jiným odesílatelem, který nebyl nastaven na MQRO_DISCARD, bude se zpráva o selhání zapsána do fronty nedoručených zpráv (DLQ).

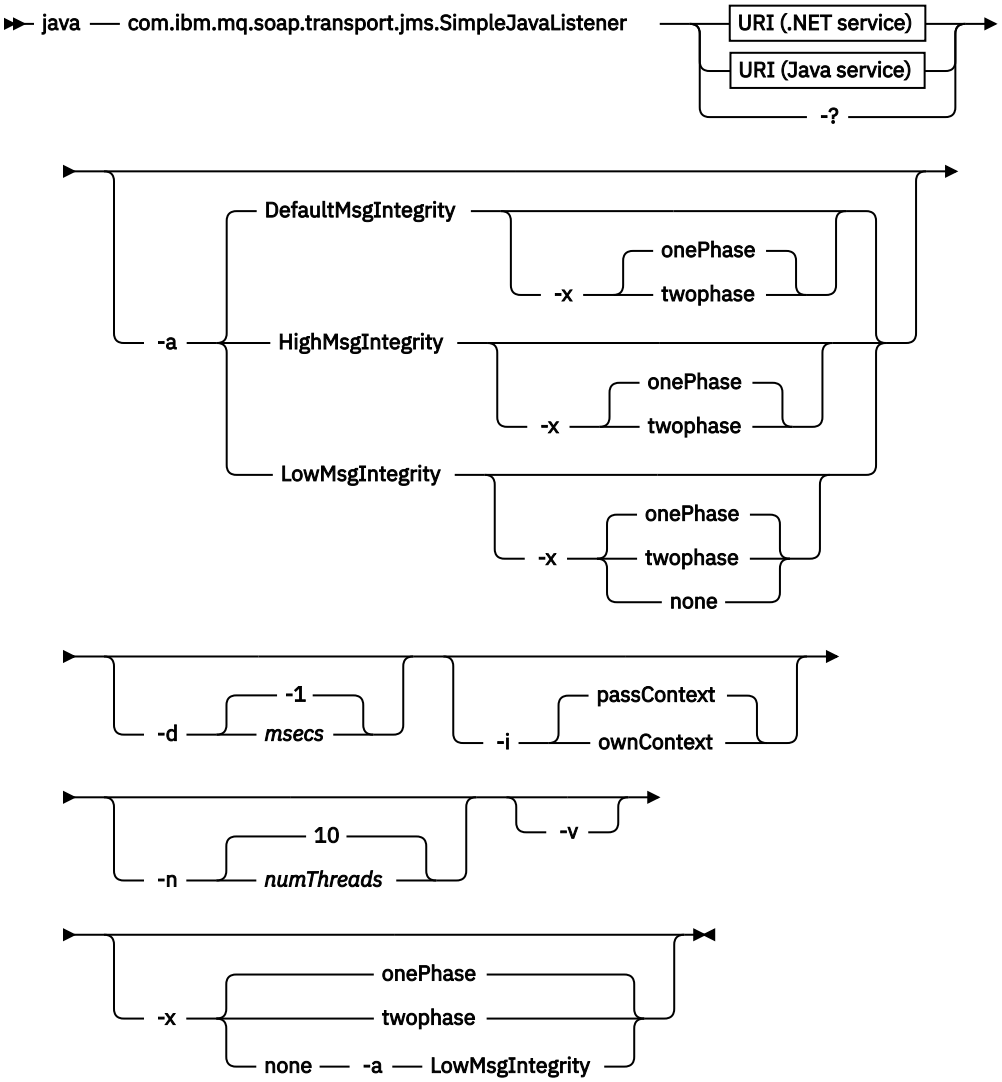
Pokud modul listener vygeneruje zprávu s hlášením, ale v procesu odeslání sestavy selže, odešle se zpráva sestavy do fronty nedoručených zpráv. Ujistěte se, že váš popisovač DLQ zpracovává tyto zprávy správně.

Pokud při pokusu o zápis do fronty nedoručených zpráv dojde k chybě při pokusu o zápis do fronty nedoručených zpráv, bude do protokolu chyb produktu WebSphere MQ zapsána zpráva. Zda bude modul listener pokračovat ve zpracování dalších zpráv, závisí na tom, která perzistence zpráv a transakční volby jsou vybrány. Pokud je modul listener spuštěn v jednofázovém transakčním režimu a zpracovává neperzistentní zprávu požadavku, bude původní zpráva zahozena. Modul listener protokolu SOAP produktu WebSphere MQ se nadále provádí. Je-li zpráva požadavku trvalá, zpráva požadavku je vrácena do fronty požadavků a dojde k ukončení modulu listener. Fronta požadavků je nastavena na získání-zablokováno, aby se zabránilo nechtěnému spuštění restartu.

Syntax diagram
.NET


amqwSOAPNETListener
URI (.NET service)
URI (Java service)
-?
-a
DefaultMsgIntegrity
-x
onePhase
twophase
HighMsgIntegrity
-x
onePhase
twophase
LowMsgIntegrity
-x
onePhase
twophase
none
-d
-1
msecs
-i
passContext
ownContext
-n
10
numThreads
-v
-w
c:\inetpub\wwwroot\
-x
onePhase
twophase
none
-a
LowMsgIntegrity

Java



Povinné parametry

URI ***platforma***
Viz “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

-?
Vytisknout text nápovědy popisující způsob použití příkazu.

Nepovinné parametry

-a ***integrityOption***
Volba *integrityOption* určuje chování modulů listener protokolu SOAP produktu WebSphere MQ , pokud nelze odeslat zprávu s nezdarem do fronty nedoručených zpráv. *integrityOption* může mít jednu z následujících hodnot:

DefaultMsgIntegrity
Pro netrvalé zprávy modul listener zobrazí varovnou zprávu a pokračuje ve zpracování s vyřazenou původní zprávou. V případě trvalých zpráv zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena. DefaultMsgIntegrita se použije, pokud je vynechána volba -a , nebo není-li zadána volba *integrityOption* .

LowMsgIntegrity
U trvalých i dočasných zpráv modul listener zobrazí varování a pokračuje v provádění a zahození zprávy.

HighMsgIntegrity
V případě trvalých i dočasných zpráv modul listener zobrazí chybovou zprávu a zazálohuje zprávu s požadavkem tak, aby zůstala ve frontě požadavků a byla ukončena.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Je-li zadána hodnota -x none , musí být zadána hodnota -a LowMsgIntegrity . Pokud jsou příznaky nekompatibilní, obslužný program implementace se ukončí s chybovou zprávou a bez kroků implementace, které byly provedeny.

-d ***ms***
Hodnota *ms* určuje počet milisekund pro modul listener protokolu SOAP produktu WebSphere MQ , který má být udržen v případě, že byly zprávy požadavků přijaty v libovolném podprocesu. Je-li parametr *ms* nastaven na hodnotu -1, modul listener zůstane naživu neomezeně dlouho.

-i ***Kontext***
Kontext určuje, zda listenery předávají kontext identity. *Kontext* má následující hodnoty:

passContext
Nastavte kontext identity původní zprávy požadavku do zprávy odpovědi. Modul listener SOAP kontroluje, zda má oprávnění uložit kontext z fronty požadavků a předat jej do fronty odpovědí. Kontextové kontroly provádějí při otevírání fronty požadavků do kontextu ukládání kontextu a ve frontě odpovědí pro předávání kontextu. Pokud nemá požadované oprávnění, nebo se volání MQOPEN nezdaří a zpráva odpovědi se nezpracuje. Zpráva odpovědi je vložena do fronty nedoručených zpráv s hlavičkou nedoručených zpráv, která obsahuje návratový kód z nezdaru MQOPEN. Modul listener poté pokračuje ve zpracování následných příchozích zpráv jako obvykle.

ownContext
Modul listener SOAP nepředává kontext. Vrácený kontext odráží ID uživatele, pod kterým je modul listener spuštěný, spíše než ID uživatele, který vytvořil původní zprávu požadavku.

Pole v kontextu původu jsou nastavena prostřednictvím správce front a nikoli modulem listener protokolu SOAP.

-n ***numThreads***
Volba *numThreads* určuje počet podprocesů v generovaných spouštěcích skriptech pro modul listener protokolu SOAP produktu WebSphere MQ . Výchozí hodnota je 10. Zvažte zvýšení tohoto čísla, pokud máte vysokou propustnost zpráv.

-v
-v nastaví podrobný výstup z externích příkazů. Chybové zprávy jsou vždy zobrazeny. Pomocí volby -v můžete vytvářet výstupní příkazy, které lze upravit, a vytvořit tak přizpůsobené skripty implementace.

-w ***serviceDirectory***
serviceDirectory je adresář obsahující webovou službu.

-x ***transakcionalita***
transactnost určuje typ transakčního řízení pro modul listener. *transakcionalitu* lze nastavit na jednu z následujících hodnot:

onePhase
IBM WebSphere MQ je použita jednofázová podpora. Pokud systém selže během zpracování, zpráva požadavku se znovu doručí do aplikace. Transakce WebSphere MQ zajistí, aby zprávy odezvy byly napsány přesně jednou.

twoPhase
Je použita dvoufázová podpora. Je-li služba zapsána správně, zpráva se doručí přesně jednou, koordinovanou s jinými prostředky, v rámci jediného potvrzeného provedení služby. Tato volba se vztahuje pouze na připojení vazeb serveru.

none
Žádná transakční podpora. Pokud dojde k selhání systému během zpracování, může dojít ke ztrátě zprávy požadavku i v případě, že je trvalá. Je možné, že služba byla nebo nemusela být provedena a že zprávy odezvy, sestavy nebo zprávy s deadutem mohou nebo nemusí být zapsány.

Obslužný program implementace kontroluje kompatibilitu příznaků -x a -a . Podrobnosti najdete v popisu příznaku -a .

Příklad .NET

```
amqwSOAPNETlistener
-u "jms:/queue?destination=myQ&connectionFactory=()
&targetService=myService&initialContextFactory=com.ibm.mq.jms.Nojndi"
-w C:/wmqsoap/demos
-n 20
```

Příklad jazyka Java

```
java com.ibm.mq.soap.transport.jms.SimpleJavaListener
-u "jms:/queue?destination=myQ&connectionFactory=()
&initialContextFactory=com.ibm.mq.jms.Nojndi"
-n 20
```

Přenos IBM WebSphere MQ pro odesílatele SOAP

Třídy odesilatele jsou k dispozici pro rámce Axis a .NET Framework 1 a .NET Framework 2. Odesílatel konstruuje požadavek SOAP a vkládá jej do fronty, a pak blokuje, dokud nepřečte odpověď z fronty odezev. Chování tříd můžete změnit předáním různých identifikátorů URI od klienta SOAP. Pro prostředí .NET Framework 3 použijte vlastní kanál produktu WebSphere MQ pro Microsoft Windows Communication Foundation (WCF).

Účel

Odesilatel SOAP produktu WebSphere MQ vloží žádost SOAP o vyvolání webové služby na frontu požadavků produktu WebSphere MQ . Odesilatel nastaví pole záhlaví v záhlaví **MQRFH2** podle voleb uvedených v identifikátoru URI nebo podle výchozího nastavení.

Potřebujete-li změnit chování odesílatele nad rámec toho, co je možné pomocí voleb identifikátoru URI, zapište si svého vlastního odesílatele. Odesílatel může pracovat s přenosem IBM WebSphere MQ pro moduly listener protokolu SOAP nebo s dalšími prostředími SOAP. Odesílatel musí zkonstruovat zprávy SOAP ve formátu definovaném v produktu WebSphere MQ. Formát je podporován modulem listener protokolu SOAP produktu IBM WebSphere MQ a také moduly listener SOAP poskytované produktem WebSphere Application Server a CICS. Odesílatel musí dodržovat pravidla pro žadatele IBM WebSphere MQ . Modul listener SOAP produktu IBM WebSphere MQ vrací zprávy odpovědí a sestav. Podrobnosti, jak nastavit volby sestavy v produktu **MQMD**, naleznete v příručce “Nastavení protokolu SOAP MQMD” na stránce 935 . Volby sestav řídí zprávy sestavy vrácené modulem listener protokolu SOAP produktu WebSphere MQ .

Popis

The WebSphere MQ SOAP Java sender is registered with the Axis host environment for the `jms:` URI prefix. Odesílatel je implementován do třídy `com.ibm.mq.soap.transport.jms.WMQSender`, která je odvozena od `org.apache.axis.handlers.BasicHandler`. Pokud prostředí hostitele Axis zjistí předponu identifikátoru URI `jms:` , vyvolá třídu `com.ibm.mq.soap.transport.jms.WMQSender` . Bloky třídy po umístění zprávy, dokud nečtou odpověď z fronty odpovědí. Není-li v intervalu časového limitu přijata žádná odezva, vyvolá odesilatel výjimku. Je-li odezva přijata v rámci intervalu časového limitu, vrátí se zpráva odpovědi klientovi pomocí rámce Axis. Vaše aplikace klienta musí být schopna zpracovat tyto zprávy s odpovědí.

Pro služby Microsoft .NET Framework 1 a .NET Framework 2 je odesilatel SOAP WebSphere MQ implementován do třídy `IBM.WMQSOAP.MQWebRequest`, která je odvozena od produktů `System.Net.WebRequest` a `System.Net.IwebRequestCreate`. Pokud prostředí .NET Framework 1 nebo .NET Framework 2 zjistí předponu URI `jms:` , vyvolá třídu `IBM.WMQSOAP.MQWebRequest` . Odesilatel vytvoří objekt `MQWebResponse` , který přečte zprávu odpovědi z fronty odpovědí a vrátí ji klientovi.

`com.ibm.mq.soap.transport.jms.WMQSender` je konečná třída a `IBM.WMQSOAP.MQWebRequest` je uzavřena. Jejich chování nelze upravit vytvořením podtříd.

Parametry

Nastavte identifikátor URI pro řízení chování odesilatele SOAP produktu IBM WebSphere MQ v klientovi SOAP webové služby. Obslužný program implementace vytváří stuby klienta webové služby obsahující volby identifikátoru URI dodávané do obslužného programu implementace.

Použít definiční tabulku kanálu s přenosem SOAP produktu WebSphere MQ SOAP pro odesílatele SOAP

Definice kanálu připojení klienta představuje alternativu k nastavení vlastností připojení v atributu ConnectionFactory identifikátoru URI webové služby. Vlastnosti připojení jsou parametry clientChannel, clientConnectiona SSL .

Popis

Vytvoření tabulky popisu kanálu klienta definováním připojení klienta. I když se klient webových služeb připojuje k různým správcům front, vytvořte všechna připojení v tabulce připojení na jediném správci front. Výchozí název a umístění tabulky připojení je *queue manager directory*/@ipcc/AMQCLCHL.TAB.

Předejte umístění tabulky připojení klientovi prostředí Java nastavením systémové vlastnosti com.ibm.mq.soap.transport.jms.mqchlurl .

Pomocí nastavení proměnných prostředí MQCHLLIB a MQCHLTAB předejte umístění tabulky připojení k klientovi .NET.

V atributu ConnectionFactory identifikátoru URI webové služby můžete zadat jak parametry kanálu připojení kanálu, tak parametry připojení kanálu. Hodnoty nastavené v ConnectionFactory mají přednost před hodnotami v tabulce definic kanálů.

Použití tabulky definic kanálů v prostředí Java

```
java -Dcom.ibm.msg.client.config.location=file:/C:/mydir/myjms.config MyAppClass
```

Obrázek 22. Spuštění klienta Java pomocí konfiguračního souboru

```
com.ibm.mq.soap.transport.jms.mqchlurl=file:/C:/ibm/wmq/qmgrs/QM1/@ipcc/AMQCLCHL.TAB
```

Obrázek 23. myjms.config

Transakce

Použijte volbu -x při spouštění modulu listener, abyste spustili webovou službu transakčně. Chcete-li vybrat integritu zpráv, nastavte volbu persistence v identifikátoru URI služby.

Webové služby.

Použijte volbu -x při spouštění modulu listener, abyste spustili webovou službu transakčně.
V prostředí .NET Framework 1 a 2 používá modul listener protokolu SOAP modul Microsoft Transaction Coordinator (MTS). Na ose Axis 1.4používá modul listener protokolu SOAP koordinované transakce správce front.

Klienti webové služby

odesilatelé SOAP nejsou transakčními transakcemi.

Vazby WebSphere MQ

Typ vazby pro odesilatele SOAP můžete nastavit. Může se připojit jako serverová aplikace WebSphere MQ nebo jako klientská aplikace. Odesilatele SOAP můžete také svázat jako klienta XA na platformě .NET.

Trvalost zpráv

Vyberte úroveň perzistence nastavením volby `Perzistence` v identifikátoru URI.

Transakce webové služby

Transakce webové služby můžete použít, protože odesilatel SOAP není transakční. Pokud zapíšete vlastního odesilatele protokolu SOAP a hodláte používat transakce webových služeb, nevytvářejte nevytvářený odesilatel protokolu SOAP. Ve stejné transakci nemůžete odeslat zprávu požadavku a přijmout zprávu odpovědi. Odeslání a přijetí nesmí být koordinováno transakcí webové služby.

Syntaxe identifikátoru URI a parametry pro implementaci webové služby

Syntaxe a parametry implementace webové služby IBM WebSphere MQ jsou definovány v identifikátoru URI. Obslužný program implementace generuje výchozí identifikátor URI založený na názvu webové služby. Výchozí hodnoty můžete přepsat definováním vlastního identifikátoru URI jako parametru pro obslužný program implementace. Obslužný program implementace obsahuje identifikátor URI v generovaných stubech klienta webové služby.

Účel

Webová služba je uvedena pomocí identifikátoru URI (Universal Resource Identifier). Syntaktický diagram uvádí identifikátor URI, který je podporován v transportu produktu IBM WebSphere MQ pro protokol SOAP. Identifikátor URI řídí parametry SOAP specifické pro produkt IBM WebSphere MQa volby použité pro přístup k cílovým službám. Identifikátor URI je kompatibilní s webovými službami hostovanými platformou .NET, Apache Axis 1, WebSphere Application Server, CICS.

Popis

Identifikátor URI je začleněn do tříd klienta webových služeb generovaných obslužným programem implementace. Klient předá identifikátor URI odesilateli protokolu SOAP produktu IBM WebSphere MQ do zprávy produktu IBM WebSphere MQ . Identifikátor URI řídí zpracování prováděné odesilatelem SOAP IBM WebSphere MQ a modulem listener SOAP produktu IBM WebSphere MQ .

Syntax

The URI syntax is as follows:

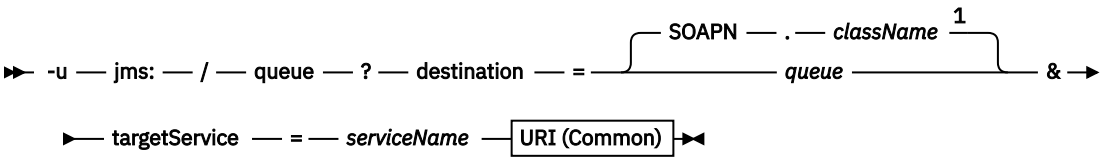
```
jms:/queue?name=value&name=value...
```

where `name` is a parameter name and *value* is an appropriate value, and the `name=`*value* element can be repeated any number of times with the second and subsequent occurrences being preceded by an ampersand (&).

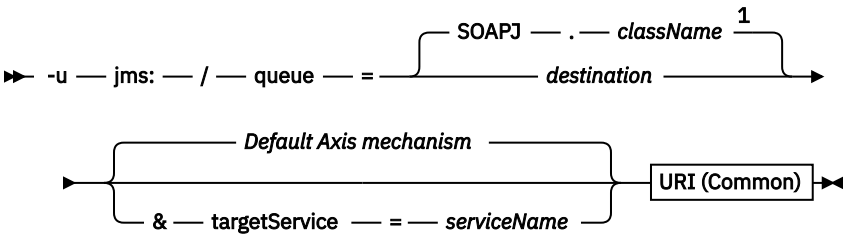
Parameter names are case-sensitive, as are names of IBM WebSphere MQ objects. If any parameter is specified more than once, the final occurrence of the parameter takes effect. Client applications can override a generated parameter by appending another copy of the parameter to the URI. If any additional unrecognized parameters are included, they are ignored.

If you store a URI in an XML string, you must represent the ampersand character as `&`. Similarly, if a URI is coded in a script, take care to escape characters such as & that would otherwise be interpreted by the shell.

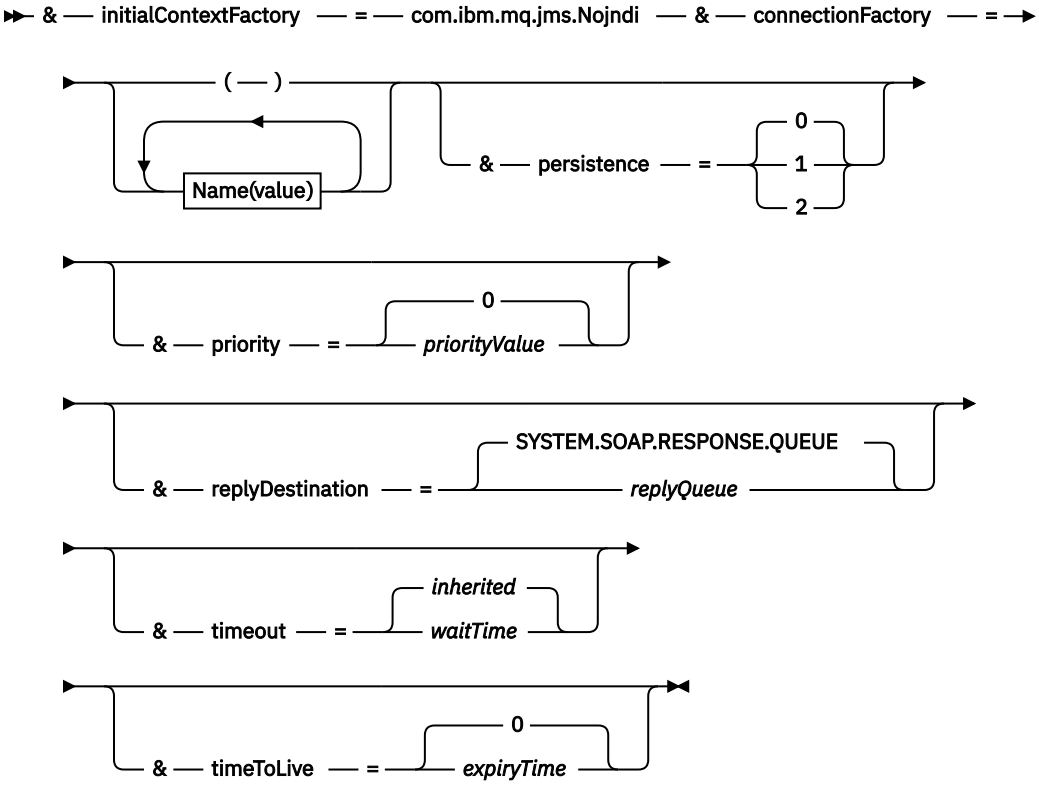
Syntax diagram
URI (.NET service)



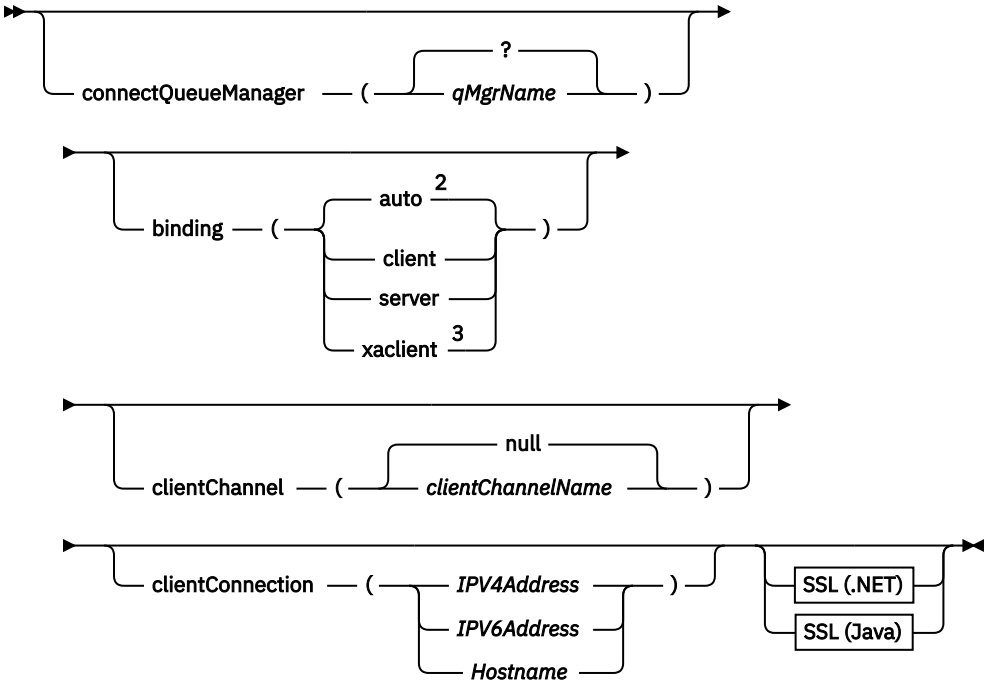
URI (Java service)



URI (Common)



***Name*(*value*)**



Poznámky:

[1] The queue manager transforms *className* to a queue name following the steps described in “Transformace místa určení do názvu fronty” na stránce 960

[2] `client` is the default if other options appropriate for a client are specified; for example `clientConnection`.

[3] `xaclient` applies to .NET only

Transformace místa určení do názvu fronty

1. *className* má předponu `SOAPJ.` pro služby Java nebo služby `SOAPN.` pro služby .NET.
2. Přípona souboru se odebere z úplného názvu cesty uvedeného v parametru *className* .
3. Výsledný řetězec je zkrácen na maximálně 48 znaků.
4. Znaky oddělovače adresářů se nahradí znaky tečky.
5. Vložené mezery jsou nahrazeny podtržítky.
6. Dvojtečka za písmenem s předponou jednotky je nahrazena tečkou pro službu .NET.

Poznámka: V některých prostředích nemusí být název fronty generovaný obslužným programem implementace jedinečný. Obslužný program implementace provádí kontrolu, zda má být vytvořena fronta. Můžete se rozhodnout přepsat obslužný program implementace změnou hierarchie adresáře implementace nebo úpravou dodaného procesu implementace.

Povinné parametry URI

`destination=`*fronta*

fronta je název cíle požadavku. Může se jednat o frontu nebo alias fronty. Je-li alias fronty alias, může se alias interpretovat jako téma.

- Je-li argument -u vynechán, je *fronta* generován z *název_třídy* pomocí kroků popsaných v “Transformace místa určení do názvu fronty” na stránce 960.
- Je-li zadán parametr-u , je povinný parametr *fronta* a musí se jednat o první parametr identifikátoru URI za počátečním parametrem `jms:/queue?` Řetězec. Zadejte buď název fronty produktu IBM WebSphere MQ , nebo název fronty a název správce front připojené pomocí symbolu @, například `SOAPN.trandemos@WMQSOAP.DEMO.QM`.

- Obslužný program implementace kontroluje, zda je název fronty, generovaný nebo poskytnutý, shodný s názvem existující fronty. provedená akce je popsána v tématu Tabulka 587 na stránce 961.

Tabulka 587. Ověření fronty

Skript modulu listener existuje?	**Skript modulu listener existuje v adresáři `./generated/server`.**		**Skript modulu listener neexistuje v adresáři `./generated/server`.**
Fronta ve skriptu modulu listener odpovídá *frontě*?	***queue* neodpovídá frontě požadavků použité ve skriptu modulu listener**	***Fronta* odpovídá frontě požadavků použité ve skriptu modulu listener**	
***fronta* existuje**	– *Implementace se ukončí s chybou.* – Služba již byla implementována v produktu `./generated/server`, ale používá jinou frontu.	– **Implementace pokračuje normálně.** – Služba již byla implementována v produktu `./generated/server`.	– *Implementace se ukončí s chybou.* – Spouštěcí skript modulu listener nebyl nalezen v produktu `./generated/server`, ale *fronta* je používána jinou službou nebo aplikací.
***fronta* neexistuje**		– **Implementace pokračuje s varováním.** – Předchozí implementace by mohla být neúspěšná, protože spuštění je platné, ale *fronta* chybí.	– **Implementace pokračuje normálně.** – Z tohoto adresáře nebyla implementována žádná služba.

`&connectionFactory=`*Název (hodnota)*

Název je jeden z následujících parametrů:

- connectQueueManager (qMgrName)
- binding (bindingType)
- clientChannel(kanál)
- clientConnection(připojení)
- "Požadované parametry SSL (Java)" na stránce 947

Popis hodnot těchto parametrů viz "Parametry továrny připojení" na stránce 963 .

`&targetService=`*serviceName*

[8]V prostředí .NET je názvem *serviceName* název služby .NET umístěný v adresáři implementace, například: `targetService=myService.asmx`. V prostředí .NET umožňuje parametr targetService , aby jeden modul listener protokolu SOAP produktu WebSphere MQ mohl zpracovávat požadavky na více služeb. Tyto služby musí být implementovány ze stejného adresáře.

[8] Pouze služba .NET

Volitelné parametry identifikátoru URI

&initialContextFactory=*contextFactory*
Hodnota *contextFactory* je povinná a musí být nastavena na `com.ibm.mq.jms.Nojndi`. Ujistěte se, že `Nojndi.jar` je v cestě ke třídě pro klienta webových služeb produktu WebSphere Application Server. Produkt `Nojndi.jar` vrací objekty produktu Java založené na obsahu parametrů `connectionFactory` a `destination` , a nikoli podle odkazu na adresář.

&targetService=*serviceName*
[9]Na ose Axis je *serviceName* úplný název služby produktu Java , například: `targetService=javaDemos.service.StockQuoteAxis`. Není-li parametr targetService zadán, bude služba načtena s použitím výchozího mechanizmu Axis.

&persistence=*messagePersistence*
messagePersistence má jednu z následujících hodnot:

0
Perzistence se dědí z definice fronty.

1
Zpráva je nestálá.

2
Zpráva je trvalá.

&priority=*priorityValue*
Hodnota *priorityValue* je v rozsahu 0 až 9. 0 je nízká priorita. Výchozí hodnota je specifická pro prostředí, která v případě IBM WebSphere MQ je 0.

&replyDestination=*replyToFronta*
Fronta na straně klienta, která má být použita pro zprávu odpovědi. Výchozí fronta odpovědí je `SYSTEM.SOAP.RESPONSE.QUEUE`.

- Spuštěním skriptu `setupWMQSOAP` vytvořte výchozí objekty SOAP produktu WebSphere MQ .
- Zadejte frontu modelu pro frontu *replyToQueue* a vytvořte dočasnou nebo trvalou dynamickou frontu odpovědí. Pro dočasné a trvalé dynamické fronty odpovědí je vytvořena samostatná instance dynamické fronty pro každý požadavek. Pokud dojde k odstranění některé z následujících událostí, dojde k odstranění této fronty:
  - Odezva je doručena a zpracována.
  - Požadavek vyprší.
  - Žádající program je ukončen.

  Chcete-li dosáhnout nejlepšího výkonu, použijte raději dočasné dynamické fronty než trvalé dynamické fronty. Neposílejte trvalou zprávu požadavku na identifikátor URI s dočasnou dynamickou frontou. SOAP modulu listener produktu IBM WebSphere MQ se nezdařilo zpracovat zprávu a chybu nageneruje. Vypršel časový limit klienta při čekání na odpověď.
- Skript `setupWMQSOAP` vytvoří výchozí trvalou dynamickou modelovou frontu s názvem `SYSTEM.SOAP.MODEL.RESPONSE.QUEUE`.

&timeout=*waitTime*
Doba (v milisekundách), po kterou klient čeká na zprávu odpovědi. Hodnota *waitTime* přepíše hodnoty nastavené infrastrukturou nebo klientskou aplikací. Není-li uvedena, hodnota aplikace, je-li uvedena, nebo výchozí nastavení infrastruktury je zděděné.

Poznámka: Mezi časovým limitem a timeToLivenení vynucena žádná relace.

&timeToLive=*expiryTime*
expiryTime je doba, uvedená v milisekundách, před vypršením platnosti zprávy. Výchozí hodnota je nula, která označuje neomezenou životnost.

Poznámka: Mezi časovým limitem a timeToLive nejsou vynuceny žádné vztahy.

[9] Pouze služba Java

Parametry továrny připojení

connectQueueManager(*qMgrName*)
qMgrName uvádí správce front, ke kterému se klient připojuje. Výchozí hodnota je prázdná.

binding(*bindingType*)
bindingType uvádí, jak je klient připojen k serveru *qMgrName*. Výchozí hodnota je auto. *bindingType* má následující hodnoty:

auto
Odesílatel zkouší následující typy připojení, v pořadí:

1. Jsou-li zadány jiné volby vhodné pro připojení klienta, odesílatel používá vazbu klienta. Další volby jsou clientConnection nebo clientChannel.
2. Použijte připojení k serveru.
3. Použijte připojení klienta.

Použijte `binding(auto)` v *URI* , pokud v klientovi SOAP neexistuje žádný lokální správce front. Připojení klienta je sestaveno pro klienta SOAP.

client
Chcete-li sestavit konfiguraci klienta pro odesílatele SOAP, použijte `binding(klient)` v *URI* .

server
Chcete-li sestavit konfiguraci serveru pro odesílatele SOAP, použijte `binding(server)` v *URI* . Pokud má připojení parametry typu klienta, připojení selže a odesilatel SOAP IBM WebSphere MQ zobrazí chybovou zprávu. Parametry typu klienta jsou clientConnection, clientChannel, nebo parametry SSL.

xaclient
xaclient lze použít pouze pro prostředí .NET, nikoli pro klienty Java. Použijte připojení typu XA-klient.

clientChannel(*kanál*)

Klient SOAP používá *kanál* k vytvoření připojení klienta IBM WebSphere MQ . *kanál* se musí shodovat s názvem kanálu připojení serveru, pokud na serveru není povolena automatická definice kanálu. Parametr clientChannel je vyžadovaný parametr, pokud jste neposkytli tabulku CCDT (Client Connection Definition table).

Zadejte tabulku CCDT v prostředí Java pomocí nastavení `com.ibm.mq.soap.transport.jms.mqchlurl`. V sadě .NET nastavte proměnné prostředí `MQCHLLIB` a `MQCHLTAB` , viz “Použít definiční tabulku kanálu s přenosem SOAP produktu WebSphere MQ SOAP pro odesílatele SOAP” na stránce 957.

clientConnection(*připojení*)
Klient SOAP používá *připojení* k vytvoření připojení klienta IBM WebSphere MQ . Výchozí název hostitele je localhosta výchozí port je 1414. Je-li *připojení* adresa TCP/IP, bude mít jeden ze tří formátů a může mít příponu s číslem portu.

Klienti JMS mohou buď použít formát: `hostname:port` , nebo 'escape' hranaté závorky používající formát `%X` , kde X je hexadecimální hodnota, která představuje znak závorky v kódové stránce identifikátoru URI. Například, v ASCII, `%28` a `%29` pro `(` a `)` .

Klienti .Net mohou používat závorky explicitně: `hostname(port)` nebo použít formát 'escaped'.

Adresa IPv4
Například `192.0.2.0`.

Adresa IPv6
Například `2001:DB8:0:0:0:0:0:0`.

Název hostitele
Například `www.example.com%281687%29`, `www.example.com:1687` nebo `www.example.com(1687)`.

SSL ***platforma***
Viz“Požadované parametry SSL (Java)” na stránce 947

Ukázkové identifikátory URI

Poznámka:

1. & v identifikátoru URI je kódováno jako &
2. Všechny výše uvedené parametry se vztahují na klienty.
3. Pouze **destination**, **connectionFactory** a **initialContextFactory** jsou použitelné pro službu WCF.

```
jms:/queue?
destination=myQ&amp;connectionFactory=()&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Obrázek 24. Identifikátor URI pro službu Axis, který poskytuje pouze požadované parametry

```
jms:/queue?destination=myQ&amp;connectionFactory=()&amp;targetService=MyService.asmx
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Obrázek 25. Identifikátor URI pro službu .NET, poskytující pouze požadované parametry

```
jms:/queue?destination=myQ@myRQM&amp;connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Obrázek 26. Identifikátor URI pro službu Axis, který dodává některé volitelné parametry connectionFactory .

```
jms:/queue?destination=myQ@myRQM&amp;connectionFactory=connectQueueManager(myconnQM)
binding(client)clientChannel(myChannel)clientConnection(myConnection)
sslPeerName(CN=MQ Test 1,O=IBM,S=Hampshire,C=GB)
&amp;initialContextFactory=com.ibm.mq.jms.Nojndi
```

Obrázek 27. Identifikátor URI pro službu Axis dodávající volbu sslPeerName parametru connectionFactory

Mechanizmus `Nojndi`

Mechanismus `Nojndi` umožňuje programům JMS, které používají rozhraní JNDI, používat stejný identifikátor URI jako programy produktu WebSphere MQ , které nepoužívají rozhraní JNDI.

K vyvolání webových služeb na serveru WebSphere Application Server můžete použít přenos WebSphere MQ pro SOAP. Produkt WebSphere Application Server SOAP prostřednictvím rozhraní JMS vyhledává prostředky platformy JMS s použitím rozhraní JNDI. Klient webové služby může být spuštěn na platformě .NET nebo pomocí produktu Axis 1.4k vyvolání webové služby a nepoužívá se k rozhraní JNDI. Chcete-li použít stejnou adresu URL pro klienta a server, musí poskytovat stejné informace o tom, zda prostředí používá rozhraní JNDI, či nikoli.

Identifikátor URI předaný transportu produktu WebSphere MQ pro protokol SOAP pomocí klienta webové služby obsahuje specifický správce front WebSphere MQ a názvy front. Tyto názvy jsou analyzovány a používány přímo podporou protokolu SOAP produktu WebSphere MQ .

Mechanismus `Nojndi` nasměruje initialContextFactory , který používá program JMS, do produktu `com.ibm.mq.jms.Nojndi`. Třída `com.ibm.mq.jms.Nojndi` je implementace rozhraní JNDI, které vrací connectionFactory a destination z adresy URL jako objekty ConnectionFactory a Queue Java. Je-li implementací platformy JMS WebSphere MQ, MQConnectionFactory a MQQueue dědí z tříd ConnectionFactory a Queue .

Pomocí mechanizmu `Nojndi` můžete poskytovat stejné informace o připojení k serveru WebSphere Application Server a .NET pomocí stejné adresy URL.

W3C SOAP over JMS URI pro klienta WebSphere MQ Axis 2

Definujte protokol SOAP prostřednictvím rozhraní W3C SOAP over JMS pro volání webové služby z klienta Axis 2 pomocí produktu WebSphere JMS MQ JMS jako přenosu SOAP. Webovou službu musí poskytovat server, který podporuje WebSphere MQ JMS a doporučené doporučení W3C SOAP přes JMS pro vazbu SOAP/JMS.

Popis

Doporučení kandidáta W3C definuje protokol SOAP nad vazbou služby JMS; Protokol SOAP nad Java Message Service 1.0. Také užitečné pro jeho příklady je Schéma identifikátoru URI pro službu Java(tm) Message Service 1.0[10].

Použijte syntaktický diagram k vytvoření protokolu W3C SOAP přes identifikátory URI JMS, které jsou syntakticky správné, a jsou přijímány klientem WebSphere MQ Axis 2. Omezuje se na definování identifikátoru URI, který přijímá klient WebSphere MQ Axis 2. Jedná se o podmnožinu doporučení W3C ve dvou ohledech:

1. Volba jms-variant topic není podporována a nesmí být zadána v identifikátoru URI předávaném klientovi WebSphere MQ Axis 2.
2. Následující vlastnosti jsou z diagramu syntaxe vynechány, protože se jedná o vlastnosti platformy JMS a nejsou součástí identifikátoru URI.
   a. bindingVersion
   b. contentType
   c. soapAction
   d. requestURI
   e. isFault

Vlastnosti JMS jsou nastaveny klientem Axis 2 nebo serverem.

Diagram rozšiřuje doporučení W3C definováním vlastního parametru connectionFactory. connectionFactory se používá jako alternativa k rozhraní JNDI k určení způsobu, jakým se klient Axis 2 připojuje ke správci front pomocí fronty.

Klient WebSphere MQ Axis 2 přijímá vlastnosti pouze jako část identifikátoru URI předaného klientovi klientské aplikace nebo proměnné prostředí. Klient WebSphere MQ Axis 2 nemá žádnou schopnost zpracovat dokument WSDL. Aplikace klienta nebo vývojový nástroj může zpracovat kód WSDL a vytvořit identifikátor URI pro předání na klienta Axis 2. Klientská aplikace WebSphere MQ Axis 2 nemůže přímo nastavit vlastnosti zprávy JMS.

Syntax

In accordance with the W3C recommendation, all the parameters can be obtained from environment variables. The environment variable names are formed by prefacing the parameter name with `soapjms_`. The syntax is: `soapjms_`*parameterName*; for example,

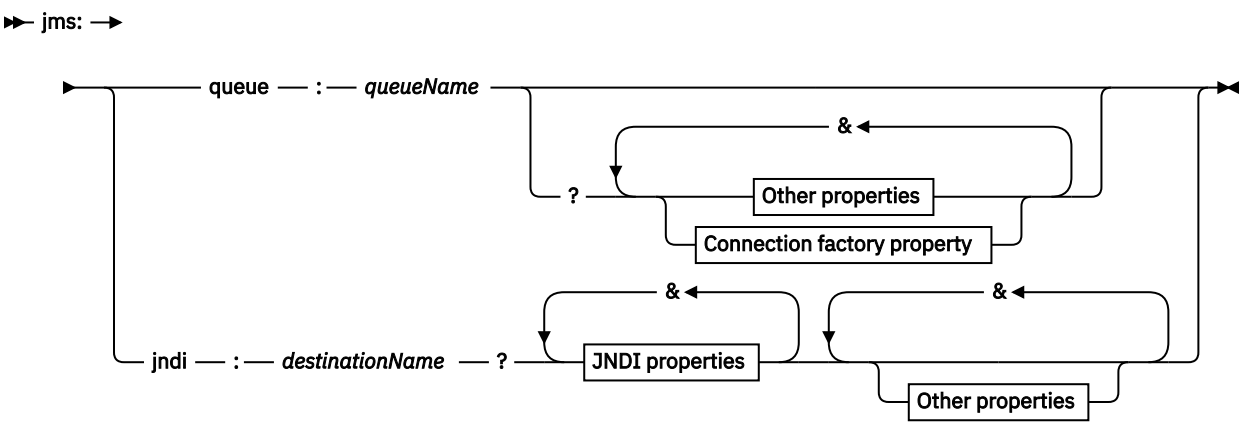
```
set soapjms_targetServer=com.example.org.stockquote
```

If a parameter is set using an environment variable it overrides the value set in the URI.

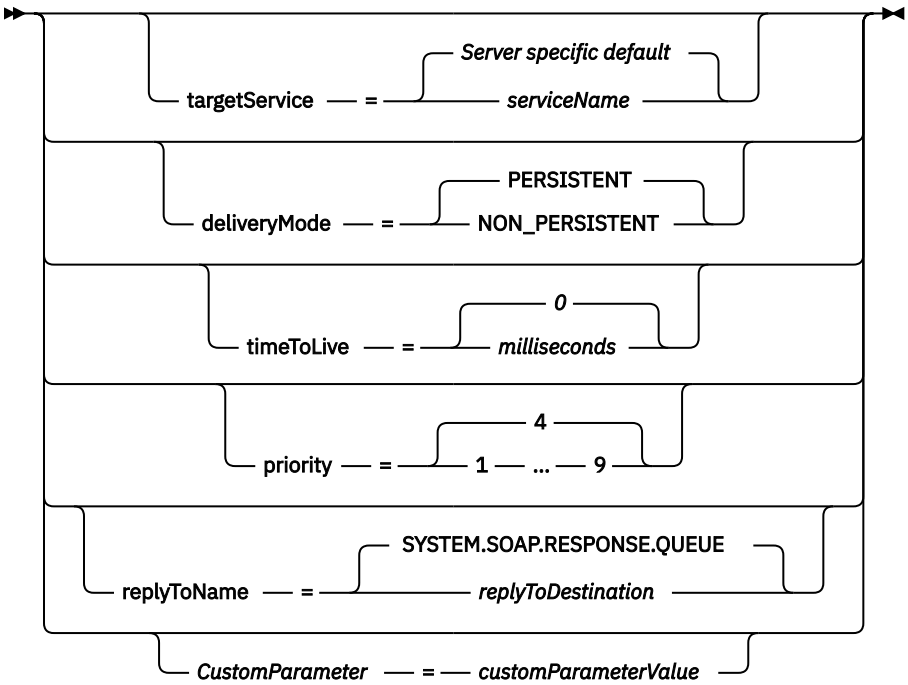
In accordance with the W3C recommendation, all the parameters can be repeated. The last instance of a parameter is used, unless overridden by an environment variable.

[10] Vyhledejte *Schéma identifikátoru URI pro rozhraní JMS* v odkazech specifikace W3C pro nejnovější koncept.

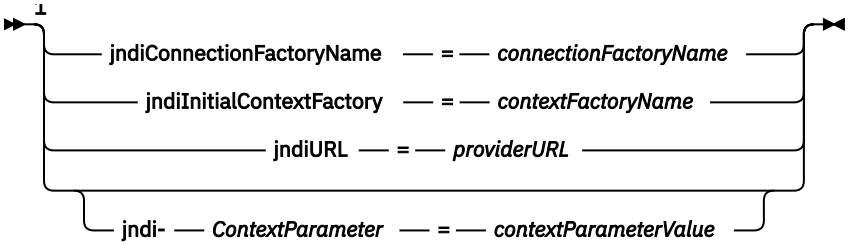
jms-uri



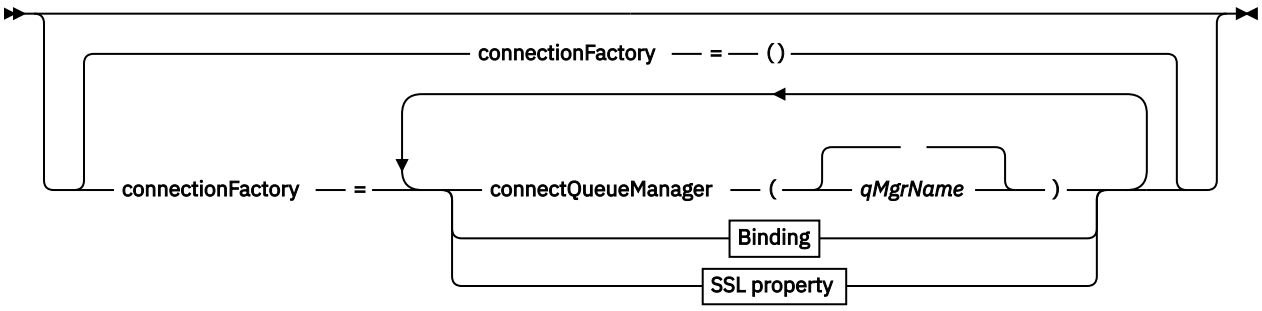
Other properties



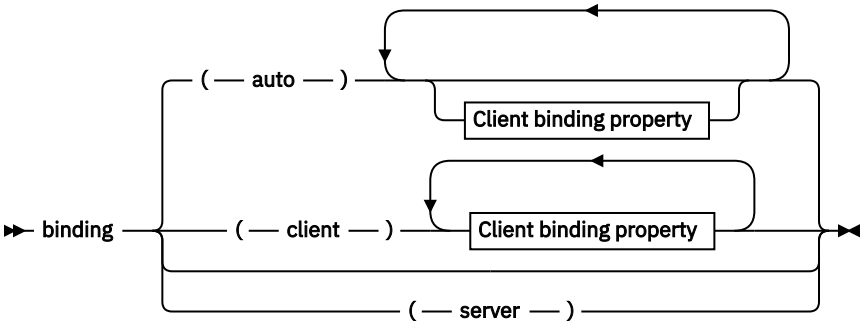
JNDI properties



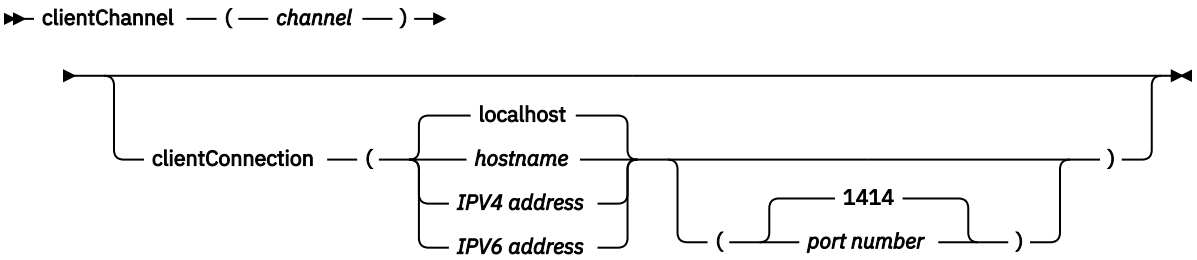
Connection factory property



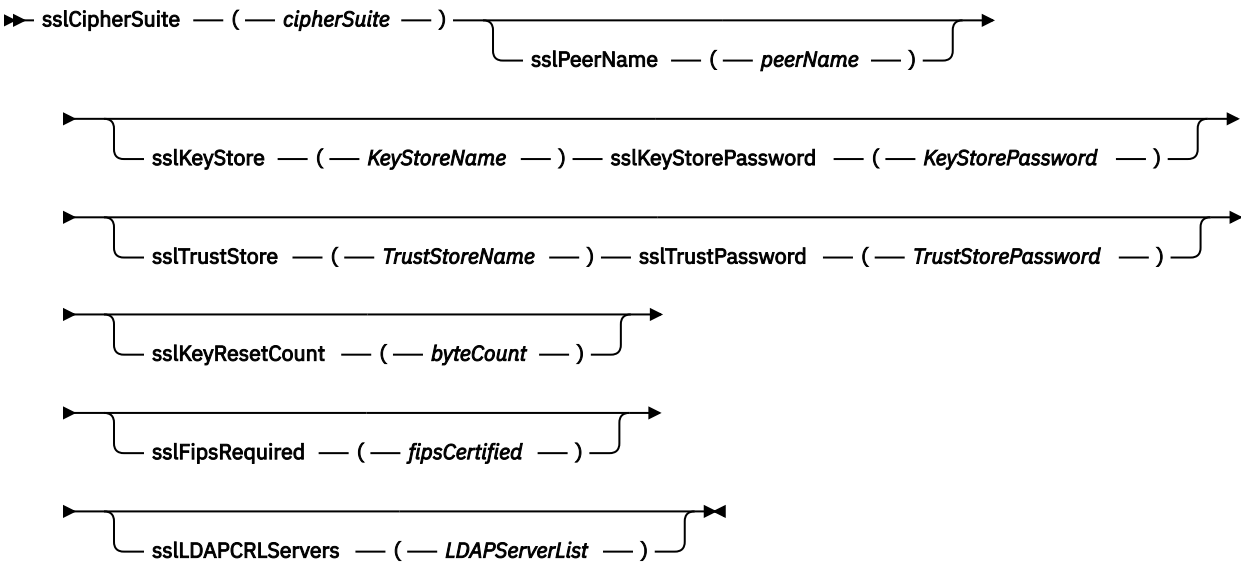
Binding



Client binding property



SSL property



Poznámky:

[1] **jndiConnectionFactoryName**, **jndiConnectionFactoryName** and **jndiURL** are all required parameters. **jndi-***ContextParameter* is optional.

Parametry

connectionFactory=*connectionFactoryParameterList*

connectionFactoryParameterList jsou parametry, které kvalifikují, jak se klient Axis 2 připojuje ke správci front, když je cílová varianta fronta.

Hodnota connectionFactory nesmí být zadána s cílovou variantou jndi .

Parametry se nepředávají serveru v identifikátoru URI požadavku.

Je-li parametr connectionFactory vynechán, musí fronta náležet k výchozímu správci front spuštěnému na stejném serveru jako klient Axis 2.

connectionFactoryParameterList:

binding(*bindingType*)

bindingType uvádí, jak je klient připojen k serveru *qMgrName*. Výchozí hodnota je auto. *bindingType* má následující hodnoty:

auto

Odesílatel zkouší následující typy připojení, v pořadí:

1. Jsou-li zadány jiné volby vhodné pro připojení klienta, odesílatel používá vazbu klienta. Další volby jsou clientConnection nebo clientChannel.
2. Použijte připojení k serveru.
3. Použijte připojení klienta.

Použijte `binding(auto)` v *URI* , pokud v klientovi SOAP neexistuje žádný lokální správce front. Připojení klienta je sestaveno pro klienta SOAP.

client

Chcete-li sestavit konfiguraci klienta pro odesílatele SOAP, použijte `binding(klient)` v *URI* .

server

Chcete-li sestavit konfiguraci serveru pro odesílatele SOAP, použijte `binding(server)` v *URI* . Pokud má připojení parametry typu klienta, připojení selže a odesilatel SOAP IBM WebSphere MQ zobrazí chybovou zprávu. Parametry typu klienta jsou clientConnection, clientChannel, nebo parametry SSL.

xaclient

xaclient lze použít pouze pro prostředí .NET, nikoli pro klienty Java. Použijte připojení typu XA-klient.

clientChannel(*kanál*)

Klient SOAP používá *kanál* k vytvoření připojení klienta IBM WebSphere MQ . *kanál* se musí shodovat s názvem kanálu připojení serveru, pokud na serveru není povolena automatická definice kanálu. Parametr clientChannel je vyžadovaný parametr, pokud jste neposkytli tabulku CCDT (Client Connection Definition table).

Zadejte tabulku CCDT v prostředí Java pomocí nastavení `com.ibm.mq.soap.transport.jms.mqchlurl`. V sadě .NET nastavte proměnné prostředí `MQCHLLIB` a `MQCHLTAB` , viz “Použít definiční tabulku kanálu s přenosem SOAP produktu WebSphere MQ SOAP pro odesílatele SOAP” na stránce 957.

clientConnection(*připojení*)

Klient SOAP používá *připojení* k vytvoření připojení klienta IBM WebSphere MQ . Výchozí název hostitele je localhosta výchozí port je 1414. Je-li *připojení* adresa TCP/IP, bude mít jeden ze tří formátů a může mít příponu s číslem portu.

Klienti JMS mohou buď použít formát: `hostname:port` , nebo 'escape' hranaté závorky používající formát %X , kde X je hexadecimální hodnota, která představuje znak závorky v kódové stránce identifikátoru URI. Například, v ASCII, %28 a %29 pro (a) .

Klienti .Net mohou používat závorky explicitně: `hostname(port)` nebo použít formát 'escaped'.

Adresa IPv4

Například `192.0.2.0`.

Adresa IPv6
Například `2001:DB8:0:0:0:0:0:0`.

Název hostitele
Například `www.example.com%281687%29`, `www.example.com:1687` nebo `www.example.com(1687)`.

sslCipherSuite(*CipherSuite*)
Volba *CipherSuite* uvádí sslCipherSuite použitou na kanálu. Volba CipherSuite určená klientem musí odpovídat sadě CipherSuite určené pro kanál připojení k serveru.

sslFipsRequired(*fipsCertified*)
fipsCertified uvádí, zda *CipherSpec* nebo *CipherSuite* musí v kanálu IBM WebSphere MQ na kanálu používat šifrovací mechanizmus certifikovaný FIPS. Efekt nastavení *fipsCertified* je stejný jako nastavení pole `FipsRequired` struktury **MQSCO** na volání `MQCONNX` .

sslKeyStore(*KeyStoreName*)
KeyStoreName uvádí sslKeyStoreName použité na kanálu. Úložiště klíčů obsahuje soukromý klíč klienta použitého k ověření klienta na serveru. Úložiště klíčů je volitelné, pokud je připojení SSL konfigurováno tak, aby přijímaly anonymní připojení klienta.

sslKeyResetCount(*bytecount*)
Parametr *bytecount* určuje počet bajtů předávaných přes kanál SSL před opětovným vyjednáním tajného klíče zabezpečení SSL. Chcete-li zakázat opětovné vyjednávání klíčů SSL, vynechte toto pole nebo nastavte hodnotu nula. Nula je jediná hodnota podporovaná v některých prostředích, viz Opětovné dohadování tajného klíče ve třídách WebSphere MQ pro Javu. Efekt nastavení sslKeyResetCount je stejný jako nastavení pole `KeyResetCount` ve struktuře **MQSCO** na volání `MQCONNX` .

sslKeyStorePassword(*KeyStoreHeslo*)
KeyStoreHeslo uvádí sslKeyStorePassword použité na kanálu.

sslLDAPCRLServers(*LDAPServerList*)

Položka *LDAPServerList* určuje seznam serverů LDAP, které mají být použity pro kontrolu seznamu odvolaných certifikátů (CRL).

U připojení klienta s povoleným SSL je *LDAPServerList* seznam serverů LDAP, které se mají použít pro kontrolu seznamu odvolaných certifikátů (CRL). Certifikát poskytnutý správcem front je kontrolován na jednom z vypsaných serverů LDAP CRL; pokud je nalezen, připojení selže. Každý server LDAP se vyzkouší, dokud není ustanoveno připojení k jednomu z nich. Pokud se nelze připojit k žádnému ze serverů, certifikát se odmítne. Po úspěšném navázání spojení s jedním z nich je certifikát přijat nebo zamítnut v závislosti na seznamech CRL přítomných na daném serveru LDAP.

Je-li parametr *LDAPServerList* prázdný, nebude certifikát náležící ke správci front kontrolován na základě seznamu odvolaných certifikátů. Je-li zadaný seznam identifikátorů URI protokolu LDAP neplatný, zobrazí se chybová zpráva. Efekt nastavení tohoto pole je stejný, jako je zahrnutí záznamů `MQAIR` a přístupu k nim ze struktury **MQSCO** na `MQCONNX`.

sslPeerName(*peerName*)
peerName uvádí sslPeerName použité na kanálu.

sslTrustStore(*TrustStoreNázev*)
Hodnota *TrustStoreName* uvádí sslTrustStoreName použitou na kanálu. Úložiště údajů o důvěryhodnosti uchovává veřejný certifikát serveru nebo jeho klíčový řetězec k ověření serveru pro klienta. Úložiště údajů o důvěryhodnosti je volitelné, pokud se ke ověření serveru používá kořenový certifikát certifikační autority. V jazyce Java jsou certifikáty kořenových certifikátů uchovávány v úložišti certifikátů prostředí JRE, `cacerts`.

sslTrustStorePassword(*TrustStoreHeslo*)
Hodnota *TrustStorePassword* uvádí sslTrustStorePassword použité na kanálu.

CustomParameter=customParameterHodnota

CustomParameter je uživatelem definovaný název vlastního parametru a hodnota parametru *customParameterValue* je hodnota parametru.

Vlastní parametry, které nejsou používány klientem Axis 2, jsou odeslány klientem Axis 2 na server SOAP. Nahlédněte do dokumentace k serveru. connectionFactory je vlastní parametr, který je používán klientem Axis 2 a který není předán serveru.

Hodnota *CustomParameter* se nesmí shodovat s názvem existujícího parametru.

Pokud je parametr *CustomParameter* spuštěn s řetězcem `jndi-` , používá se při vyhledávání místa určení rozhraní JNDI, viz jndi-.

`deliveryMode=`*deliveryMode*

deliveryMode nastavuje perzistenci zpráv. Výchozí hodnota je `PERSISTENT`.

`jndi:`*destinationName*

destinationName je název cíle rozhraní JNDI, který je mapován na frontu JMS. Je-li uvedena varianta cíle jndi , musíte zadat *destinationName*.

`jndiConnectionFactoryName=`*connectionFactory*

Objekt *connectionFactoryName* nastavuje název rozhraní JNDI továrny připojení. Je-li cílová varianta jndi, musí být zadán název *connectionFactory* .

`jndiInitialContextFactory=`*contextFactory*

contextFactoryName nastavuje název rozhraní JNDI počáteční kontextové továrny. Je-li cílová varianta jndi, musí být poskytnut *contextFactoryName* . Viz Použití rozhraní JNDI pro načtení spravovaných objektů v aplikaci platformy JMS.

`jndiURL=`*providerURL*

jndiURL nastavuje adresu URL poskytovatele JNDI. Je-li cílová varianta jndi, musí být zadána hodnota *jndiURL* .

`jndi-`*ContextParameter=contextParameterHodnota*

jndi-*ContextParameter* je uživatelem definovaný název vlastního parametru, který se používá k předávání informací poskytovateli rozhraní JNDI. *contextParameterHodnota* je informace, která je předávána.

`priority=`*priorityValue*

priorityValue nastavuje prioritu zprávy JMS. `0` je nízká, 9 je vysoká. Standardní hodnota je 4.

`queue:`*queueName*

queueName je název fronty JMS, na které je požadavek SOAP umístěn. Je-li určena varianta fronty, musí být zadán název fronty. Pokud fronta nepatří do výchozího správce front na stejném serveru jako klient, nastavte parametr connectionFactory .

`replyToName=`*replyToCíl*

Parametr *replyToDestination* nastavuje název cílové fronty. Je-li cílová varianta jndi, název je název rozhraní JNDI, který musí být mapován na frontu. Je-li variantou fronta , názvem je fronta platformy JMS. Standardní hodnota je `SYSTEM.SOAP.RESPONSE.QUEUE`.

`targetService=`*serviceName*

Název používaný serverem SOAP ke spuštění cílové webové služby.

Na ose Axis je *serviceName* plně kvalifikovaný název služby Java, například: `targetService=www.example.org.StockQuote`. Není-li parametr targetService zadán, bude služba načtena s použitím výchozího mechanizmu Axis.

`timeToLive=`*milisekundy*

Nastavte *milisekundy* na dobu před vypršením platnosti zprávy. Výchozí hodnota `0`je, že zpráva nikdy nevyprší.

Příklady

```
jms:jndi:REQUESTQ
    ?jndiURL=file:/C:/JMSAdmin
    &jndiInitialContextFactory=com.sun.jndi.fscontext.RefFSContextFactory
    &jndiConnectionFactoryName=ConnectionFactory
    &replyToName=RESPONSEQ
    &deliveryMode(NON_PERSISTENT)
```

Obrázek 28. Chcete-li odeslat požadavek SOAP/JMS, použijte jms: jndi .

```
jms:queue:SOAPJ.demos
    ?connectionFactory=connectQueueManager(QM1)
     Bind(Client)
     ClientChannel(SOAPClient)
     ClientConnection(www.example.org(1418))
    &deliveryMode(NON_PERSISTENT)
```

Obrázek 29. Chcete-li odeslat požadavek SOAP/JMS, použijte jms: queue .

Podporované webové služby

Kód, který byl zapsán, aby se spustil jako webová služba, nemusí být upraven, aby používal přenos IBM WebSphere MQ pro SOAP. Je třeba implementovat služby jinak, aby se spouštěly s přenosem produktu IBM WebSphere MQ pro protokol SOAP, nikoli pomocí protokolu HTTP.

Popis

Přenos WebSphere MQ pro SOAP poskytuje modul listener SOAP ke spuštění služeb pro prostředí .NET Framework 1 a .NET 2 a pro Axis 1.4. Vlastní kanál produktu WebSphere MQ pro produkt Microsoft Windows Communication Foundation spouští služby pro prostředí .NET Framework 3. Server WebSphere Application Server a CICS poskytují podporu pro spouštění služeb v rámci přenosu WebSphere MQ pro SOAP. Vytvořte vlastní export pro použití produktu WebSphere Enterprise Service Bus nebo WebSphere Process Server.

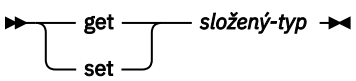
Modul listener protokolu SOAP produktu WebSphere MQ může zpracovávat transakce SOAP transakčně. Spusťte **amqwdeployWMQService** pomocí volby -x . Dvoufázová volba je podporována pouze pro moduly listener používající vazby serveru. Jiná prostředí mohou poskytovat transakční podporu pro přenos WebSphere MQ pro SOAP. Nahlédněte do jejich dokumentace.

Přenos WebSphere MQ pro protokol SOAP v současné době nepodporuje vznikající průmyslový standard protokolu SOAP prostřednictvím rozhraní JMS, který byl odeslán na úroveň W3C. Při hledání vlastnosti JMS BindingVersion je možné rozlišit zprávu SOAP/JMS zapisovaná do nového standardu. Přenos protokolu SOAP produktu WebSphere MQ nenastavujte vlastnost BindingVersion .

Osa 1.4

Třída Java může být obvykle použita bez úprav. Typy argumentů pro metody ve webové službě musí být podporovány jádrem Axis. Další podrobnosti naleznete v dokumentaci Axis. Pokud služba používá komplexní objekt jako argument nebo vrací jeden objekt, musí tento objekt vyhovovat specifikaci Java™ . Viz příklady v tématech Obrázek 32 na stránce 973, Obrázek 33 na stránce 973a Obrázek 34 na stránce 974:

1. Mějte konstruktor bez veřejného parametru.
2. Všechny komplexní typy objektu typu bean musí mít veřejné metody getter a setter ve tvaru:



Připravte službu pro implementaci pomocí obslužného programu **amqwdeployWMQService** . Služba je vyvolána modulem listener protokolu SOAP produktu WebSphere MQ , který používá ke spuštění služby produkt `axis.jar` .

Jediný dvoufázový správce transakcí podporovaný pro Axis 1.4 je WebSphere MQ.

Dodaný obslužný program implementace nepodporuje případ, kdy služba vrací objekt v jiném balíku pro samotnou službu. Chcete-li použít objekt vrácený v jiném balíku, zapište vlastní obslužný program implementace. Při použití volby -v můžete základní obslužný program implementace založit na dodané ukázce nebo zachytávat příkazy, které vytváří. Upravte příkazy tak, aby vytvářely přizpůsobený skript.

If the service uses classes that are external to the Axis infrastructure and the WebSphere MQ SOAP run time environment, you must set the correct CLASSPATH. Chcete-li změnit produkt CLASSPATH, změňte vygenerovaný skript, který spustí nebo definuje listenery tak, aby zahrnoval požadované služby, jedním z následujících způsobů:

- Po volání do produktu **amqwsetcp**ukončete program CLASSPATH přímo ve skriptu.
- Vytvořte skript specifický pro službu, chcete-li upravit CLASSPATH a vyvolat tento skript ve vygenerovaném skriptu po volání příkazu **amqwsetcp**.
- Vytvořte upravený proces implementace pro automatické přizpůsobení CLASSPATH ve vygenerovaném skriptu.

.NET Framework 1 a .NET Framework 2

Služba, která již byla připravována jako webová služba HTTP, není třeba upravovat pro použití jako webová služba produktu WebSphere MQ . Je třeba ji implementovat pomocí obslužného programu **amqwdeployWMQService** .

Jediný dvoufázový správce transakcí podporovaný pro .NET Framework 1 a .NET 2 je Microsoft Transaction Server (MTS).

Pokud kód služby nebyl připravován jako webová služba HTTP, je třeba ji převést na webovou službu. Deklarujte třídu jako webovou službu a identifikujte, jak jsou parametry jednotlivých metod formátovány. Musíte zkontrolovat, zda jsou jakékoli argumenty pro metody služby kompatibilní s prostředím. Obrázek 30 na stránce 973 a Obrázek 31 na stránce 973 zobrazit třídu .NET, která byla připravena jako webová služba. Provedená přidání jsou zobrazena tučným písmem.

Obrázek 30 na stránce 973 používá programovací model programování pro webovou službu .NET. V modelu-za modelem je zdroj pro službu oddělen od souboru `.asmx` . Soubor `.asmx` deklaruje název přidruženého zdrojového souboru s klíčovým slovem Codebehind . Produkt WebSphere MQ obsahuje ukázky vložených i kódových webových služeb .NET Web Services.

Před implementací pomocí obslužného programu implementace produktu **amqwdeployWMQService** musí být kompilován zdroj webových služeb .NET. Služba je kompilována do knihovny (`.dll`). Knihovna musí být umístěna v podadresáři `./bin` v adresáři implementace.

.NET Framework 3

Vytvořte vlastní kanál produktu WebSphere MQ pro produkt Microsoft Windows Communication Foundation (WCF) k vyvolání služeb implementovaných do prostředí .NET Framework 3. Popis postupu konfigurace WCF pro použití přenosu WebSphere MQ pro SOAP najdete v tématu Vlastní kanál produktuIBM WebSphere MQ pro produkt Microsoft Windows Communication Foundation (WCF) .

WebSphere Application Server

Webové služby hostované serverem WebSphere Application Server můžete vyvolat pomocí produktu WebSphere MQ Transport for SOAP; viz téma Použití protokolu SOAP prostřednictvím rozhraní JMS k transportu webových služeb (zamítnuto).

Chcete-li generovat klienta webových služeb, je třeba upravit kód WSDL generovaný implementací služby JMS na server WebSphere Application Server. Soubor WSDL vytvořený implementací do produktu WebSphere Application Server obsahuje identifikátor URI s odkazem JNDI na rozhraní JMS `InitialContextFactory`. Je třeba upravit odkaz JNDI na Nojndi a poskytnout atributy připojení, jak je popsáno v tématu “Syntaxe identifikátoru URI a parametry pro implementaci webové služby” na stránce 958.

CICS

Aplikace CICS můžete vyvolat pomocí přenosu WebSphere MQ Transport pro SOAP, viz téma Konfigurace systému CICS pro webové služby.

WebSphere Enterprise Service Bus a WebSphere Process Server for Multiplatforms

WebSphere ESB a WebSphere Process Server for Multiplatforms podporují protokol SOAP prostřednictvím rozhraní JMS s připravenou vazbou sestavení, pouze při použití výchozího poskytovatele systému zpráv WebSphere Application Server. Vytvořte vlastní vazbu pro službu JMS pro podporu přenosu WebSphere MQ pro SOAP. Viz Vázání dat služby JMS.

Příklad

```
<%@ WebService Language="C#" CodeBehind="Quote.asmx.cs" Class="Quote.QuoteDotNet" %>
```

Obrázek 30. Definice služby pro .NET Framework 2: `Quote.asmx`

```
<%@ WebService Language="C#" CodeBehind="Quote.asmx.cs" Class="Quote.QuoteDotNet" %>
using System;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;

namespace Quote {
    [WebService(Namespace = "http://www.example.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    public class QuoteDotNet : System.Web.Services.WebService {
        [WebMethod]
        public string getQuote(String symbol){
            return symbol.ToUpper();
        }
    }
}
```

Obrázek 31. Implementace služby pro .NET Framework 2: `Quote.asmx.cs`

```
package org.example.www;
public interface CustomerInfoInterface extends java.rmi.Remote {
    public org.example.www.CustomerRecord
           getCustomerName(org.example.www.CustomerRecord request)
           throws java.rmi.RemoteException, org.example.www.GetCustomerName_faultMsg;
}
```

Obrázek 32. Rozhraní služby JAX-RPC Java s použitím komplexního typu

```
package org.example.www;
public class CustomerInfoPortImpl implements org.example.www.CustomerInfoInterface{
    public org.example.www.CustomerRecord
           getCustomerName(org.example.www.CustomerRecord request)
           throws java.rmi.RemoteException, org.example.www.GetCustomerName_faultMsg {
        request.setName(request.getID().toString());
        return request;
    }
}
```

Obrázek 33. Implementace služby Java JAX-RPC s použitím komplexního typu

```
package org.example.www;
public class CustomerRecord  {
    private java.lang.String name;
    private java.lang.Integer ID;
    public CustomerRecord() {}
    public java.lang.String getName() {
        return name; }
    public void setName(java.lang.String name) {
        this.name = name; }
    public java.lang.Integer getID() {
        return ID; }
    public void setID(java.lang.Integer ID) {
        this.ID = ID; }
}
```

Obrázek 34. Implementace objektu typu bean služby Java JAX-RPC komplexního typu

Přenos IBM WebSphere MQ pro klienty webové služby SOAP

Existující protokol SOAP prostřednictvím klienta HTTP můžete znovu použít s přenosem IBM WebSphere MQ pro protokol SOAP. Chcete-li převést klienta na práci s přenosem produktu IBM WebSphere MQ pro protokol SOAP, musíte provést některé malé úpravy kódu a procesu sestavení.

Kódování

Klienti JAX-RPC musí být napsány v jazyce Java. Klienti .NET Framework 1 a 2 mohou být napsány v libovolném jazyce, který používá běhovou komponentu Common Language. Příklady kódu jsou poskytnuty v C# a Visual Basic.

Úroveň transakční podpory závisí na prostředí klienta a na vzoru interakce protokolu SOAP. Požadavek SOAP a odpověď SOAP nemohou být částí stejné atomické transakce.

Musíte volat `IBM.WMQSOAP.Register.Extension()` v klientovi .NET Framework 1, .NET Framework 2. V klientu webové služby JAX-RPC jazyka Java `com.ibm.mq.soap.Register.extension` registrujte odesilatele protokolu SOAP produktu WebSphere MQ . Metoda registruje přenos WebSphere MQ pro odesílatele SOAP jako obslužnou rutinu pro zprávy SOAP s použitím protokolu `jms:` .

Chcete-li vytvořit klienta .NET Framework 3, vygenerujte proxy klienta Windows Communication Foundation pomocí nástroje **svcutil** ; viz Generování proxy klienta WCF a konfiguračních souborů aplikace pomocí nástroje svcutil s metadaty ze spuštěné služby.

Knihovny potřebné k sestavení a spuštění klientů .NET Framework 1 a 2

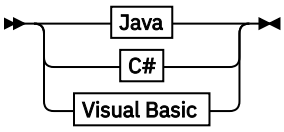
- amqsoap
- Systém
- System.Web.Services
- System.Xml

Knihovny potřebné k sestavení a spuštění klientů Axis 1.4

- *MQ_Install*\java\lib\com.ibm.mq.soap.jar;
- *MQ_Install*\java\lib\com.ibm.mq.commonservices.jar;
- *MQ_Install*\java\lib\soap\axis.jar;
- *MQ_Install*\java\lib\soap\jaxrpc.jar
- *MQ_Install*\java\lib\soap\saaj.jar;
- *MQ_Install*\java\lib\soap\commons-logging-1.0.4.jar;
- *MQ_Install*\java\lib\soap\commons-discovery-0.2.jar;
- *MQ_Install*\java\lib\soap\wsdl4j-1.5.1.jar;
- *MQ_Install*\java\jre\lib\xml.jar;

- *MQ_Install*\java\lib\soap\servlet.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jar;
- *MQ_Install*\java\lib\com.ibm.mq.headers.jar;
- *MQ_Install*\java\lib\com.ibm.mq.pcf.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.remote.jar;
- *MQ_Install*\java\lib\com.ibm.mq.jmqi.local.jar;
- *MQ_Install*\java\lib\connector.jar;
- *MQ_Install*\java\lib\jta.jar;
- *MQ_Install*\java\lib\jndi.jar;
- *MQ_Install*\java\lib\ldap.jar

Register SOAP extension



Java

C#

Visual Basic

Příklady klientů

Obrázek 35 na stránce 975 je příkladem klienta .NET Framework 1 nebo .NET Framework 2 C#, který používá vložený programovací model. Metoda **IBM.WMQSOAP.Register.Extension()** registruje odesilatele WebSphere MQ SOAP s rozhraním .NET jako obslužnou rutinu protokolu `jms:` .

```
using System;
namespace QuoteClientProgram {
    class QuoteMain {
        static void Main(string[] args) {
            try {
                IBM.WMQSOAP.Register.Extension();
                Quote q = new Quote();
                Console.WriteLine("Response is: " + q.getQuote("ibm"));
             }  catch (Exception e) {
                Console.WriteLine("Exception is: " + e);
             }
        }
    }
}
```

Obrázek 35. C# Ukázka klienta webové služby

Obrázek 36 na stránce 976 je příklad klienta Java, který používá statické rozhraní klienta JAX-RPC. Metoda **com.ibm.mq.soap.Register.extension();** registruje odesílatele SOAP produktu WebSphere MQ pomocí serveru proxy služby pro zpracování protokolu `jms:` .

```
package org.example.www;
import com.ibm.mq.soap.Register;
public class QuoteClient {
   public static void main(String[] args) {
      try {
         Register.extension();
         QuoteSOAPImplServiceLocator locator = new QuoteSOAPImplServiceLocator();
         System.out.println("Response = "
                  + locator.getOrgExampleWwwQuoteSOAPImpl_Wmq().getQuote("IBM"));
      } catch (Exception e) {
         System.out.println("Exception = " + e.getMessage());
      }
   }
}
```

Obrázek 36. Příklad klienta webové služby Java

Uživatelské procedury, uživatelské procedury rozhraní API a odkazy na instalovatelné služby

Pomocí odkazů uvedených v této části můžete rozvíjet uživatelské procedury, uživatelské procedury rozhraní API a instalovatelné aplikace služeb:

- “Struktura MQIEP” na stránce 977
- “Odkaz na výstupní bod pro převod dat” na stránce 980
- “MQ_PUBLISH_EXIT-Uživatelská procedura publikování” na stránce 983
- “Volání uživatelských procedur kanálů a datové struktury” na stránce 991
- “Popis uživatelské procedury rozhraní” na stránce 1053
- “Referenční informace o rozhraní instalovatelných služeb” na stránce 1113

Související pojmy

“Odkaz na aplikace MQI” na stránce 7
Prostřednictvím odkazů uvedených v této části můžete vyvíjet aplikace MQI s následujícími odkazy:

“Třídy IBM WebSphere MQ pro knihovny Java” na stránce 1377
Umístění tříd produktu IBM WebSphere MQ pro knihovny Java se liší v závislosti na platformě. Uveďte toto umístění, když spustíte aplikaci.

Související úlohy

Vývoj aplikací

Související odkazy

“Odkaz SOAP” na stránce 919
Přenos produktu WebSphere MQ pro referenční informace SOAP uspořádané abecedně.

“Referenční materiál pro most produktu IBM WebSphere MQ pro protokol HTTP” na stránce 1175
Referenční témata pro most IBM WebSphere MQ pro HTTP, která jsou uspořádána abecedně

“Třídy a rozhraní .NET produktuIBM WebSphere MQ” na stránce 1209
Třídy a rozhraní .NET produktu IBM WebSphere MQ jsou seřazeny abecedně. Jsou popsány vlastnosti, metody a konstruktory.

“IBM WebSphere MQ Třídy C++” na stránce 1270
Třídy jazyka C++ produktu IBM WebSphere MQ zapouzdřují rozhraní MQI (Message Queue Interface) produktu IBM WebSphere MQ . K dispozici je jeden soubor záhlaví C + +, **imqi.hpp**, který pokrývá všechny tyto třídy.

Třídy WebSphere MQ pro rozhraní JMS

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 informace:

- 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 assembleru.

Postup při psaní programů je uveden v následujících tématech:

- Psaní uživatelské procedury pro převod dat pro produkt WebSphere MQ v systémech UNIX and Linux
- Psaní uživatelské procedury pro převod dat pro produkt WebSphere 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 588 na stránce 980.

Tabulka 588. Zdrojové soubory skeletonu

Platforma	**Soubor**
AIX	amqsvfc0.c
IBM i	QMQMSAMP/QCSRC (AMQSVFC4)

Tabulka 588. Zdrojové soubory skeletonu (pokračování)	
Platforma	**Soubor**
HP-UX	amqsvfc0.c
Linux	amqsvfc0.c
z/OS	CSQ4BAX8 (“1” na stránce 981) CSQ4BAX9 (“2” na stránce 981) CSQ4CAX9 (“3” na stránce 981)
Solaris	amqsvfc0.c
Systémy Windows	amqsvfc0.c

Notes:

1. Znázorňuje volání MQXCVNC.
2. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití ve všech prostředích s výjimkou CICS.
3. Modul wrapper pro fragmenty kódu generované obslužným programem pro použití v prostředí 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 znaková sada UCS2) musí být použity 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 867 .

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
CVTMQMDTA (Konverze datového typu WebSphere MQ)

Windows, systémy UNIX and Linux
`crtmqcvx` (Vytvoření převodu WebSphere MQ -ukončení)

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. .

Chybové zprávy v systémech Windows, UNIX and Linux

Příkaz `crtmqcvx` vrací zprávy v rozsahu AMQ7953 až AMQ7970.

Tyto zprávy jsou uvedeny v příručce Kódy příčin *WebSphere MQ Messages*.

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* tyto 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 na systému z/OS

```
 DCL 1 TEST,
       2 SERIAL_NUMBER  FIXED BIN(31),
       2 ID             CHAR(5),
       2 VERSION        FIXED BIN(15),
       2 CODE           CHAR(4),        /* not to be converted */
       2 DIMENSIONS(3)  FIXED BIN(31),
       2 NAME           CHAR(24);
```

MQ_PUBLISH_EXIT-Uživatelská procedura publikování

Volání MQ_PUBLISH_EXIT může zkontrolovat a 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

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 589 na stránce 984 shrnuje pole ve struktuře:

Tabulka 589. 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, MQINQMMP nebo 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 590 na stránce 988 shrnuje pole ve struktuře:

Tabulka 590. 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 591 na stránce 989 shrnuje pole ve struktuře:

Tabulka 591. 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 Inquire Subscription Status (Response).

SubType je vstupní pole pro ukončení.

***SubOptions* (MQLONG)**
SubOptions jsou volby odběru; viz “Volby (MQLONG)” na stránce 527 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 526, 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; vizSelektory .

SelectionString je vstupní pole pro ukončení.

***SubLevel* (MQLONG)**
SubLevel je úroveň zachycení přidružená k odběru; další podrobnosti viz “SubLevel (MQLONG)” na stránce 537 .

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 produktu WebSphere 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 WebSphere MQ lze zapsat v následujících programovacích jazycích:

Platforma	Programovací jazyky
WebSphere 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*.)
WebSphere MQ for IBM i	ILE C, ILE COBOL a ILE RPG
Všechny ostatní platformy WebSphere MQ	C

Uživatelské procedury v jazyce Java můžete také psát pro použití pouze s aplikacemi Java a JMS. Další informace o vytváření a použití kanálů pro třídy WebSphere MQ pro prostředí Java naleznete v tématu Použití uživatelských procedur kanálů v třídách WebSphere MQ pro jazyk Java a pro třídy WebSphere MQ pro platformu JMS naleznete v tématu Použití kanálu pro uživatelské procedury se třídami produktu WebSphere MQ pro platformu JMS.

Uživatelské procedury produktu WebSphere MQ nelze zapsat v produktu TAL nebo Visual Basic. Deklarace struktury MQCD je však k dispozici ve Visual Basic pro použití v rámci volání MQCONNX z klientského programu WebSphere MQ MQI.

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 217.

Definiční soubory dat

Soubory definic dat jsou dodávány s produktem WebSphere MQ pro každý z podporovaných programovacích jazyků. 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 ×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*; tato hodnota 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*; tato hodnota 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.

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).

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 1037 .
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 999 .

MQXWAIT-Čekání na ukončení

Volání MQXWAIT čeká na výskyt události. Lze ji použít pouze z uživatelské procedury kanálu na systému 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 od ECB MVS (řídicí blok událostí). ECB je popsána v popisu řídicího bloku MQXWD.

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 1052 .

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 části “MQ_CHANNEL_EXIT-Ukončení kanálu” na stránce 992. 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:

Systémy UNIX
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 1000
Toto téma uvádí všechna pole ve struktuře MQCD a popisuje každé pole.

"Deklarace C" na stránce 1025
Toto deklarace je deklarací C pro strukturu MQCD.

"Deklarace COBOL" na stránce 1027
Toto deklarace je deklarací COBOL pro strukturu MQCD.

"Deklarace RPG (ILE)" na stránce 1029
Toto prohlášení je deklarací RPG pro strukturu MQCD.

"Deklarace asembleru System/390" na stránce 1032
Toto prohlášení je deklarací assembleru System/390 pro strukturu MQCD.

"Deklarace jazyka Visual Basic" na stránce 1033
Toto prohlášení je prohlášení o Visual Basicu struktury MQCD.

"Změna polí MQCD v uživatelské proceduře kanálu" na stránce 1035
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.

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.

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.

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 kanálu na velkém serveru UNIX větší "váhu" než jiný kanál na malém PC počítače, a zvolit algoritmus zvolí server UNIX 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.

V případě klientských aplikací napsaných v jazycích C, C++ nebo .NET Framework (včetně plně spravovaného prostředí .NET) se seznam aktualizuje, pokud byla tabulka CCDT od vytvoření seznamu upravena.

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.

V případě klientských aplikací napsaných v jazycích C, C++ nebo .NET Framework (včetně plně spravovaného prostředí .NET) se seznam aktualizuje, pokud byla tabulka CCDT od vytvoření seznamu upravena.

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 WebSphere MQ MQI client můžete nakonfigurovat tak, aby znovu automaticky připojil aplikaci klienta. Produkt IBM WebSphere 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|MQRCN_YES|MQRCN_Q_MGR|
MQRCN_DISABLED)
```

Parametry

MQRCN_NO
MQRCN_NO je výchozí hodnota.

Pokud není přepsáno produktem MQCONNX, klient se automaticky nepřipojí automaticky.

MQRCN_YES
Pokud není přepsáno produktem MQCONNX, klient se znovu připojí automaticky.

MQRCN_Q_MGR
Pokud nebude přepsán produktem MQCONNX, klient se znovu připojí automaticky, ale pouze se stejným správcem front. Volba QMGR má stejný účinek jako MQCNO_RECONNECT_Q_MGR.

MQRCN_DISABLED
Opětovné připojení je zakázáno, i když je vyžádáno klientským programem pomocí volání modulu MQCONNX MQI.

Automatické opětovné připojení klienta není podporováno třídami IBM WebSphere MQ pro jazyk Java.

Tabulka 592. 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
stanza CHANNELS konfiguračního souboru klienta
Související odkazy
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 k serveru se používá pouze mezi voláními rozhraní API produktu WebSphere MQ z 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.

- Na systému z/OSse při zadání parametru TCPKEEP (YES) v objektu správce front vyskytne udržení aktivity TCP/IP.
- 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 volitelné informace, `ip-addr` je uveden v desítkové tečkové notaci IPv4 , v hexadecimálním formátu IPv6 nebo alfanumerickém tvaru 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.

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 (IBM i, UNIX, a Windows).

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, určuje identifikátor uživatele, který má být použit agentem kanálu zpráv pro autorizaci pro přístup k prostředkům produktu WebSphere 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 999 , 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 999 , 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 produkt WebSphere 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 produkt WebSphere 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 neprázdné pouze v systémech UNIX a Windows a 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 následující:

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 vlastnosti související s rozhraním JMS, být 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 114
“MQPROP_ * (kontrolní hodnoty vlastností fronty a kanálu a maximální délka vlastností)” na stránce 151

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 uživatel připojit, který v systémech UNIX, Linux a Windows je vždy neprázdný.

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 999 , 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 produkt WebSphere 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 999 , 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 produkt WebSphere 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 999 , 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 produkt WebSphere 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 stejně jako ve verzích starších než WebSphere MQ verze 7.0 s ohledem na následující atributy:

- Sdílení konverzace
- Dopředné čtení
- `STOP CHANNEL(<channelname>) MODE(QUIESCE)`
- Synchronizační signály
- Asynchronní spotřeba klienta

Hodnota 1 je minimální hodnota pro chování produktu WebSphere MQ V7.0 . Přestože je na instanci kanálu povolena pouze jedna konverzace, je k dispozici asynchronní spotřeba a verze 7 prezenčního signálu `CLNTCONN-SVRCONN` a zastavení kanálu v klidovém kanálu.

Toto je vstupní pole pro ukončení. Není přítomna, pokud *Version* je menší než MQCD_VERSION_9.

Výchozí hodnota tohoto pole je 10.

Poznámka: Limity kanálu *MaxInstances* a *MaxInstancesPerClient* aplikované na kanál omezují počet instancí kanálu, nikoli počet konverzací, které mohou 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.

SSLCipherSpec (MQCHAR32)
Toto pole uvádí specifikaci šifry, která se používá při používání SSL.

Je-li hodnota SSLCipherSpec prázdná, kanál nepoužívá SSL. Pokud pole není prázdné, obsahuje toto pole řetězec určující použití CipherSpec .

Tento parametr je platný pro všechny typy kanálů. Je podporován v systémech AIX, HP-UX, Linux, IBM i, Solaris, Windowsa 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 SSL.

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 SSL, 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 SSL.

Je-li během úspěšného navázání komunikace SSL obdržen certifikát, je rozlišující název předmětu certifikátu zkopírován do pole MQCD, ke kterému má přístup parametr 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: Uživatelské aplikace zabezpečení vytvořené před vydáním produktu WebSphere MQ v7.1 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 .

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 WebSphere 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 neprázdné pouze v systémech UNIX a Windows a 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.

Verze 2 není používána žádným aktuálním produktem IBM WebSphere MQ .

MQCD_VERSION_3
Struktura definice kanálu verze 3.

Verze 3 je nejvyšší, abyste mohli nastavit pole na MQSeries verze 2 v následujících prostředích: HP Integrity NonStop Servera systémy UNIX and Linux , které nejsou uvedeny jinde.

MQCD_VERSION_4
Struktura definice kanálu verze 4.

Verze 4 není používána žádným aktuálním produktem IBM WebSphere MQ .

MQCD_VERSION_5
Struktura definice kanálu verze 5.

Verze 5 je nejvyšší hodnota, kterou lze nastavit na serveru MQSeries for OS/390 verze 5, vydání 2.

MQCD_VERSION_6
Struktura definice kanálu verze 6.

Verze 6 není aktuální verzí struktury MQCD žádného existujícího produktu IBM WebSphere MQ . Strukturu MQCD verze 6 však lze předat produktu MQCONNX pomocí polí ClientConnOffset nebo ClientConnPtr ve struktuře MQCNO .

Na distribuovaných platformách verze 6 je výchozí verzí v inicializátorů MQCD_DEFAULT a MQCD_CLIENT_CONN_DEFAULT . Chcete-li odkazovat na pole MQCD_VERSION_7, MQCD_VERSION_8nebo MQCD_VERSION_9 v poli MQCD, explicitně inicializujte pole MQCD **Version** na MQCD_VERSION_7, MQCD_VERSION_8nebo MQCD_VERSION_9 .

V systému z/OSje výchozí hodnotou MQCD_VERSION_7 .

MQCD_VERSION_7
Struktura definice kanálu verze 7.

Verze 7 je nejvyšší, že můžete nastavit pole na IBM WebSphere MQ Version 5.3 v následujících prostředích: AIX, HP-UX, Solaris, Windowsa na systémech IBM WebSphere MQ for z/OS Version 5.3 a Version 5.3.1. MQCD_VERSION_7 je výchozí hodnota pro verze produktu IBM WebSphere MQ for z/OS.

MQCD_VERSION_8
Struktura definice kanálu verze 8.

Verze 8 je nejvyšší, kterou lze nastavit na hodnotu IBM WebSphere MQ Version 6.0 na všech platformách.

MQCD_VERSION_9
Struktura definice kanálu verze 9.

Verze 9 je nejvyšší, když můžete nastavit pole na IBM WebSphere MQ Version 7.0 a IBM WebSphere MQ Version 7.0.1 na všech platformách.

MQCD_VERSION_10
Struktura definice kanálu verze 10.

Verze 10 je nejvyšší hodnotou, kterou můžete nastavit na IBM WebSphere MQ Version 7.1 a IBM WebSphere MQ Version 7.5 na všech platformách.

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 Version 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.

```
typedef struct tagMQCD MQCD;
 typedef MQCD  MQPOINTER PMQCD;
 typedef PMQCD MQPOINTER PPMQCD;

 struct tagMQCD {
   MQCHAR    ChannelName[20];           /* Channel definition name */
   MQLONG    Version;                   /* Structure version number */
   MQLONG    ChannelType;               /* Channel type */
   MQLONG    TransportType;             /* Transport type */
   MQCHAR    Desc[64];                  /* Channel description */
   MQCHAR    QMgrName[48];              /* Queue-manager name */
   MQCHAR    XmitQName[48];             /* Transmission queue name */
   MQCHAR    ShortConnectionName[20];   /* First 20 bytes of */
                                        /* connection name */
   MQCHAR    MCAName[20];               /* Reserved */
   MQCHAR    ModeName[8];               /* LU 6.2 Mode name */
   MQCHAR    TpName[64];                /* LU 6.2 transaction program */
                                        /* name */
   MQLONG    BatchSize;                 /* Batch size */
   MQLONG    DiscInterval;              /* Disconnect interval */
   MQLONG    ShortRetryCount;           /* Short retry count */
   MQLONG    ShortRetryInterval;        /* Short retry wait interval */
   MQLONG    LongRetryCount;            /* Long retry count */
   MQLONG    LongRetryInterval;         /* Long retry wait interval */
   MQCHAR    SecurityExit[128];         /* Channel security exit name */
   MQCHAR    MsgExit[128];              /* Channel message exit name */
   MQCHAR    SendExit[128];             /* Channel send exit name */
   MQCHAR    ReceiveExit[128];          /* Channel receive exit name */
   MQLONG    SeqNumberWrap;             /* Highest allowable message */
                                        /* sequence number */
   MQLONG    MaxMsgLength;              /* Maximum message length */
   MQLONG    PutAuthority;              /* Put authority */
   MQLONG    DataConversion;            /* Data conversion */
   MQCHAR    SecurityUserData[32];      /* Channel security exit user */
                                        /* data */
   MQCHAR    MsgUserData[32];           /* Channel message exit user */
                                        /* data */
   MQCHAR    SendUserData[32];          /* Channel send exit user */
                                        /* data */
   MQCHAR    ReceiveUserData[32];       /* Channel receive exit user */
                                        /* data */
   /* Ver:1 */
   MQCHAR    UserIdentifier[12];        /* User identifier */
   MQCHAR    Password[12];              /* Password */
   MQCHAR    MCAUserIdentifier[12];     /* First 12 bytes of MCA user */
                                        /* identifier */
   MQLONG    MCAType;                   /* Message channel agent type */
   MQCHAR    ConnectionName[264];       /* Connection name */
   MQCHAR    RemoteUserIdentifier[12];  /* First 12 bytes of user */
                                        /* identifier from partner */
   MQCHAR    RemotePassword[12];        /* Password from partner */
   /* Ver:2 */
   MQCHAR    MsgRetryExit[128];         /* Channel message retry exit */
                                        /* name */
   MQCHAR    MsgRetryUserData[32];      /* Channel message retry exit */
                                        /* user data */
   MQLONG    MsgRetryCount;             /* Number of times MCA will */
                                        /* try to put the message, */
                                        /* after first attempt has */
                                        /* failed */
   MQLONG    MsgRetryInterval;          /* Minimum interval in */
                                        /* milliseconds after which */
```

```
                                    /* the open or put operation */
                                    /* will be retried */
  /* Ver:3 */
  MQLONG    HeartbeatInterval;      /* Time in seconds between */
                                    /* heartbeat flows */
  MQLONG    BatchInterval;          /* Batch duration */
  MQLONG    NonPersistentMsgSpeed;  /* Speed at which */
                                    /* nonpersistent messages are */
                                    /* sent */
  MQLONG    StrucLength;            /* Length of MQCD structure */
  MQLONG    ExitNameLength;         /* Length of exit name */
  MQLONG    ExitDataLength;         /* Length of exit user data */
  MQLONG    MsgExitsDefined;        /* Number of message exits */
                                    /* defined */
  MQLONG    SendExitsDefined;       /* Number of send exits */
                                    /* defined */
  MQLONG    ReceiveExitsDefined;    /* Number of receive exits */
                                    /* defined */
  MQPTR     MsgExitPtr;             /* Address of first MsgExit */
                                    /* field */
  MQPTR     MsgUserDataPtr;         /* Address of first */
                                    /* MsgUserData field */
  MQPTR     SendExitPtr;            /* Address of first SendExit */
                                    /* field */
  MQPTR     SendUserDataPtr;        /* Address of first */
                                    /* SendUserData field */
  MQPTR     ReceiveExitPtr;         /* Address of first */
                                    /* ReceiveExit field */
  MQPTR     ReceiveUserDataPtr;     /* Address of first */
                                    /* ReceiveUserData field */
  /* Ver:4 */
  MQPTR     ClusterPtr;             /* Address of a list of */
                                    /* cluster names */
  MQLONG    ClustersDefined;        /* Number of clusters to */
                                    /* which the channel belongs */
  MQLONG    NetworkPriority;        /* Network priority */
  /* Ver:5 */
  MQLONG    LongMCAUserIdLength;    /* Length of long MCA user */
                                    /* identifier */
  MQLONG    LongRemoteUserIdLength; /* Length of long remote user */
                                    /* identifier */
  MQPTR     LongMCAUserIdPtr;       /* Address of long MCA user */
                                    /* identifier */
  MQPTR     LongRemoteUserIdPtr;    /* Address of long remote */
                                    /* user identifier */
  MQBYTE40  MCASecurityId;          /* MCA security identifier */
  MQBYTE40  RemoteSecurityId;       /* Remote security identifier */
  /* Ver:6 */
  MQCHAR    SSLCipherSpec[32];      /* SSL CipherSpec */
  MQPTR     SSLPeerNamePtr;         /* Address of SSL peer name */
  MQLONG    SSLPeerNameLength;      /* Length of SSL peer name */
  MQLONG    SSLClientAuth;          /* Whether SSL client */
                                    /* authentication is required */
  MQLONG    KeepAliveInterval;      /* Keepalive interval */
  MQCHAR    LocalAddress[48];       /* Local communications */
                                    /* address */
  MQLONG    BatchHeartbeat;         /* Batch heartbeat interval */
  /* Ver:7 */
  MQLONG    HdrCompList[2];         /* Header data compression */
                                    /* list */
  MQLONG    MsgCompList[16];        /* Message data compression */
                                    /* list */
  MQLONG    CLWLChannelRank;        /* Channel rank */
  MQLONG    CLWLChannelPriority;    /* Channel priority */
  MQLONG    CLWLChannelWeight;      /* Channel weight */
  MQLONG    ChannelMonitoring;      /* Channel monitoring */
  MQLONG    ChannelStatistics;      /* Channel statistics */
  /* Ver:8 */
  MQLONG    SharingConversations;   /* Limit on sharing */
                                    /* conversations */
  MQLONG    PropertyControl;        /* Message property control */
  MQLONG    MaxInstances;           /* Limit on SVRCONN channel */
                                    /* instances */
  MQLONG    MaxInstancesPerClient;  /* Limit on SVRCONN channel */
                                    /* instances per client */
  MQLONG    ClientChannelWeight;    /* Client channel weight */
  MQLONG    ConnectionAffinity;     /* Connection affinity */
  /* Ver:9 */
  MQLONG    BatchDataLimit;         /* Batch data limit */
  MQLONG    UseDLQ;                 /* Use Dead Letter Queue */
  MQLONG    DefReconnect;           /* Default client reconnect */
                                    /* option */
```

```
  /* Ver:10 */
 };
```

Deklarace COBOL

Toto deklarace je deklarací COBOL pro strukturu MQCD.

```
** MQCD structure
       10  MQCD.
      ** Channel definition name
       15  MQCD-CHANNELNAME PIC X(20).
      ** Structure version number
       15  MQCD-VERSION PIC S9(9) BINARY.
      ** Channel type
       15  MQCD-CHANNELTYPE PIC S9(9) BINARY.
      ** Transport type
       15  MQCD-TRANSPORTTYPE PIC S9(9) BINARY.
      ** Channel description
       15  MQCD-DESC PIC X(64).
      ** Queue-manager name
       15  MQCD-QMGRNAME PIC X(48).
      ** Transmission queue name
       15  MQCD-XMITQNAME PIC X(48).
      ** First 20 bytes of connection name
       15  MQCD-SHORTCONNECTIONNAME PIC X(20).
      ** Reserved
       15  MQCD-MCANAME PIC X(20).
      ** LU 6.2 Mode name
       15  MQCD-MODENAME PIC X(8).
      ** LU 6.2 transaction program name
       15  MQCD-TPNAME PIC X(64).
      ** Batch size
       15  MQCD-BATCHSIZE PIC S9(9) BINARY.
      ** Disconnect interval
       15  MQCD-DISCINTERVAL PIC S9(9) BINARY.
      ** Short retry count
       15  MQCD-SHORTRETRYCOUNT PIC S9(9) BINARY.
      ** Short retry wait interval
       15  MQCD-SHORTRETRYINTERVAL PIC S9(9) BINARY.
      ** Long retry count
       15  MQCD-LONGRETRYCOUNT PIC S9(9) BINARY.
      ** Long retry wait interval
       15  MQCD-LONGRETRYINTERVAL PIC S9(9) BINARY.
      ** Channel security exit name
       15  MQCD-SECURITYEXIT PIC X(20).
      ** Channel message exit name
       15  MQCD-MSGEXIT PIC X(20).
      ** Channel send exit name
       15  MQCD-SENDEXIT PIC X(20).
      ** Channel receive exit name
       15  MQCD-RECEIVEEXIT PIC X(20).
      ** Highest allowable message sequence number
       15  MQCD-SEQNUMBERWRAP PIC S9(9) BINARY.
      ** Maximum message length
       15  MQCD-MAXMSGLENGTH PIC S9(9) BINARY.
      ** Put authority
       15  MQCD-PUTAUTHORITY PIC S9(9) BINARY.
      ** Data conversion
       15  MQCD-DATACONVERSION PIC S9(9) BINARY.
      ** Channel security exit user data
       15  MQCD-SECURITYUSERDATA PIC X(32).
      ** Channel message exit user data
       15  MQCD-MSGUSERDATA PIC X(32).
      ** Channel send exit user data
       15  MQCD-SENDUSERDATA PIC X(32).
      ** Channel receive exit user data
       15  MQCD-RECEIVEUSERDATA PIC X(32).
      ** Ver:1 **
      ** User identifier
       15  MQCD-USERIDENTIFIER PIC X(12).
      ** Password
       15  MQCD-PASSWORD PIC X(12).
      ** First 12 bytes of MCA user identifier
       15  MQCD-MCAUSERIDENTIFIER PIC X(12).
      ** Message channel agent type
       15  MQCD-MCATYPE PIC S9(9) BINARY.
      ** Connection name
       15  MQCD-CONNECTIONNAME PIC X(264).
      ** First 12 bytes of user identifier from partner
       15  MQCD-REMOTEUSERIDENTIFIER PIC X(12).
```

```
** Password from partner
 15  MQCD-REMOTEPASSWORD PIC X(12).
** Ver:2 **
** Channel message retry exit name
 15  MQCD-MSGRETRYEXIT PIC X(20).
** Channel message retry exit user data
 15  MQCD-MSGRETRYUSERDATA PIC X(32).
** Number of times MCA will try to put the message, after first
** attempt has failed
 15  MQCD-MSGRETRYCOUNT PIC S9(9) BINARY.
** Minimum interval in milliseconds after which the open or put
** operation will be retried
 15  MQCD-MSGRETRYINTERVAL PIC S9(9) BINARY.
** Ver:3 **
** Time in seconds between heartbeat flows
 15  MQCD-HEARTBEATINTERVAL PIC S9(9) BINARY.
** Batch duration
 15  MQCD-BATCHINTERVAL PIC S9(9) BINARY.
** Speed at which nonpersistent messages are sent
 15  MQCD-NONPERSISTENTMSGSPEED PIC S9(9) BINARY.
** Length of MQCD structure
 15  MQCD-STRUCLENGTH PIC S9(9) BINARY.
** Length of exit name
 15  MQCD-EXITNAMELENGTH PIC S9(9) BINARY.
** Length of exit user data
 15  MQCD-EXITDATALENGTH PIC S9(9) BINARY.
** Number of message exits defined
 15  MQCD-MSGEXITSDEFINED PIC S9(9) BINARY.
** Number of send exits defined
 15  MQCD-SENDEXITSDEFINED PIC S9(9) BINARY.
** Number of receive exits defined
 15  MQCD-RECEIVEEXITSDEFINED PIC S9(9) BINARY.
** Address of first MsgExit field
 15  MQCD-MSGEXITPTR POINTER.
** Address of first MsgUserData field
 15  MQCD-MSGUSERDATAPTR POINTER.
** Address of first SendExit field
 15  MQCD-SENDEXITPTR POINTER.
** Address of first SendUserData field
 15  MQCD-SENDUSERDATAPTR POINTER.
** Address of first ReceiveExit field
 15  MQCD-RECEIVEEXITPTR POINTER.
** Address of first ReceiveUserData field
 15  MQCD-RECEIVEUSERDATAPTR POINTER.
** Ver:4 **
** Address of a list of cluster names
 15  MQCD-CLUSTERPTR POINTER.
** Number of clusters to which the channel belongs
 15  MQCD-CLUSTERSDEFINED PIC S9(9) BINARY.
** Network priority
 15  MQCD-NETWORKPRIORITY PIC S9(9) BINARY.
** Ver:5 **
** Length of long MCA user identifier
 15  MQCD-LONGMCAUSERIDLENGTH PIC S9(9) BINARY.
** Length of long remote user identifier
 15  MQCD-LONGREMOTEUSERIDLENGTH PIC S9(9) BINARY.
** Address of long MCA user identifier
 15  MQCD-LONGMCAUSERIDPTR POINTER.
** Address of long remote user identifier
 15  MQCD-LONGREMOTEUSERIDPTR POINTER.
** MCA security identifier
 15  MQCD-MCASECURITYID PIC X(40).
** Remote security identifier
 15  MQCD-REMOTESECURITYID PIC X(40).
** Ver:6 **
** SSL CipherSpec
 15  MQCD-SSLCIPHERSPEC PIC X(32).
** Address of SSL peer name
 15  MQCD-SSLPEERNAMEPTR POINTER.
** Length of SSL peer name
 15  MQCD-SSLPEERNAMELENGTH PIC S9(9) BINARY.
** Whether SSL client authentication is required
 15  MQCD-SSLCLIENTAUTH PIC S9(9) BINARY.
** Keepalive interval
 15  MQCD-KEEPALIVEINTERVAL PIC S9(9) BINARY.
** Local communications address
 15  MQCD-LOCALADDRESS PIC X(48).
** Batch heartbeat interval
 15  MQCD-BATCHHEARTBEAT PIC S9(9) BINARY.
** Ver:7 **
** Header data compression list
 15  MQCD-HDRCOMPLIST PIC S9(9) BINARY.
```

```
      ** Message data compression list
       15  MQCD-MSGCOMPLIST PIC S9(9) BINARY.
      ** Channel rank
       15  MQCD-CLWLCHANNELRANK PIC S9(9) BINARY.
      ** Channel priority
       15  MQCD-CLWLCHANNELPRIORITY PIC S9(9) BINARY.
      ** Channel weight
       15  MQCD-CLWLCHANNELWEIGHT PIC S9(9) BINARY.
      ** Channel monitoring
       15  MQCD-CHANNELMONITORING PIC S9(9) BINARY.
      ** Channel statistics
       15  MQCD-CHANNELSTATISTICS PIC S9(9) BINARY.
      ** Ver:8 **
      ** Limit on sharing conversations
       15  MQCD-SHARINGCONVERSATIONS PIC S9(9) BINARY.
      ** Message property control
       15  MQCD-PROPERTYCONTROL PIC S9(9) BINARY.
      ** Limit on SVRCONN channel instances
       15  MQCD-MAXINSTANCES PIC S9(9) BINARY.
      ** Limit on SVRCONN channel instances per client
       15  MQCD-MAXINSTANCESPERCLIENT PIC S9(9) BINARY.
      ** Client channel weight
       15  MQCD-CLIENTCHANNELWEIGHT PIC S9(9) BINARY.
      ** Connection affinity
       15  MQCD-CONNECTIONAFFINITY PIC S9(9) BINARY.
      ** Ver:9 **
      ** Batch data limit
       15  MQCD-BATCHDATALIMIT PIC S9(9) BINARY.
      ** Use Dead Letter Queue
       15  MQCD-USEDLQ PIC S9(9) BINARY.
      ** Default client reconnect option
       15  MQCD-DEFRECONNECT PIC S9(9) BINARY.
      ** Ver:10 **
```

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* SSL CipherSpec
D  CDSCS               1217   1248
D* Address of SSL peer name
D  CDSPN               1249   1264*
D* Length of SSL peer name
D  CDSPL               1265   1268I 0
D* Whether SSL client authentication is required
D  CDSCA               1269   1272I 0
D* Keepalive interval
D  CDKAI               1273   1276I 0
D* Local communications address
D  CDLOA               1277   1324
D* Batch heartbeat interval
D  CDBHB               1325   1328I 0
D* Ver:7 **
D* Header data compression list
D  CDHCL0
D  CDHCL1              1329   1332I 0
D  CDHCL2              1333   1336I 0
D  CDHCL                        10I 0 DIM(2) OVERLAY(CDHCL0)
D* Message data compression list
D  CDMCL0
D  CDMCL1              1337   1340I 0
D  CDMCL2              1341   1344I 0
D  CDMCL3              1345   1348I 0
D  CDMCL4              1349   1352I 0
D  CDMCL5              1353   1356I 0
D  CDMCL6              1357   1360I 0
D  CDMCL7              1361   1364I 0
D  CDMCL8              1365   1368I 0
D  CDMCL9              1369   1372I 0
D  CDMCL10             1373   1376I 0
D  CDMCL11             1377   1380I 0
D  CDMCL12             1381   1384I 0
D  CDMCL13             1385   1388I 0
D  CDMCL14             1389   1392I 0
D  CDMCL15             1393   1396I 0
D  CDMCL16             1397   1400I 0
D  CDMCL                        10I 0 DIM(16) OVERLAY(CDMCL0)
D* Channel rank
D  CDCWCR              1401   1404I 0
D* Channel priority
D  CDCWCP              1405   1408I 0
D* Channel weight
D  CDCWCW              1409   1412I 0
D* Channel monitoring
D  CDCHLMON            1413   1416I 0
D* Channel statistics
D  CDCHLST             1417   1420I 0
D* Ver:8 **
D* Limit on sharing conversations
D  CDSHC               1421   1424I 0
D* Message property control
D  CDPRC               1425   1428I 0
D* Limit on SVRCONN channel instances
D  CDMXIN              1429   1432I 0
D* Limit on SVRCONN channel instances per client
D  CDMXIC              1433   1436I 0
D* Client channel weight
D  CDCLNCHLW           1437   1440I 0
D* Connection affinity
D  CDCONNAFF           1441   1444I 0
D* Ver:9 **
D* Batch data limit
D  CDBDL               1445   1448I 0
D* Use Dead Letter Queue
D  CDUDLQ              1449   1452I 0
```

```
     D* Default client reconnect option
     D  CDDRCN              1453   1456I 0
     D* Ver:10 **
```

Deklarace asembleru System/390

Toto prohlášení je deklarací assembleru System/390 pro strukturu MQCD.

```
MQCD                           DSECT
MQCD_CHANNELNAME               DS   CL20   Channel definition name
MQCD_VERSION                   DS   F      Structure version number
MQCD_CHANNELTYPE               DS   F      Channel type
MQCD_TRANSPORTTYPE             DS   F      Transport type
MQCD_DESC                      DS   CL64   Channel description
MQCD_QMGRNAME                  DS   CL48   Queue-manager name
MQCD_XMITQNAME                 DS   CL48   Transmission queue name
MQCD_SHORTCONNECTIONNAME       DS   CL20   First 20 bytes of connection
*                                          name
MQCD_MCANAME                   DS   CL20   Reserved
MQCD_MODENAME                  DS   CL8    LU 6.2 Mode name
MQCD_TPNAME                    DS   CL64   LU 6.2 transaction program name
MQCD_BATCHSIZE                 DS   F      Batch size
MQCD_DISCINTERVAL              DS   F      Disconnect interval
MQCD_SHORTRETRYCOUNT           DS   F      Short retry count
MQCD_SHORTRETRYINTERVAL        DS   F      Short retry wait interval
MQCD_LONGRETRYCOUNT            DS   F      Long retry count
MQCD_LONGRETRYINTERVAL         DS   F      Long retry wait interval
MQCD_SECURITYEXIT              DS   CLn    Channel security exit name
MQCD_MSGEXIT                   DS   CLn    Channel message exit name
MQCD_SENDEXIT                  DS   CLn    Channel send exit name
MQCD_RECEIVEEXIT               DS   CLn    Channel receive exit name
MQCD_SEQNUMBERWRAP             DS   F      Highest allowable message
*                                          sequence number
MQCD_MAXMSGLENGTH              DS   F      Maximum message length
MQCD_PUTAUTHORITY              DS   F      Put authority
MQCD_DATACONVERSION            DS   F      Data conversion
MQCD_SECURITYUSERDATA          DS   CL32   Channel security exit user data
MQCD_MSGUSERDATA               DS   CL32   Channel message exit user data
MQCD_SENDUSERDATA              DS   CL32   Channel send exit user data
MQCD_RECEIVEUSERDATA           DS   CL32   Channel receive exit user data
MQCD_USERIDENTIFIER            DS   CL12   User identifier
MQCD_PASSWORD                  DS   CL12   Password
MQCD_MCAUSERIDENTIFIER         DS   CL12   First 12 bytes of MCA user
*                                          identifier
MQCD_MCATYPE                   DS   F      Message channel agent type
MQCD_CONNECTIONNAME            DS   CL264  Connection name
MQCD_REMOTEUSERIDENTIFIER      DS   CL12   First 12 bytes of user
*                                          identifier from partner
MQCD_REMOTEPASSWORD            DS   CL12   Password from partner
MQCD_MSGRETRYEXIT              DS   CLn    Channel message retry exit name
MQCD_MSGRETRYUSERDATA          DS   CL32   Channel message retry exit user
*                                          data
MQCD_MSGRETRYCOUNT             DS   F      Number of times MCA will try to
*                                          put the message, after the
*                                          first attempt has failed
MQCD_MSGRETRYINTERVAL          DS   F      Minimum interval in
*                                          milliseconds after which the
*                                          open or put operation will be
*                                          retried
MQCD_HEARTBEATINTERVAL         DS   F      Time in seconds between
*                                          heartbeat flows
MQCD_BATCHINTERVAL             DS   F      Batch duration
MQCD_NONPERSISTENTMSGSPEED     DS   F      Speed at which nonpersistent
*                                          messages are sent
MQCD_STRUCLENGTH               DS   F      Length of MQCD structure
MQCD_EXITNAMELENGTH            DS   F      Length of exit name
MQCD_EXITDATALENGTH            DS   F      Length of exit user data
MQCD_MSGEXITSDEFINED           DS   F      Number of message exits defined
MQCD_SENDEXITSDEFINED          DS   F      Number of send exits defined
MQCD_RECEIVEEXITSDEFINED       DS   F      Number of receive exits defined
MQCD_MSGEXITPTR                DS   F      Address of first MSGEXIT field
MQCD_MSGUSERDATAPTR            DS   F      Address of first MSGUSERDATA
*                                          field
MQCD_SENDEXITPTR               DS   F      Address of first SENDEXIT field
MQCD_SENDUSERDATAPTR           DS   F      Address of first SENDUSERDATA
*                                          field
MQCD_RECEIVEEXITPTR            DS   F      Address of first RECEIVEEXIT
*                                          field
MQCD_RECEIVEUSERDATAPTR        DS   F      Address of first
```

```
*                                       RECEIVEUSERDATA field
MQCD_CLUSTERPTR              DS   F      Address of a list of cluster
*                                       names
MQCD_CLUSTERSDEFINED         DS   F      Number of clusters to which the
*                                       channel belongs
MQCD_NETWORKPRIORITY         DS   F      Network priority
MQCD_LONGMCAUSERIDLENGTH     DS   F      Length of long MCA user
*                                       identifier
MQCD_LONGREMOTEUSERIDLENGTH  DS   F      Length of long remote user
*                                       identifier
MQCD_LONGMCAUSERIDPTR        DS   F      Address of long MCA user
*                                       identifier
MQCD_LONGREMOTEUSERIDPTR     DS   F      Address of long remote user
*                                       identifier
MQCD_MCASECURITYID           DS   XL40   MCA security identifier
MQCD_REMOTESECURITYID        DS   XL40   Remote security identifier
MQCD_SSLCIPHERSPEC           DS   CL32   SSL CipherSpec
MQCD_SSLPEERNAMEPTR          DS   F      Address of SSL peer name
MQCD_SSLPEERNAMELENGTH       DS   F      Length of SSL peer name
MQCD_SSLCLIENTAUTH           DS   F      Whether SSL client
*                                       authentication is required
MQCD_KEEPALIVEINTERVAL       DS   F      Keepalive interval
MQCD_LOCALADDRESS            DS   CL48   Local communications address
MQCD_BATCHHEARTBEAT          DS   F      Batch heartbeat interval
MQCD_HDRCOMPLIST             DS   CL2    Header data compression list
MQCD_MSGCOMPLIST             DS   CL16   Message data compression list
MQCD_CLWLCHANNELRANK         DS   F      Channel rank
MQCD_CLWLCHANNELPRIORITY     DS   F      Channel priority
MQCD_CLWLCHANNELWEIGHT       DS   F      Channel weight
MQCD_CHANNELMONITORING       DS   F      Channel monitoring
MQCD_CHANNELSTATISTICS       DS   F      Channel statistics
MQCD_SHARINGCONVERSATIONS    DS   F      Limit on sharing
*                                       conversations
MQCD_PROPERTYCONTROL         DS   F      Message property
*                                       control
MQCD_SHARINGCONVERSATIONS    DS   F      Limit on sharing conversations
MQCD_PROPERTYCONTROL         DS   F      Message property control
MQCD_MAXINSTANCES            DS   F      Limit on SVRCONN chl instances
MQCD_MAXINSTANCESPERCLIENT   DS   F      Limit on SVRCONN chl instances
                                        per client
MQCD_CLIENTCHANNELWEIGHT     DS   F      Channel weight
MQCD_CONNECTIONAFFINITY      DS   F      Connection Affinty
MQCD_BATCHDATALIMIT          DS   F      Batch data limit
MQCD_USEDLQ                  DS   F      Use dead-letter queue
MQCD_DEFRECONNECT            DS   F      Default client reconnect option
MQCD_LENGTH                  EQU  *-MQCD
                             ORG  MQCD
MQCD_AREA                    DS   CL(MQCD_LENGTH)
```

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  'SSL CipherSpec'
  SSLPeerNamePtr         As MQPTR   'Address of SSL peer name'
  SSLPeerNameLength      As Long    'Length of SSL peer name'
  SSLClientAuth          As Long    'Whether SSL client'
                                    'authentication is required'
  KeepAliveInterval      As Long    'Keepalive interval'
  LocalAddress           As String*48  'Local communications address'
  BatchHeartbeat         As Long    'Batch heartbeat interval'
  HdrCompList(0 to 1)    As Long2   'Header data compression list'
  MsgCompList(0 To 15)   As Long16  'Message data compression list'
  CLWLChannelRank        As Long    'Channel Rank'
  CLWLChannelPriority    As Long    'Channel priority'
  CLWLChannelWeight      As Long    'Channel Weight'
  ChannelMonitoring      As Long    'Channel Monitoring control'
  ChannelStatistics      As Long    'Channel Statistics'
End Type
```

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 WebSphere 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.

Pole	**Kód příčiny ukončení**	**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
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

Pole	Kód příčiny ukončení	Typ ukončení	Typ kanálu
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
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

Pole	Kód příčiny ukončení	Typ ukončení	Typ kanálu
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
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 1038
Toto téma uvádí všechna pole v rámci struktury MQCXP a popisuje každé pole.

“Deklarace C” na stránce 1048
Toto prohlášení je prohlášení C pro strukturu MQCXP.

“Deklarace COBOL” na stránce 1049
Toto deklarace je deklarací COBOL pro strukturu MQCXP.

“Deklarace RPG (ILE)” na stránce 1050
Toto prohlášení je deklarací RPG pro strukturu MQCXP.

“Deklarace asembleru System/390” na stránce 1051
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_2
Struktura výstupního parametru kanálu Version-2 .

Toto pole má tuto hodnotu v následujících prostředích: HP Integrity NonStop Server.

MQCXP_VERSION_3
Struktura výstupního parametru kanálu Version-3 .

Toto pole má tuto hodnotu v následujících prostředích: systémy UNIX neuvedené 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 .

Toto pole má tuto hodnotu v následujících prostředích: z/OS, AIX, HP-UX, Linux, IBM i, Solaris, Windows.

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 307

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 999). 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 WebSphere 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 WebSphere MQ při zahájení 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 WebSphere 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 pod vedením služby IBM .Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_3.

CapabilityFlags (MQLONG)
Toto pole uvádí příznaky schopnosti.

Definovány jsou následující:

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 velikost prostoru v bajtech, který funkce kanálu WebSphere kanálu MQ rezervuje v přenosové vyrovnávací paměti pro použití k použití. 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 systému 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 SSL.

SSLRemCertIssNamePtr (PMQVOID)
Toto pole uvádí adresu úplného rozlišujícího názvu vydavatele vzdáleného certifikátu.

Jeho hodnotou je ukazatel Null, pokud se nejedná o kanál SSL.

Toto je vstupní pole pro ukončení. Pole není přítomno, pokud *Version* je menší než MQCXP_VERSION_6.

Poznámka: Chování uživatelských procedur zabezpečení kanálu pro určení rozlišujícího názvu subjektu a rozlišujícího názvu vydavatele se změnilo ve vydání produktu WebSphere MQ v7.1. Další informace najdete v tématu Uživatelské programy zabezpečení kanálu.

SecurityParms (PMQCSP)
Toto pole uvádí adresu struktury MQSCP, která se používá 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.

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 1035.

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 1035 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.

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 SSL certificate */
  MQLONG    SSLRemCertIssNameLength;   /* Length of
                                  distinguished name of issuer
                                  of remote SSL certificate */
  MQPTR     SSLRemCertIssNamePtr;      /* Address of
                                  distinguished name of issuer
                                  of remote SSL certificate */
  PMQVOID   SecurityParms;     /* Security parameters */
  MQLONG    CurHdrCompression; /* Header data compression
                                  used for current message */
  MQLONG    CurMsgCompression; /* Message data compression
                                  used for current message */
  /* Ver:6 */
  MQHCONN   Hconn;             /* Connection handle */
  MQBOOL    SharingConversations;      /* Multiple conversations
                                  possible on channel inst?  */
  /* Ver:7 */
  MQLONG    MCAUserSource;     /* Source of the provided MCA user ID */
  PMQIEP    pEntryPoints;     /* Address of the MQIEP structure */
}
  /* Ver:8 */
;
```

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 SSL
** certificate
   15  MQCXP-SSLREMCERTISSNAMELENGTH PIC S9(9) BINARY.
** Address of distinguished name of issuer of remote SSL
** certificate
   15  MQCXP-SSLREMCERTISSNAMEPTR    POINTER.
** Security parameters
   15  MQCXP-SECURITYPARMS           PIC S9(18) BINARY.
** Header data compression used for current message
   15  MQCXP-CURHDRCOMPRESSION       PIC S9(9) BINARY.
** Message data compression used for current message
   15  MQCXP-CURMSGCOMPRESSION       PIC S9(9) BINARY.
** Connection handle
   15 MQCXP-HCONN                    PIC S9(9) BINARY.
** Multiple conversations possible on channel instance?
   15 MQCXP-SHARINGCONVERSATIONS     PIC S9(9) BINARY.
** Source of the provided MCA user ID
   15 MQCXP-MCAUSERSOURCE            PIC S9(9) BINARY.
```

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 SSL certificate
     D  CXSSLCU              161    172
     D* Length of distinguished name of issuer of remote SSL certificate
     D  CXSRCINL             173    176I 0
     D* Address of distinguished name of issuer of remote SSL certificate
     D  CXSRCINP             177    192*
     D* Security parameters
     D  CXSECP               193    208*
     D* Header data compression used for current message
     D  CXCHC                209    212I 0
     D* Message data compression used for current message
     D  CXCMC                213    216I 0
     D* Connection handle
     D  CXHCONN               217    220I   0
     D* Multiple conversations possible on channel instance?
     D  CXSHARECONV           221    224I   0
     D* Source of the provided MCA user ID
     D  MCAUSERSOURCE         225    228I   0
```

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 SSL certificate
MQCXP_SSLREMCERTISSNAMELENGTH   DS   F     Length of distinguished name
*                                  of issuer of remote SSL certificate
MQCXP_SSLREMCERTISSNAMEPTR      DS   F     Address of distinguished name
*                                  of issuer of remote SSL certificate
MQCXP_SECURITYPARMS     DS   F    Address of security parameters
MQCXP_CURHDRCOMPRESSION         DS   F     Header data compression used for
*                                  current message
MQCXP_CURMSGCOMPRESSION         DS   F      Message data compression used for
*                                  current message
MQCXP_HCONN             DS   F Connection handle
MQCXP_SHARINGCONVERSATIONS      DS   F Multiple conversations possible on
*                                  channel inst?
MQCXP_MCAUSERSOURCE     DS   F    Source of the provided MCA user ID

MQCXP_LENGTH            EQU  *-MQCXP
```

```
                        ORG  MQCXP
MQCXP_AREA              DS   CL(MQCXP_LENGTH)
```

MQXWD-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 1052
Toto téma uvádí všechna pole v rámci struktury MQXWD a popisuje každé pole.

"Deklarace C" na stránce 1053
Toto deklarace je deklarací C pro strukturu MQXWD.

"Deklarace asembleru System/390" na stránce 1053
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)
```

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 pracovat s produkty Websphere MQ verze 7.0 a verzí 7.1 , je napsat uživatelskou proceduru, která obsahuje odkazy ve verzi 7.0 s názvem mqm.Lib a v případě neprimárních nebo přesunutých uživatelských procedur, aby se zajistilo, že aplikace nalezne správné 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 verze 7.0 .)
11. Je-li k dispozici více instalací produktu IBM WebSphere MQ , použijte výstupní procedury vytvořené pro dřívější verzi produktu IBM WebSphere MQ, protože nové funkce přidané v novější verzi nemusí pracovat s dřívějšími verzemi. Další informace o změnách mezi verzemi naleznete v tématu Co se změnilo v produktu WebSphere MQ 7.5.

Struktura výstupních parametrů rozhraní API produktu IBM WebSphere 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 produktu WebSphere 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
Přeskočte funkci rozhraní API produktu WebSphere 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 najdete v tématu "Jak správci front zpracovávají výstupní funkce" na stránce 1059.

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 WebSphere 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 najdete v tématu “Jak správci front zpracovávají výstupní funkce” na stránce 1059.

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 modul API produktu WebSphere 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 zapisuje tuto oblast do trasování produktu WebSphere 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, k němuž je aplikace připojena, která byla vyvolána jako výsledek zpracování volání rozhraní API produktu WebSphere 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.

Další podrobnosti o použití oblasti pro výstupní řetěz najdete v tématu “Hlavička výstupních řetězců a záhlaví oblasti uživatelských řetězců (MQACH)” na stránce 1062 .

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 1064.

Obslužná rutina ukončení nastaví toto pole na správnou hodnotu při každém vstupu do každé funkce ukončení v závislosti na volání rozhraní API produktu WebSphere 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 593 na stránce 1059 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 593. 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
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 WebSphere 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 WebSphere MQ , které vedlo k řízení ukončované funkce ukončení. 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 WebSphere 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 WebSphere 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 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.

Všimněte si, že ukazatel je dokončen pouze v případě, že připojení zpracovává jménem kanálu WebSphere 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.

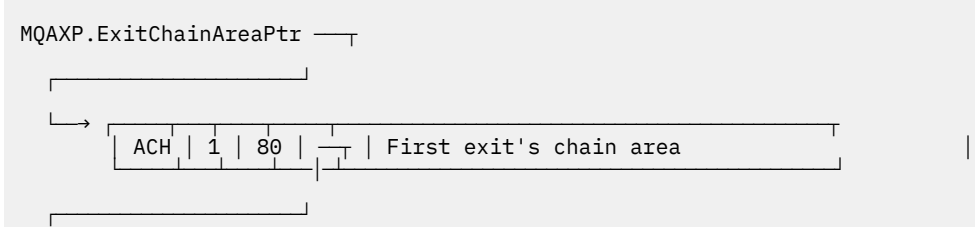
Když uživatelská procedura vytvoří strukturu MQACH, musí inicializovat toto pole vlastním názvem ExitInfo, takže později může být tato struktura MQACH nalezena buď jinou instancí této uživatelské procedury, nebo spolupracujícím výstupem.

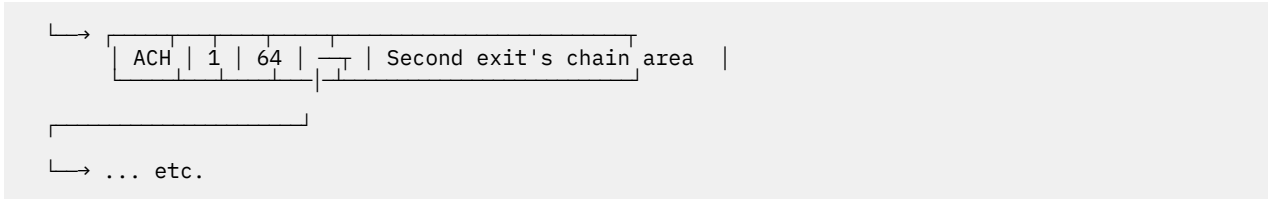
Počáteční hodnota tohoto pole, definovaná parametrem MQACH_DEFAULT, je řetězec s nulovou délkou ({""}).

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 před a za body vyvolání ukončení rozhraní API produktu WebSphere MQ , které mají 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 WebSphere MQ (MQXR_BEFORE)
- Po volání rozhraní API produktu WebSphere MQ (MQXR_AFTER)

Funkce (MQLONG)-vstup
Identifikátor funkce, platné hodnoty, pro které jsou konstanty MQXF_ * (viz “Externí konstanty” na stránce 1064).

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 594 na stránce 1067.)

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 594. Platné kombinace identifikátorů funkcí a ExitReasons

Funkce	**ExitReason**
KONFIGURACE MQXF_INIT VÝRAZ MQXF_TERM	PŘIPOJENÍ MQXR_CONNECTION

Tabulka 594. Platné kombinace identifikátorů funkcí a ExitReasons (pokračování)	
Funkce	**ExitReason**
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:
  - Ve vstupu do uživatelské funkce rozhraní API produktu *před* WebSphere MQ lze hodnotu pole nastavit aplikačním programem nebo předchozí vyvolání funkce ukončení.
  - Na výstupu z funkce uživatelské procedury rozhraní API produktu *před* WebSphere MQ lze hodnotu pole ponechat beze změny nebo ji nastavit na některou jinou hodnotu uživatelskou funkcí.
  - Při vstupu do uživatelské funkce rozhraní API *za* WebSphere MQ může být hodnotou pole hodnota nastavená správcem front po zpracování volání rozhraní API WebSphere MQ nebo může být nastavena na hodnotu při předchozím vyvolání výstupní funkce v řetězci funkcí uživatelské procedury.
  - Na výstupu z funkce ukončení volání funkce *po* WebSphere MQ API lze hodnotu pole ponechat beze změny nebo ji nastavit na některou 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 BEF0RE nemusí být okamžitě následováno voláním AFTER .

Volání BEF0RE může být následováno dalším voláním BEF0RE . 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).
- Vyčištění prováděcího prostředí pro provedení procedury *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. Uživatelská procedura by *neměla* 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 conatin 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 popisované rozhraní funkce MQ_GET_EXIT se používá pro výstupní funkci MQXF_GET i pro uživatelskou proceduru produktu “MQXF_DATA_CONV_ON_GET” na stránce 1084 .

   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:

- Funkce uživatelské procedury připojení pro provedení zpracování *před* a *po* zpracování MQCONN
- Výstupní funkce rozšíření připojení pro provedení operace *před* a *po* zpracování MQCONNX

Stejné rozhraní, které je zde popsáno, je vyvoláno pro funkce ukončení volání MQCONN a MQCONNX.

Když agent kanálu zpráv (MCA) odpovídá na příchozí připojení klienta, může se agent MCA připojit a vytvořit počet volání rozhraní API produktu WebSphere 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 příčiny MQXR_BEFORE *nesmí* vydala všechny volání rozhraní API produktu WebSphere 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 292.

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 1084 .

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 popisované rozhraní funkce MQ_GET_EXIT se používá pro výstupní funkci MQXF_GET i pro uživatelskou proceduru produktu “MQXF_DATA_CONV_ON_GET” na stránce 1084 .

   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

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 MQXEP k registraci příkazových slov WebSphere MQ API a bodů ENTRY a EXIT, v nichž má zájem.
- Ukončení není nutné zachytávat všechny příkazy rozhraní API produktu WebSphere 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 WebSphere 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*), při kterých se mohou vyskytnout chyby. 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 595. 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 pro hodnoty MQXCC_ * a MQXR2_ *[1] 2. Vyčistit prostředí

Tabulka 595. Výstupní chyby rozhraní API a odpovídající akce, které je třeba provést (pokračování)		
Err Pt	**Popis**	**Akce**
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 pro 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 pro 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
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 pro hodnoty MQXCC_ * a MQXR2_ *[1] 2. Jednotka *po* funkci MQ_CONNX_EXIT 1, je-li požadována

Tabulka 595. Výstupní chyby rozhraní API a odpovídající akce, které je třeba provést (pokračování)		
Err Pt	**Popis**	**Akce**
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_ *, jedná se o 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_ *, jedná se o 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_ *, jedná se o 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
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 pro 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 pro 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* WebSphere 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 WebSphere MQ se nevydá.
  - Volání rozhraní API produktu WebSphere 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 WebSphere MQ bylo úspěšně vydáno.
- Pro jinou výstupní funkci volání rozhraní API *za* WebSphere MQ 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 WebSphere 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 WebSphere 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.

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 217.

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 vyvolána při vytvoření kontextu uživatelské aplikace produktu WebSphere 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje, aby komponenta produktu používala produkt v libovolném definovaném způsobu.

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 s. Další informace viz “MQCSP-parametry zabezpečení” na stránce 307.

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. Na vstupu ve vstupu k funkci authenticate user, tento identifikuje aktuální kontext identity. The authenticate user function can change this, at which point the queue manager adopts the new identity context. 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 ×ComponentDataDélka-vstup/výstup

Data komponent. This data is kept by the queue manager on behalf of this particular component; any changes made to it by any of the functions provided by this component are preserved, and presented the next time one of this component's functions is called.

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.

reason
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 produktu na těchto kódů příčiny najdete v tématu Kódy příčiny.

Vyvolání jazyka C

```
MQZ_AUTHENTICATE_USER (QMgrName, SecurityParms, ApplicationContext,
                  IdentityContext, &CorrelationPtr, ComponentData,
                  &Continuation, &CompCode, &Reason);
```

Deklarujte parametry předané 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 za účelem získání informací. 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
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.

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 ×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.

reason
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 příčiny 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 za účelem získání informací. 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 1170.

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 ×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.

reason
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 příčiny 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 za účelem získání informací. 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 1170.

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: 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.

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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

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: 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 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.

reason
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 příčiny 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 za účelem získání informací. 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 ×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.

reason
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 příčiny 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 za účelem získání informací. 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 1167 .

Je-li *StartEnumeration* MQZSE_CONTINUE, *Filter* je ignorován a lze jej zadat jako ukazatel null.

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.

AuthorityDataDélka
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 ×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.

reason
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 příčiny 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 za účelem získání informací. 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 1172.

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 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.

reason
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 příčiny 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 za účelem získání informací. 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.

reason
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 příčiny 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 za účelem získání informací. 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 1170.

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.

reason
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 příčiny rozhraní API.

Syntaxe

MQZ_GET_AUTHORITY_2*(QMgrName, EntityData, EntityType, ObjectName, ObjectType, Authority, ComponentData, Continuation, CompCode, Reason)*

Vyvolání jazyka C

```
MQZ_GET_AUTHORITY_2 (QMgrName, &EntityData, EntityType, ObjectName,
                     ObjectType, &Authority, ComponentData,
                     &Continuation, &CompCode, &Reason);
```

Parametry předané službě jsou deklarovány takto:

```
MQCHAR48  QMgrName;          /* Queue manager name */
MQZED     EntityData;        /* Entity data */
MQLONG    EntityType;        /* Entity type */
MQCHAR48  ObjectName;        /* Object name */
MQLONG    ObjectType;        /* Object type */
MQLONG    Authority;         /* Authority of entity */
MQBYTE    ComponentData[n];  /* Component data */
MQLONG    Continuation;      /* Continuation indicator set by
                                component */
MQLONG    CompCode;          /* Completion code */
MQLONG    Reason;            /* Reason code qualifying CompCode */
```

MQZ_GET_EXPLICIT_AUTHORITY-Získat explicitní oprávnění

Tato funkce je poskytována komponentou služby autorizace MQZAS_VERSION_1 a je spuštěna správcem front, 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.

Na platformách UNIX pro vestavěné správce oprávnění k produktu WebSphere MQ (OAM) je vrácené 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 za účelem získání informací. 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 ×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.

reason
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 příčiny 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 za účelem získání informací. 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 1170.

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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

DélkaComponentData
Typ: MQLONG-vstup

Délka dat komponenty. Délka (v bajtech) oblasti *ComponentData* . Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE ×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
Verze 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.

reason
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 příčiny 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 za účelem získání informací. 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 ×IntAttrCount-output

Celočíselné atributy. Pole celočíselných atributů. Celočíselné atributy jsou vráceny ve stejném pořadí jako odpovídající celočíselné selektory v poli *Selectors* .

CharAttrPočet
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 ×CharAttrCount-výstup

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 ×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 ×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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné použít v libovolném definovaném způsobu.

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.

reason
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 za účelem získání informací. 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_ *.

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.

reason
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 příčiny 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 za účelem získání informací. 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 1170.

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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

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 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.

reason
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 příčiny 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 za účelem získání informací. 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 ×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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

DélkaComponentData
Typ: MQLONG-vstup

Délka dat komponenty. Délka (v bajtech) oblasti *ComponentData* . Tato délka je definována v konfiguračních datech komponenty.

ComponentData
Typ: MQBYTE ×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.

reason
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 příčiny 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 za účelem získání informací. 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.

ResolvedQMgrNázev
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.

reason
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 příčiny 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 za účelem získání informací. 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.

ResolvedQMgrNázev
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: 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-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.

reason
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 příčiny 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 za účelem získání informací. Rozhraní autorizační služby nevyžaduje komponentu, aby ji bylo možné definovat jakýmkoli způsobem.

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.

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.

reason
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 příčiny 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 596. Pole v MQZAC

Pole	**Popis**
StrucId	Identifikátor struktury
verze	Číslo verze struktury
ProcessId	Identifikátor procesu
ThreadId	Identifikátor podprocesu
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 systému UNIX toto pole uvádí skutečné ID uživatele aplikace. V systému Windows toto pole uvádí ID uživatele aplikace.

EffectiveUserID
Typ: MQCHAR12 -Vstup

Efektivní identifikátor uživatele. V systému UNIX toto pole uvádí platné 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.

- 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 597. Pole v MQZAD

Pole	Popis
StrucId	Identifikátor struktury
verze	Číslo verze struktury
ProfileName	Identifikátor procesu
ObjectType	Identifikátor podprocesu
Oprávnění	Název aplikace
EntityDataPtr	Identifikátor uživatele
EntityType	Prostředí
Volby	Typ volajícího

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.

Následující konstanta uvádí číslo verze aktuální verze:

V_AKTUÁLNÍ_VERZE MQZAD_
Aktuální verze datové struktury oprávnění.

ProfileName
Typ: MQCHAR48 -Vstup

Název profilu.

U parametru *Filtr* je toto pole názvem 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.

Pro parametr *AuthorityBuffer* je toto pole názvem profilu, který odpovídá zadaným kritériím výběru.

ObjectType
Typ: MQLONG-vstup

Typ objektu.

U parametru *Filtr* je toto pole typem objektu, pro který jsou vyžadována data oprávnění. Je-li hodnota MQOT_ALL, je vrácena data oprávnění pro všechny typy objektů.

Pro parametr *AuthorityBuffer* je toto pole typ objektu, na který se vztahuje profil určený parametrem *ProfileName* .

Hodnota je jedna z následujících možností; pro parametr *Filter* je také platná hodnota MQOT_ALL:

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 *Filtr* je toto pole ignorováno.

Pro parametr *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.

Pro parametr *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.

V případě 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 požadována data oprávnění. Je-li hodnota MQZAET_NONE, vrátí se data oprávnění pro všechny typy entit.

Pro parametr *AuthorityBuffer* toto pole uvádí typ entity, který je identifikován strukturou MQZED, na kterou ukazuje parametr *EntityDataPtr* .

Hodnota je jedna z následujících možností; pro parametr *Filter* je také platná hodnota MQZAET_NONE:

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
Zobrazte všechny profily, které se používají k výpočtu kumulativního oprávnění, které má entita k objektu uvedenému parametrem *ProfileName* . Parametr *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, hodnoty *ProfileName*, *ObjectType*, *EntityType*a název entity, které jsou zadány ve struktuře *EntityDataPtr* MQZED, musí být 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 */
};
```

Pole

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 598. 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 */
```

Pole

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.

reason
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 příčiny 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 599. 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 */
};
```

Pole

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 produktem WebSphere MQ for Windows. Identifikátor SID lze použít k doplnění pole *UserIdentifier* a k ustanovení pověření uživatele.

Tabulka 1. shrnuje pole ve struktuře.

Tabulka 600. 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 425.

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* naleznete v tématu “AccountingToken (MQBYTE32)” na stránce 386.

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 388.

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 */
};
```

Pole

Referenční materiál pro most produktu IBM WebSphere MQ pro protokol HTTP

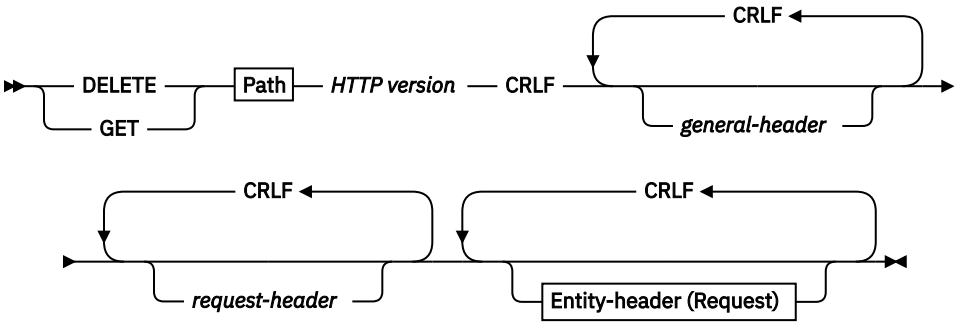
Referenční témata pro most IBM WebSphere MQ pro HTTP, která jsou uspořádána abecedně

HTTP DELETE: Most WebSphere MQ pro příkaz HTTP

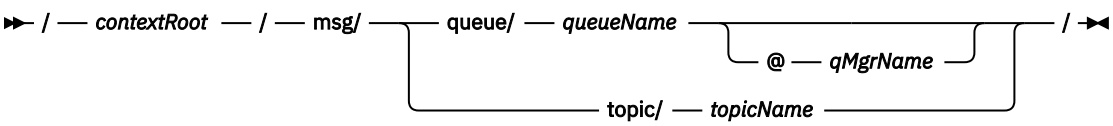
Operace HTTP **DELETE** získá zprávu z fronty WebSphere MQ , nebo načte publikování z tématu. Zpráva se odebere z fronty. Je-li publikace zachována, nebude odebrána. Zpráva odpovědi se odešle zpět klientovi včetně informací o zprávě.

Syntaxe

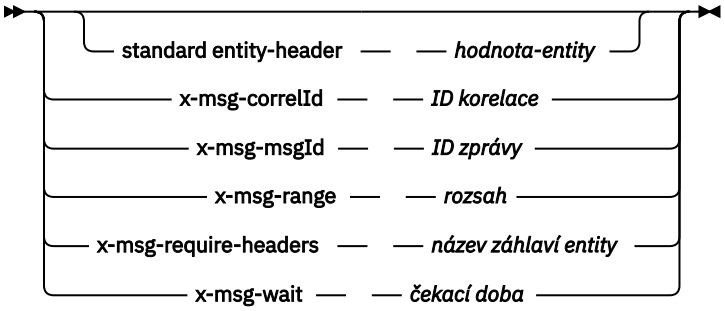
Požadavek



Path



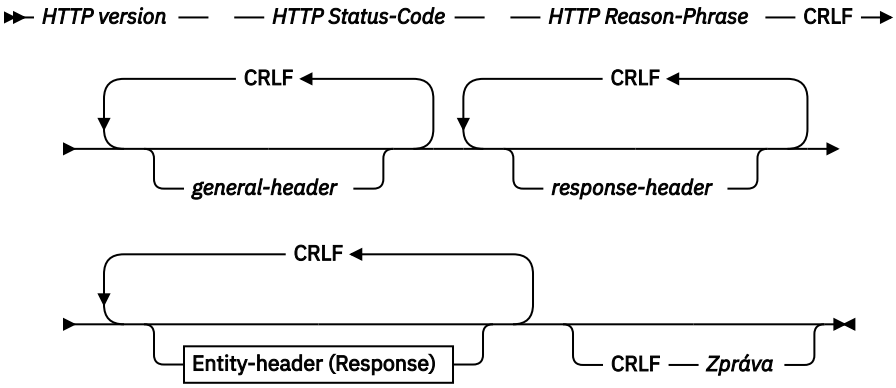
entity-záhlaví (požadavek)



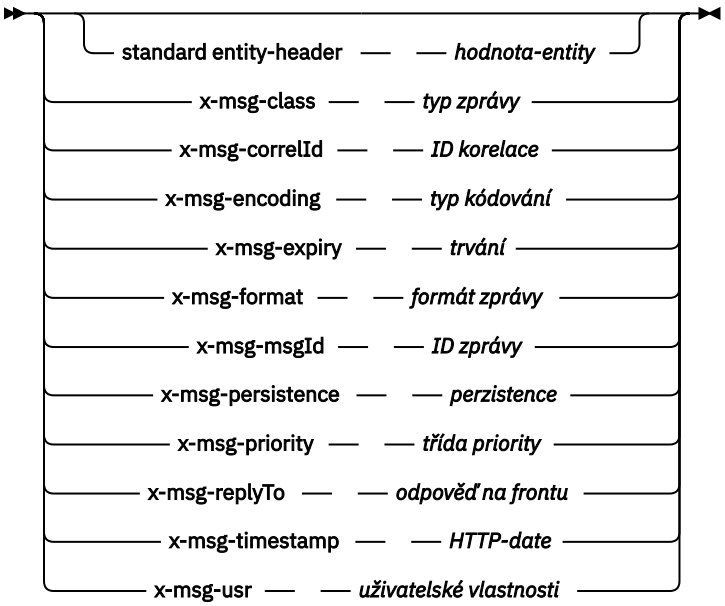
Poznámka:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

Odezva



entity-header (odezva)



Parametry požadavku

Cesta

Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

požadavek-požadavek
Viz HTTP/1.1 - 5.3 Pole záhlaví požadavku. Pole Hostitel je povinné u požadavku HTTP/1.1 . Často se automaticky vloží do nástroje, který používáte k vytvoření požadavku klienta.

entity-header (Request) (záhlaví entity)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jedno ze záhlaví entit uvedených v diagramu syntaxe požadavku.

Parametry odezvy

Cesta
Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

záhlaví odpovědi
Viz HTTP/1.1 - 6.2 Pole záhlaví odezvy.

entity-header (Odezva)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jeden z záhlaví objektů nebo odpovědí uvedených v diagramu syntaxe odezvy. Hodnota `Content-Length` je vždy přítomna v odpovědi. Je nastaven na nulu, pokud neexistuje tělo zprávy.

zpráva
Tělo zprávy.

Popis

Pokud je požadavek HTTP **DELETE** úspěšný, zpráva odezvy obsahuje data načtená z fronty produktu WebSphere MQ . Počet bajtů v těle zprávy je vrácen v záhlaví HTTP `Content-Length` . Stavový kód pro odezvu HTTP je nastaven na `200 OK`. Je-li parametr `x-msg-range` zadán jako `0`, nebo `0-0`, pak je stavový kód odpovědi HTTP `204 No Content`.

Pokud je požadavek HTTP **DELETE** neúspěšný, bude odezva zahrnovat most WebSphere MQ pro chybovou zprávu HTTP a stavový kód HTTP.

Příklad HTTP DELETE

HTTP **DELETE** získá zprávu z fronty a odstraní ji, nebo načte a odstraní publikaci. Ukázka **HTTPDELETE** Java je příkladem požadavku HTTP **DELETE**, který čte zprávu z fronty. Místo použití jazyka Java můžete vytvořit požadavek HTTP **DELETE** pomocí formuláře prohlížeče nebo sady nástrojů AJAX.

Produkt Obrázek 37 na stránce 1178 je požadavkem HTTP na odstranění další zprávy ve frontě s názvem `myQueue`. V odezvě se na klienta vrátí tělo zprávy. Ve výrazech produktu WebSphere MQ je HTTP **DELETE** destruktivním získacím.

Požadavek obsahuje záhlaví požadavku HTTP `x-msg-wait`, které instruuje produkt WebSphere MQ Bridge pro protokol HTTP, jak dlouho čekat, než zpráva dorazí do fronty. Požadavek dále obsahuje záhlaví požadavku `x-msg-require-headers`, které určuje, že klient má v odezvě přijmout ID korelace zprávy.

```
DELETE /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

*Obrázek 37. Příklad požadavku HTTP **DELETE***

Obrázek 38 na stránce 1178je odezva vrácena klientovi. ID korelace se vrátí na klienta, jak požadovalo záhlaví požadavku `x-msg-require-headers`.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that is retrieved from the queue.
```

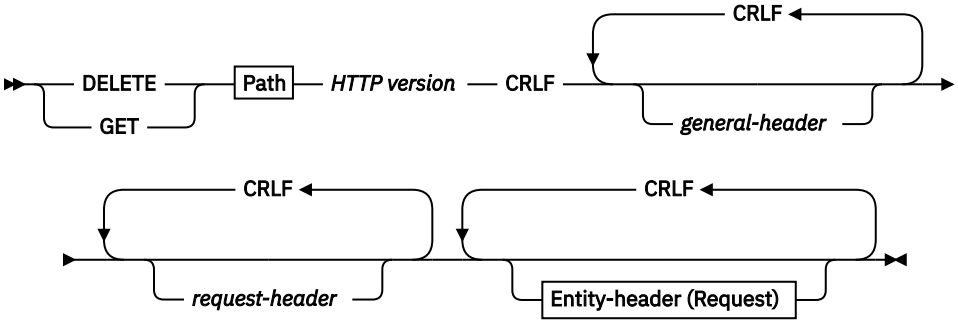
*Obrázek 38. Příklad odezvy HTTP **DELETE***

HTTP GET: Most WebSphere MQ pro příkaz HTTP

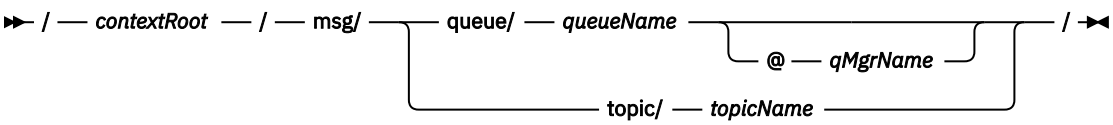
Operace HTTP **GET** získá zprávu z fronty WebSphere MQ . Zpráva bude ponechána ve frontě. Operace HTTP **GET** je ekvivalentní procházení fronty WebSphere MQ .

Syntaxe

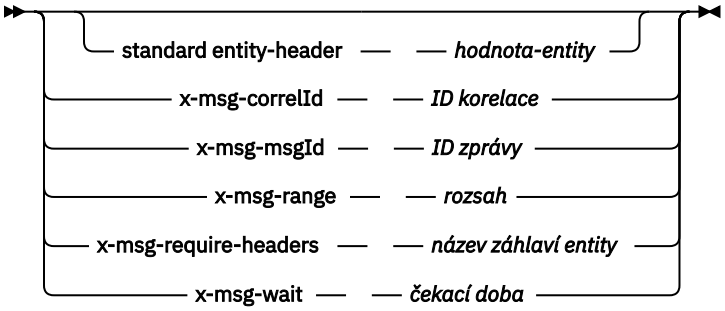
Požadavek



Path



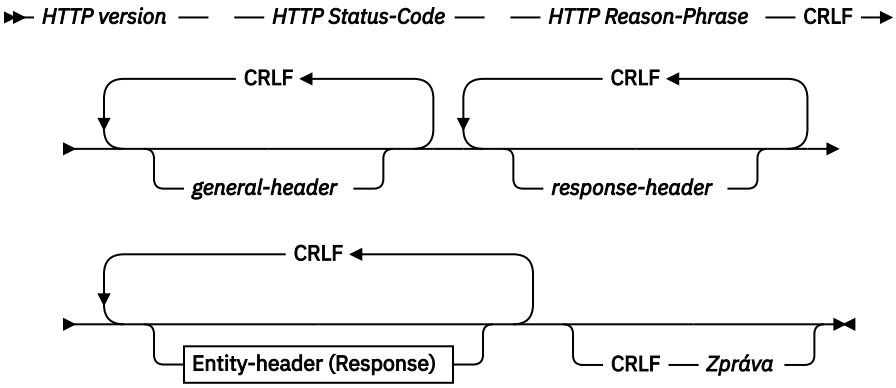
entity-záhlaví (požadavek)



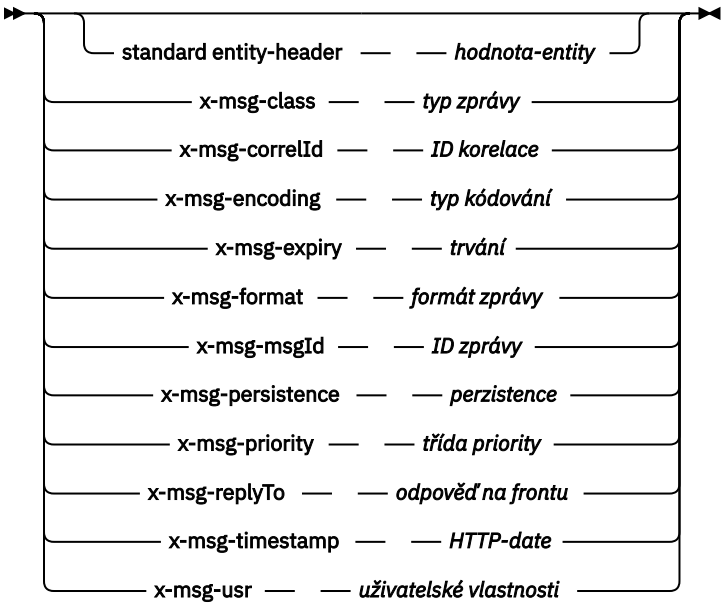
Poznámka:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

Odezva



entity-header (odezva)



Parametry požadavku

Cesta

Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

požadavek-požadavek
Viz HTTP/1.1 - 5.3 Pole záhlaví požadavku. Pole Hostitel je povinné u požadavku HTTP/1.1 . Často se automaticky vloží do nástroje, který používáte k vytvoření požadavku klienta.

entity-header (Request) (záhlaví entity)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jedno ze záhlaví entit uvedených v diagramu syntaxe požadavku.

Parametry odezvy

Cesta
Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

záhlaví odpovědi
Viz HTTP/1.1 - 6.2 Pole záhlaví odezvy.

entity-header (Odezva)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jeden z záhlaví objektů nebo odpovědí uvedených v diagramu syntaxe odezvy. Hodnota `Content-Length` je vždy přítomna v odpovědi. Je nastaven na nulu, pokud neexistuje tělo zprávy.

zpráva
Tělo zprávy.

Popis

Pokud je požadavek HTTP **GET** úspěšný, zpráva odezvy obsahuje data načtená z fronty produktu WebSphere MQ . Počet bajtů v těle zprávy je vrácen v záhlaví HTTP `Content-Length` . Stavový kód pro odezvu HTTP je nastaven na `200 OK`. Je-li parametr `x-msg-range` zadán jako `0`, nebo `0-0`, pak je stavový kód odpovědi HTTP `204 No Content`.

Pokud je požadavek HTTP **GET** neúspěšný, bude odezva zahrnovat most WebSphere MQ pro chybovou zprávu HTTP a stavový kód HTTP.

Příklad HTTP GET

HTTP **GET** získá zprávu z fronty. Zpráva ve frontě zůstane. V termínech WebSphere MQ je HTTP **GET** požadavek na procházení. Požadavek HTTP **GET** můžete vytvořit pomocí klienta Java, formuláře prohlížeče nebo sady nástrojů AJAX.

Obrázek 39 na stránce 1181 je požadavek HTTP na procházení další zprávy ve frontě s názvem `myQueue`.

Požadavek obsahuje záhlaví požadavku HTTP `x-msg-wait`, které instruuje produkt WebSphere MQ Bridge pro protokol HTTP, jak dlouho čekat, než zpráva dorazí do fronty. Požadavek dále obsahuje záhlaví požadavku `x-msg-require-headers`, které určuje, že klient má v odezvě přijmout ID korelace zprávy.

```
GET /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
x-msg-wait: 10
x-msg-require-headers: correlID
```

*Obrázek 39. Příklad požadavku HTTP **GET***

Obrázek 40 na stránce 1181 je odezva vrácená klientovi. ID korelace se vrátí na klienta, jak požadovalo záhlaví požadavku `x-msg-require-headers`.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 50
Content-Type: text/plain; charset=utf-8
x-msg-correlId: 1234567890

Here is my message body that appears on the queue.
```

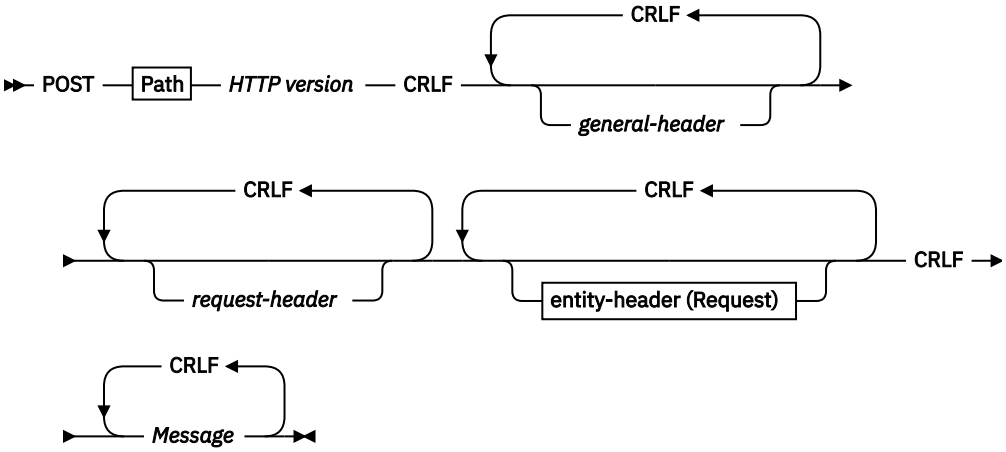
*Obrázek 40. Příklad odezvy HTTP **GET***

HTTP POST: Most WebSphere MQ pro příkaz HTTP

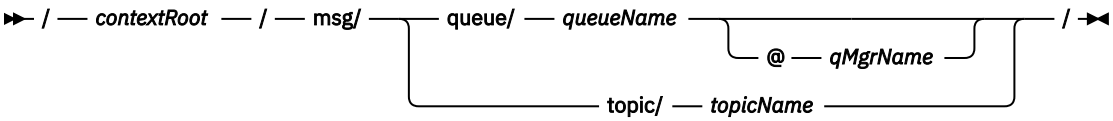
Operace HTTP **POST** vloží zprávu do fronty produktu WebSphere MQ nebo publikuje zprávu do daného tématu.

Syntax

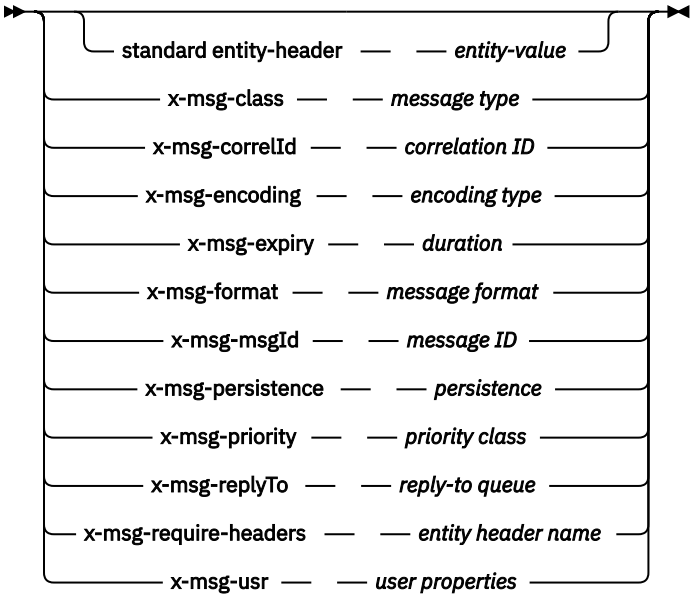
Request



Path



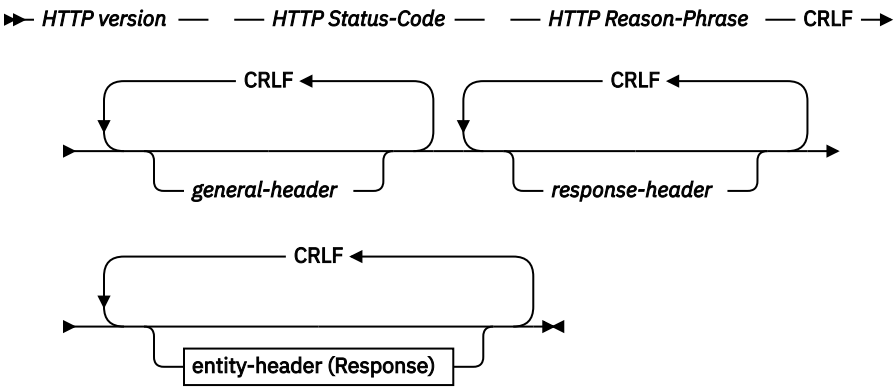
entity-header (Request)



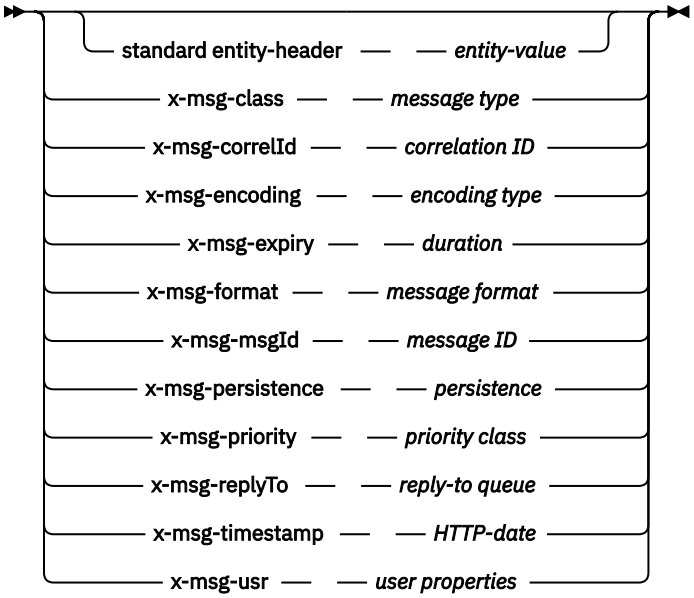
Poznámka:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

Response



entity-header (Response)



Parametry požadavku

Cesta
Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

požadavek-požadavek
Viz HTTP/1.1 - 5.3 Pole záhlaví požadavku. Pole Hostitel je povinné u požadavku HTTP/1.1 . Často se automaticky vloží do nástroje, který používáte k vytvoření požadavku klienta.

entity-header (Request) (záhlaví entity)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jedno ze záhlaví entit uvedených v diagramu syntaxe požadavku. Hodnoty `Content-Length` a `Content-Type` by měly být vloženy do požadavku a často je automaticky vkládají pomocí nástroje, který používáte k vytvoření žádosti klienta. Hodnota typu `Content-Type` se musí shodovat s typem definovaným ve vlastní záhlaví entity `x-msg-class` , je-li zadán.

zpráva
Zpráva, která má být vložena do fronty, nebo publikování na téma.

Parametry odezvy

Cesta
Viz “Formát identifikátoru URI” na stránce 1209.

Verze HTTP
Verze HTTP; například `HTTP/1.1`

general-header = záhlaví
Viz HTTP/1.1 - 4.5 Obecná pole záhlaví.

záhlaví odpovědi
Viz HTTP/1.1 - 6.2 Pole záhlaví odezvy.

entity-header (Odezva)
Viz HTTP/1.1 - 7.1 Pole záhlaví entity. Jeden z záhlaví objektů nebo odpovědí uvedených v diagramu syntaxe odezvy. Hodnota `Content-Length` je vždy přítomna v odpovědi. Je nastaven na nulu, pokud neexistuje tělo zprávy.

Popis

Pokud není zahrnuto záhlaví `x-msg-usr` a třída zprávy je BYTES nebo TEXT, zpráva zařazená do fronty nemá MQRFH2.

Použijte entitu HTTP a záhlaví požadavku v požadavku HTTP **POST** pro nastavení vlastností zprávy, která je vložena do fronty. Můžete také použít `x-msg-require-headers` k požadavku, která záhlaví se vrátí ve zprávě s odezvou.

Pokud je požadavek HTTP **POST** úspěšný, entita zprávy odpovědi je prázdná a její obsah-délka je nula. Stavový kód HTTP je `200 OK`.

Pokud je požadavek HTTP **POST** neúspěšný, bude odezva zahrnovat most WebSphere MQ pro chybovou zprávu HTTP a stavový kód HTTP. Zpráva WebSphere MQ není vložena do fronty nebo tématu.

Příklad HTTP POST

HTTP **POST** umístí zprávu do fronty nebo publikování do tématu. Ukázka **HTTPPOST** Java je příkladem požadavku HTTP **POST** zprávy na frontu. Namísto použití jazyka Java můžete vytvořit požadavek HTTP **POST** s pomocí formuláře prohlížeče nebo sady nástrojů AJAX.

Obrázek 41 na stránce 1184 ukazuje požadavek HTTP na vložení zprávy do fronty s názvem `myQueue`. Tento požadavek obsahuje záhlaví HTTP `x-msg-correlId` k nastavení ID korelace zprávy produktu WebSphere MQ .

```
POST /msg/queue/myQueue/ HTTP/1.1
Host: www.example.org
Content-Type: text/plain
x-msg-correlID: 1234567890
Content-Length: 50

Here is my message body that is posted on the queue.
```

*Obrázek 41. Příklad požadavku HTTP **POST** na frontu*

Obrázek 42 na stránce 1184 zobrazuje odezvu odeslanou zpět klientovi. Není žádný obsah odezvy.

```
HTTP/1.1 200 OK
Date: Wed, 2 Jan 2007 22:38:34 GMT
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
Content-Length: 0
```

Obrázek 42. Příklad odezvy HTTP POST

Záhlaví HTTP

Most WebSphere MQ pro protokol HTTP podporuje záhlaví HTTP vlastních požadavků, záhlaví HTTP vlastních entit a podmnožinu standardních záhlaví HTTP.

Postup HTTP je předpona všech vlastních záhlaví s produktem `x-`, produkt WebSphere MQ Bridge pro záhlaví HTTP má předponu `x-msg-`. Chcete-li například nastavit použití záhlaví priority `x-msg-priority`.

Poznámka:

- Většina hodnot záhlaví rozlišuje velikost písmen. Například, když používáte záhlaví `msgId` , `NONE` je klíčové slovo, zatímco `none` je `msgID`.
- Špatně zadaná záhlaví jsou ignorována.

Vlastní záhlaví HTTP entity

Vlastní záhlaví HTTP entity obsahují informace o zprávách produktu WebSphere MQ . Pomocí záhlaví entit můžete nastavit hodnoty v deskriptoru zpráv (`MQMD`) nebo hodnoty dotazu v produktu `MQMD`. Další záhlaví entity, `x-msg-usr`, nastaví a vrátí všechny informace o vlastnostech uživatele, které chcete přidružit k požadavku.

Záhlaví entit můžete používat v různých kontextech požadavků HTTP:

DELETE
: Záhlaví entit lze použít pouze pro záhlaví `x-msg-correlId`nebo `x-msg-msgId`, případně obojí, záhlaví HTTP požadavku **DELETE** . Výsledkem záhlaví je výběr konkrétní zprávy `MsgId` a `CorrelId` v `MQGET`a odstranění zprávy z její fronty.

GET
: Záhlaví entit lze použít pouze pro záhlaví `x-msg-correlId`nebo `x-msg-msgId`, případně obojí, záhlaví HTTP požadavku **GET** . Výsledkem záhlaví je výběr konkrétní zprávy podle hodnot `MsgId` a `CorrelId` v `MQGET` pro procházení.

POST
: V požadavku **POST** HTTP můžete použít libovolné záhlaví entity, kromě `x-msg-timestamp`.

x-msg-require-headers
: V libovolném požadavku HTTP **GET**, **POST** nebo **DELETE** můžete přidat více záhlaví entit do záhlaví požadavku `x-msg-require-headers` , oddělené čárkami. Výsledkem je vrácení uvedených záhlaví entit ve zprávě odpovědi HTTP obsahující hodnotu přidružené vlastnosti zprávy.

Popis jednotlivých seznamů záhlaví, v nichž jsou záhlaví zpracovány záhlaví produktu WebSphere MQ pro protokol HTTP. Například v záhlaví **POST**, `x-msg-require-headers`je záhlaví zpracováno pomocí mostu WebSphere MQ pro protokol HTTP v požadavku HTTP **POST** nebo v záhlaví požadavku `x-msg-require-headers` buď v požadavku HTTP **POST**, **GET**nebo **DELETE** . Je-li záhlaví obsaženo v kontextu, v němž není povoleno, záhlaví se ignoruje. Nehlásí se žádná chyba.

Můžete vložit libovolné standardní záhlaví HTTP do požadavků, které budou zpracovány webovým serverem, nebo jiné obslužné rutiny požadavků. Podobně může odezva obsahovat jiná standardní záhlaví HTTP vložená webovým serverem nebo jinými obslužnými rutinami odpovědí.

Vlastní záhlaví HTTP požadavku

Tři vlastní záhlaví HTTP požadavku, `x-msg-range`, `x-msg-require-headers`a `x-msg-wait`, předávají další informace o požadavku HTTP na server. Pracují jako modifikátory požadavků. Pomocí parametru `x-msg-range`můžete omezit množství dat zpráv vrácených v rámci odezvy. Pomocí `x-msg-require-headers`můžete požádat o odpověď, aby obsahovala informace o výsledku požadavku. Při použití volby `x-msg-wait`můžete upravit dobu, po kterou klient čeká na odpověď HTTP.

Standardní záhlaví HTTP

Ve požadavku HTTP/1.1 musí být uvedeno standardní záhlaví požadavku HTTP `Host` .

Standardní záhlaví entit HTTP `Content-Length` a `Content-Type` lze v požadavku zadat.

Záhlaví `Content-Length`, `Content-Location`, `Content-Range`, `Content-Type`a `Server` standard HTTP mohou být vrácena jako odezva na požadavek. Uveďte jedno nebo více ze standardních záhlaví HTTP v záhlaví `x-msg-request-header` ve zprávě požadavku.

Abecední seznam hlaviček HTTP

`class`: entita HTTP `x-msg-class` -záhlaví

Nastavit nebo vrátit typ zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpr-třída`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	POST, `x-msg-require-headers`
Povolené hodnoty	**BYTES** **MAP** **STREAM** **TEXT**
Výchozí hodnota	BYTES

Popis

- V požadavku HTTP **POST** nastaví typ vytvořené zprávy.
- Uvedení hlavičky třídy na **GET** nebo **DELETE** vrátí `400 Bad Request` s tělem entity `MQHTTP40007`.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-class` ve zprávě odpovědi HTTP na typ zprávy.
- Je-li pro toto záhlaví zadána neplatná hodnota, bude vrácena zpráva `MQHTTP40005` .
- Není-li záhlaví `x-msg-class` uvedeno a typ obsahu zprávy je `application/x-www-form-urlencoded`, předpokládá se, že se jedná o objekt mapy JMS.

`Content-Length`: Entita HTTP-záhlaví

Nastavte nebo vraťte délku těla zprávy v bajtech.

Typ	Popis
Název záhlaví HTTP	`Délka obsahu`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	`x-msg-require-headers`
Povolená a vrácená hodnota	***Integer value*** Délka těla zprávy v bajtech.

Popis

- Hodnota `Content-Length` je v požadavku HTTP volitelná. Pro **GET** nebo **DELETE** musí být délka nula. Pokud je zadán parametr **POST**, je-li zadán parametr `Content-Length` a neodpovídá délce řádku zprávy, je zpráva oříznuta nebo doplněna hodnotami null na zadanou délku.
- Hodnota `Content-Length` je vždy vrácena v odpovědi HTTP i v případě, že neexistuje žádný obsah. V takovém případě je hodnota nula.

`Content-Location`: Entita HTTP-záhlaví

Vrací frontu nebo téma, na které se odkazuje požadavek, ve standardním záhlaví `Content-Location` ve zprávě odpovědi HTTP.

Typ	Popis
Název záhlaví HTTP	`Content-Location`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	`x-msg-require-headers`

Typ	Popis

Návratová hodnota	Identifikátor URI ve formátu,
	`/msg/queue/queuename`
	, nebo
	`/msg/topic/topicname`

Popis

- Při požadavku v záhlaví `x-msg-require-headers`vrací záhlaví entity `Content-Location` frontu nebo téma, na které se odkazuje v požadavku HTTP.

`Content-Range`: Entita HTTP-záhlaví

Vrátí rozsah bajtů vybraných ze zprávy produktu WebSphere MQ v záhlaví `Content-Range` v odezvě HTTP.

Typ	Popis
Název záhlaví HTTP	`Obsah-rozsah`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	`x-msg-require-headers`
Návratová hodnota	***`String`*** Vrací dolní limit, *`m`* a horní limit, *`n`* vráceného podřetězce a *`length`* celé zprávy. Například `m-n/length`

Popis

-
- Hodnota `Content-Range` je vrácena v odezvě HTTP pouze v případě, že je v požadavku **GET** nebo **DELETE** zadán parametr `Content-Range` , který obsahuje záhlaví požadavku `x-msg-range` .
- Je-li parametr `x-msg-range` zadán v požadavku **GET** nebo **DELETE** , v odezvě se vrátí rozsah bajtů zadaný v záhlaví Content-Range. Je-li například `x-msg-range: 0-60` použit v požadavku na zprávu obsahující `100` bajtů, záhlaví rozsahu obsahu uchovává řetězec `0-60/100`
- Požadavek `x-msg-range` také vrátí rozsah obsahu v záhlaví `x-msg-range` v odpovědi HTTP.

`Content-Type`: Entita HTTP-záhlaví

Nastavte nebo vraťte třídu zprávy JMS ve zprávě produktu WebSphere MQ podle typu obsahu HTTP.

Typ	Popis
Název záhlaví HTTP	`Content-Type`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`
Povolená nebo vrácená hodnota	***`media-type`*** Pro typy médií, které jsou podporovány, viz Tabulka 601 na stránce 1188.

Tabulka 601. Mapování mezi `x-msg-class` a HTTP `Content-Type`	
x-zpr-třída	**HTTP Content-Type**
BYTES	`application/octet-stream` `application/xml`
TEXT	`text/*`
MAP	`application/x-www-form-urlencoded` `application/xml` (volitelné)
STREAM	`application/xml` (volitelné)

Popis

- V případě požadavku HTTP **POST** zadejte buď `Content-Type` , nebo `x-msg-class`. Pokud uvedete obě, musí být konzistentní, nebo bude vrácena výjimka HTTP `Bad Request` , je vrácena hodnota `Status code 400` . Vynecháte-li oba typy `Content-Type` i `x-msg-class`, předpokládá se `Content-Type` z `text/*` .
- Typ `Content-Type` je vždy nastaven v odezvě na HTTP **GET** nebo **DELETE** , který má tělo zprávy. Parametr `Content-Type` je nastaven podle pravidel v produktu Tabulka 602 na stránce 1188.

Tabulka 602. Mapování typů zpráv na `x-msg-class` a `Content-Type`			
Formát zprávy	**Typ zprávy platformy JMS**	**x-zpr-třída**	**Content-Type**
Vše kromě `MQFMT_STRING`	Není	BYTES	`application/octet-stream`
`MQFMT_STRING`	Není	TEXT	`text/plain`
`MQFMT_NONE`	`jms_bytes`	BYTES	`application/octet-stream`
`MQFMT_NONE`	`jms_text`	TEXT	`text/plain`
`MQFMT_NONE`	`jms_map`	MAP	`application/xml`
`MQFMT_NONE`	`jms_stream`	STREAM	`application/xml`

`correlId`: HTTP `x-msg-correlId` záhlaví entity

Nastavte nebo vraťte identifikátor korelace.

Typ	**Popis**
Název záhlaví HTTP	`x-msg-correlId`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**DELETE**, **GET**, **POST**, `x-msg-require-headers`
Povolené hodnoty	***`String value`*** Příklad: `x-msg-correlId: mycorrelationid`

Typ	Popis

	Řetězce uzavřené v uvozovkách jsou povoleny; například:
	`x-msg-correlId: "my id"`
	Hex value
	Hexadecimální hodnota s předponou `0x:`; například:
	`x-msg-correlId: 0x:43c1d23a`
	Hexadecimální hodnota po 0x: je omezena na 48 znaků představujících 24 bajtů. Další data jsou ignorována.

Výchozí hodnota	Nelze použít

Popis

- V případě požadavku HTTP **POST** nastavuje ID korelace vytvářené zprávy.
- Na požadavku HTTP **GET** nebo **DELETE** vyberte zprávu z fronty nebo tématu. Pokud neexistuje žádná zpráva s uvedeným ID korelace, je vrácena odezva `HTTP 504 Gateway Timeout`. `x-msg-correlId` lze použít s `x-msg-msgID` pro výběr zprávy z fronty nebo tématu, které se shoduje s oběma selektory.
- Uvedeno v `x-msg-require-headers`, nastavuje `x-msg-corelId` ve zprávě odpovědi HTTP na ID korelace zprávy.
- Vodorovný bílý znak je povolen za předponou `0x:` .

Poznámka:

- Zadání hodnoty `x-msg-correlId` bez hodnoty v požadavku HTTP **GET** nebo **DELETE** , například `"x-msg-correlId:"`, vrací další zprávu ve frontě nebo tématu bez ohledu na její ID korelace.
- Pokud uvedete selektor 24 nebo méně znaků nebo 0x: následováno 48 znaků nebo méně, most WebSphere MQ pro HTTP používá optimalizovaný selektor pro zlepšení výkonu.
- Selektor zpráv JMS obsahující JMSCorrelationID se používá při výběru zpráv z fronty. Tento selektor se chová způsobem popsaným v části Chování výběru.

encoding: HTTP `x-msg-encoding` entita-záhlaví

Nastavit nebo vrátit kódování zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpr-kódování`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`
Povolené hodnoty	Čárkami oddělený seznam následujících hodnot: **DECIMAL_NORMAL** **DECIMAL_REVERSED** **FLOAT_IEEE_NORMAL** **FLOAT_IEEE_REVERSED** **FLOAT_S390** **INTEGER_NORMAL** **INTEGER_REVERSED**

Typ	Popis
	Například `x-msg-encoding: INTEGER_NORMAL,DECIMAL_NORMAL,FLOAT_IEEE_NORMAL` **Poznámka:** Hodnota je citlivá na velikost písmen
Výchozí hodnota	`DECIMAL_NORMAL, FLOAT_IEEE_NORMAL, INTEGER_NORMAL`

Popis

- V požadavku HTTP **POST** určuje kódování vytvořené zprávy.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-encoding` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastavuje `x-msg-encoding` ve zprávě odpovědi HTTP na vlastnost kódování zprávy.

`expiry`: HTTP `x-msg-expiry` entity-header

Nastavte nebo vraťte dobu trvání vypršení platnosti zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpr-zpr`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	**`UNLIMITED`**
	Například;
	`x-msg-expiry: UNLIMITED`
	`Integer value`
	Milisekundy před vypršením.
	Například;
	`x-msg-expiry: 10000`

Výchozí hodnota	`UNLIMITED`

Popis

- Je-li nastaven na požadavek HTTP **POST** , platnost zprávy požadavku vyprší v uvedeném čase.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-expiry` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-expiry` ve zprávě odpovědi HTTP do doby vypršení platnosti zprávy.
- `UNLIMITED` uvádí, že zpráva nikdy nevyprší platnost.
- Ukončení platnosti zprávy začíná od okamžiku, kdy zpráva dorazí do fronty, protože latence sítě výsledků je ignorována.
- Maximální hodnota je omezena hodnotou WebSphere MQ až 214748364700 milisekund. Je-li hodnota větší než uvedená hodnota, předpokládá se maximální možná doba vypršení platnosti.

`format`: HTTP `x-msg-format` entity-header

Nastavte nebo vraťte formát zprávy produktu WebSphere MQ .

Typ	Popis
Název záhlaví HTTP	`x-msg-format`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	**NONE**
	Například
	`x-msg-format: NONE`
	String value
	Libovolná uživatelem definovaná hodnota až osmi znaků.
	Například
	`x-msg-format: myformat`

Výchozí hodnota	None

Popis

- Je-li nastaven na požadavek HTTP **POST** , nastavte formát zprávy požadavku.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-format` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastaví formát `x-msg-format` ve zprávě odpovědi HTTP na formát zprávy.
- `NONE` rozlišuje velká a malá písmena a indikuje, že formát zprávy je prázdný.
- Je použita hodnota `x-msg-format` , i když je v rozporu s typem média požadavku HTTP. Viz Tabulka 603 na stránce 1191.

Tabulka 603. Mapování typu content-type a `x-msg-class` do formátu zprávy

`x-zpr-třída`	**Content-Type**	**Formát zpráv ve frontě/tématu**
BYTES	• `application/octet-stream` • application/xml	Zpráva WebSphere MQ : `MQFMT` nastavena na `MQC.MQFMT_NONE`
TEXT	• `text/*`	Zpráva WebSphere MQ : `MQFMT` nastavena na `MQC.MQFMT_STRING`
MAP	• `application/x-www-form-urlencoded` • `application/xml` (volitelné)	`JMSMap`
STREAM	• `application/xml` (volitelné)	`JMSStream`

msgId: entita HTTP `x-msg-msgId` -záhlaví

Nastavte nebo vraťte identifikátor zprávy.

Typ	Popis
Název záhlaví HTTP	`x-msg-msgId`
Typ záhlaví HTTP	Záhlaví subjektu

Typ	Popis
Platné ve zprávě požadavku HTTP	**DELETE**, **GET**, **POST**, `x-msg-require-headers`

Povolené hodnoty	***String value***
	Například
	`x-msg-msgId: mymsgid`
	Řetězce uzavřené v uvozovkách, například, `x-msg-msgId: "my id"`
	Hex value
	Hexadecimální hodnota s předponou `0x:`; například:
	`x-msg-msgId: 0x:43c1d23a`

Výchozí hodnota	Nelze použít

Popis

- V případě požadavku HTTP **POST** nastavuje ID zprávy vytvořené zprávy.
- Na požadavku HTTP **GET** nebo **DELETE** vyberte zprávu z fronty nebo tématu. Pokud neexistuje žádná zpráva s uvedeným ID zprávy, je vrácena odezva `HTTP 504 Gateway Timeout`. `x-msg-msgId` lze použít spolu s parametrem `x-msg-correlID` pro výběr zprávy z fronty nebo tématu, které odpovídá oběma selektorům.
- Uvedeno v `x-msg-require-headers`, vrací `x-msg-msgId` do odpovědi HTTP na ID zprávy zprávy.
- Vodorovný bílý znak je povolen za předponou `0x:`.

Poznámka: Určením parametru `x-msg-msgId` bez hodnoty v požadavku HTTP **GET** nebo **DELETE** , například `"x-msg-msgId:"`, vrátí další zprávu ve frontě nebo tématu bez ohledu na její ID zprávy.

Při výběru zpráv z fronty se používá selektor zpráv JMS obsahující JMSMessageID . Tento selektor se chová způsobem popsaným v části Chování výběru .

`persistence`: HTTP `x-msg-persistence` entita-header

Nastavte nebo vraťte trvání zprávy.

Typ	Popis
Název záhlaví HTTP	`x-msg-persistence`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`
Povolené hodnoty	**NON_PERSISTENT** Zpráva nepřežije selhání systému nebo správce front se restartuje. Například `x-msg-persistence: NON_PERSISTENT` **PERSISTENT** Zpráva přežije selhání systému a restartuje správce front. Například

Typ	Popis
	`x-msg-persistence: PERSISTENT` **AS_DESTINATION** Vztahuje se pouze na **POST** . Použijte výchozí trvání cíle, jak je určeno poskytovatelem zpráv. **Poznámka:** Rozlišovat malá a velká písmena
Výchozí hodnota	NON_PERSISTENT

Popis

- Je-li nastaven na požadavek HTTP **POST** , nastavte perzistenci zprávy požadavku.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-persistence` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastavuje `x-msg-persistence` ve zprávě odpovědi HTTP na perzistenci zprávy.

`priority`: HTTP `x-msg-priority` entity-záhlaví

Nastavit nebo vrátit prioritu zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpr-priorita`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	**LOW**
	Například
	`x-msg-priority: LOW`
	MEDIUM
	Tato priorita je rovna úrovni priority WebSphere MQ na úrovni 4. Například
	`x-msg-priority: MEDIUM`
	HIGH
	Například
	`x-msg-priority: HIGH`
	Integer value
	řetězcová reprezentace celého čísla v rozsahu 0 až 9; například,
	`x-msg-priority: 3`
	AS_DESTINATION
	Vztahuje se pouze na **POST** .
	Použijte výchozí prioritu cíle, jak je určeno poskytovatelem zpráv.
	Poznámka: Rozlišovat malá a velká písmena

Výchozí hodnota	MEDIUM

Popis

- Je-li nastaven na požadavek HTTP **POST** , nastavte prioritu zprávy požadavku.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-priority` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-priority` ve zprávě odpovědi HTTP na prioritu zprávy.

`priority`: HTTP `x-msg-priority` entity-záhlaví

Nastavit nebo vrátit prioritu zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpr-priorita`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	**LOW**
	Například
	`x-msg-priority: LOW`
	MEDIUM
	Tato priorita je rovna úrovni priority WebSphere MQ na úrovni 4. Například
	`x-msg-priority: MEDIUM`
	HIGH
	Například
	`x-msg-priority: HIGH`
	Integer value
	řetězcová reprezentace celého čísla v rozsahu 0 až 9; například,
	`x-msg-priority: 3`
	AS_DESTINATION
	Vztahuje se pouze na **POST** .
	Použijte výchozí prioritu cíle, jak je určeno poskytovatelem zpráv.
	Poznámka: Rozlišovat malá a velká písmena

Výchozí hodnota	MEDIUM

Popis

- Je-li nastaven na požadavek HTTP **POST** , nastavte prioritu zprávy požadavku.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-priority` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-priority` ve zprávě odpovědi HTTP na prioritu zprávy.

`replyTo`: entita HTTP `x-msg-replyTo` -záhlaví

Nastavte nebo vraťte název odpovědi na zprávu-do fronty a správce front.

Typ	Popis
Název záhlaví HTTP	`x-msg-replyTo`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	***URI***
	Identifikátor URI dvoubodového spojení; například,
	`x-msg-replyTo: /msg/queue/myReplyQueue`
	`x-msg-replyTo: /msg/queue/myReplyQueue@myReplyQueueManager`
	Poznámka: Rozlišovat malá a velká písmena

Výchozí hodnota	`MEDIUM`

Popis

- Je-li nastaven na požadavek HTTP **POST** , nastavte cíl zprávy replyTo požadavku.
- Na požadavku HTTP **GET** nebo **DELETE** se záhlaví `x-msg-replyTo` ignoruje.
- Uvedeno v `x-msg-require-headers`, nastavuje `x-msg-replyTo` ve zprávě odpovědi HTTP na frontu odpovědí a správce front ve zprávě .

Poznámka: Identifikátor URI produktu v odpovědi HTTP může obsahovat název správce front, ke kterému je připojen most produktu WebSphere MQ pro protokol HTTP.

`Server`: záhlaví odpovědi HTTP

Vrací informace o serveru a protokolu, ke kterému je klient připojen.

Typ	Popis
Název záhlaví HTTP	SERVER
Typ záhlaví HTTP	Záhlaví odezvy
Platné ve zprávě požadavku HTTP	`x-msg-require-headers`

Návratová hodnota	`WMQ-HTTP/1.1 JEE-Bridge/1.1`
	, nebo
	`Server: Product-token WMQ-HTTP/1.1 JEE-Bridge/1.1`

Popis

- Je-li produkt WebSphere MQ Bridge pro protokol HTTP implementován na aplikační server, je k záhlaví odezvy serveru připojen most produktu WebSphere MQ pro podrobnosti protokolu HTTP. Například most WebSphere MQ pro protokol HTTP implementovaný do produktu WebSphere Application Server Community Edition, který se nazývá Apache-Coyote, poskytuje odpověď:

```
Server: Apache-Coyote/1.1 WMQ-HTTP/1.1 JEE-Bridge/1.1
```

`require-headers`: záhlaví požadavku HTTP `x-msg-require-headers`

Nastavte, která záhlaví se mají vrátit ve zprávě odpovědi HTTP.

Typ	Popis
Název záhlaví HTTP	`x-msg-require-headers`
Typ záhlaví HTTP	Hlavička požadavku
Platné ve zprávě požadavku HTTP	**POST**, **GET**, **DELETE**

Povolené hodnoty	Čárkami oddělený seznam názvů záhlaví entity:
	ALL
	ALL-USR
	class
	content-location
	correlId
	encoding
	expiry
	format
	msgId
	NO_require-headers
	persistence
	priority
	replyTo
	server
	timestamp
	usr-*property name*
	Například
	`x-msg-require-headers: msgId`
	nebo
	`x-msg-require-headers: expiry,correlId,timestamp`
	Žádost o specifickou vlastnost:
	`x-msg-require-headers: usr-myCustomProperty`
	Požadavek na všechny vlastnosti:
	`x-msg-require-headers: ALL-USR, ALL`

Výchozí hodnota	`NO_require-headers`

Popis

- Hodnota `x-msg-require-headers` nerozlišuje velká a malá písmena, s výjimkou případů `ALL`, `NO_require-headers`a `ALL-USR` konstant a *`property-name`* proměnných.

`timestamp`: entita HTTP `x-msg-timestamp` -záhlaví

Vraťte časové razítko zprávy.

Typ	Popis
Název záhlaví HTTP	`x-zpráva-timestamp`

Typ	Popis
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	`x-msg-require-headers`
Návratová hodnota	***HTTP-date*** Datum ve formátu; den, datum a čas měsíce, rok-zone; například, `Sun, 06 Nov 1994 08:49:37 GMT` Definováno produktem RFC 822a aktualizováno v produktu RFC 1123.
Výchozí hodnota	Nelze použít

Popis

- Na základě požadavku HTTP **POST**, **GET** nebo **DELETE** se záhlaví `x-msg-timestamp` ignoruje.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-timestamp` ve zprávě odpovědi HTTP na časové razítko zprávy.

`usr`: HTTP `x-msg-usr` -záhlaví entity

Nastavte nebo vraťte uživatelské vlastnosti.

Typ	Popis
Název záhlaví HTTP	`x-zpráv-usr`
Typ záhlaví HTTP	Záhlaví subjektu
Platné ve zprávě požadavku HTTP	**POST**, `x-msg-require-headers`

Povolené hodnoty	Viz “Syntaxe” na stránce 1198; například:
	`x-msg-usr: myProp1;5;i1,`
	`x-msg-usr: myProp2;"My String";string`

Výchozí hodnota	Nelze použít

Popis

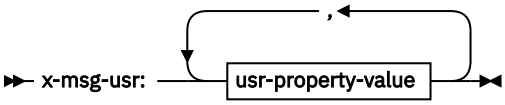
- Je-li nastaven na požadavek HTTP **POST** , nastavte uživatelské vlastnosti zprávy požadavku.
- Na základě požadavku HTTP **GET** nebo **DELETE** je záhlaví `x-msg-usr` ignorováno.
- Uvedeno v `x-msg-require-headers`, nastaví `x-msg-usr` ve zprávě odpovědi HTTP na uživatelské vlastnosti zprávy.
- Na zprávě může být nastaveno více vlastností. Do jednoho záhlaví `x-msg-usr` zadejte více vlastností oddělených čárkami nebo použijte dvě nebo více samostatných instancí záhlaví `x-msg-usr` .
- V odpovědi na požadavek **GET** nebo **DELETE** můžete požádat o vrácení specifické vlastnosti. Zadejte název vlastnosti v záhlaví požadavku `x-msg-require-headers` s použitím předpony `usr-`. Například

```
x-msg-require-headers: usr-myProp1
```

- Chcete-li požádat o vrácení všech uživatelských vlastností v odezvě, použijte konstantu `ALL-USR` . Například

```
x-msg-require-headers: ALL-USR
```

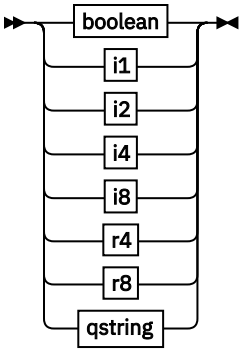
Syntaxe



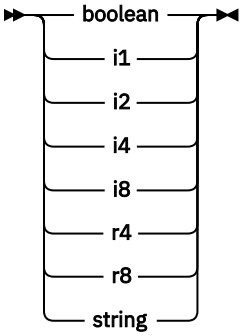
usr-property-value

property-name

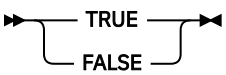
usr-value



usr-type

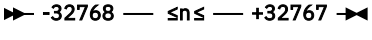


boolean

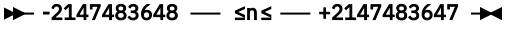


i1

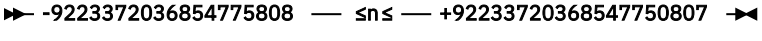
i2



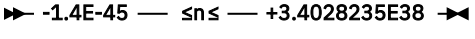
i4



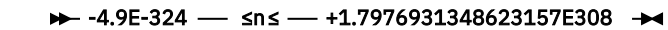
i8



r4



r8



qstring

wait: Požadavek HTTP x-msg-wait -záhlaví

Nastavte dobu čekání na příchod zprávy, než bude vrácena zpráva odpovědi `HTTP 504 Gateway Timeout` .

Typ	Popis
Název záhlaví HTTP	`x-zpráva-wait`
Typ záhlaví HTTP	Hlavička požadavku
Platné ve zprávě požadavku HTTP	**GET**, **DELETE**

Povolená hodnota	**NO_WAIT**
	Například
	`x-msg-wait: NO_WAIT`
	Integer value
	doba v milisekundách, po kterou most produktu WebSphere MQ pro protokol HTTP čeká na doručení zprávy, například:
	`x-msg-wait: 8`

Výchozí hodnota	`NO_WAIT`

Popis

- Na základě požadavku HTTP **POST** je záhlaví `x-msg-wait` ignorováno.
- V požadavku HTTP **GET** nebo **DELETE** určuje parametr `x-msg-wait` dobu čekání na příchod zprávy před vrácením odezvy produktu `HTTP 504 Gateway Timeout` .
- `NO_WAIT` rozlišuje velká a malá písmena.
- Předvolená maximální čekací doba je 35000. Výchozí nastavení lze změnit nastavením parametru `maximum_wait_time` servletu. Další informace naleznete v části Instalace, konfigurace a ověření mostu produktu WebSphere MQ pro protokol HTTP .
- Nastavíte-li hodnotu větší než `maximum_wait_time`, použije se místo toho `maximum_wait_time` .

návratové kódy HTTP

Seznam návratových kódů z mostu produktu WebSphere MQ pro protokol HTTP

Most WebSphere MQ pro protokol HTTP vrací čtyři typy chyb:

Chyby servletu
MQHTTP0001 a MQHTTP0002 jsou chyby servletu. Jsou protokolovány, ale nejsou vráceny klientovi HTTP.

Úspěšné operace
Stavový kód HTTP v rozsahu 200-299 označuje úspěšnou operaci.

Chyby klienta
Stavový kód HTTP v rozsahu 400-499 označuje chybu klienta. Produkt WebSphere MQ Bridge pro návratové kódy HTTP v rozsahu MQHTTP40001 - MQHTTP49999 odpovídá chybám klienta.

Chyby serveru

Stavový kód HTTP v rozsahu 500-599 označuje chybu klienta. Produkt WebSphere MQ Bridge pro návratové kódy HTTP v rozsahu MQHTTP50001 - MQHTTP59999 odpovídá chybám serveru.

Pokud dojde k chybě serveru, výstupem protokolu chyb aplikačního serveru je úplný výstup trasování zásobníku. Trasování zásobníku se vrátí také klientovi HTTP v odpovědi HTTP. Zacházejte s trasováním zásobníku v aplikaci klienta nebo ji odkažte na administrátora aplikačního serveru, aby problém vyřešil.

Pokud trasování zásobníku obsahuje chyby adaptéru prostředků, podívejte se do dokumentace pro váš adaptér prostředků.

Abecední seznam návratových kódů

HTTP 200: OK

Tato třída stavového kódu označuje, že požadavek byl úspěšně přijat, chápán a přijat.

Stavový kód HTTP
`200 OK`

HTTP 204: Chybí obsah

Odesláno za úspěšným HTTP **GET** nebo **DELETE** a `x-msg-range: 0` bylo odesláno v požadavku.

Stavový kód HTTP
`204 No Content`

MQHTTP0001: V kontextu servletu není určena faktorie připojení.

Chyba servletu

Vysvětlení
Chyba servletu

Stavový kód HTTP
Není

Odpověď programátora
Kde jsou tyto chyby protokolovány, jsou specifické pro váš aplikační server. Další informace naleznete v dokumentaci k aplikačnímu serveru.

MQHTTP0002: Nelze získat správce připojení pro *queueOrTopic* s použitím názvu rozhraní JNDI *jndiNameTried*

Chyba servletu

Vysvětlení
Chyba servletu

Stavový kód HTTP
Není

Odpověď programátora
Kde jsou tyto chyby protokolovány, jsou specifické pro váš aplikační server. Další informace naleznete v dokumentaci k aplikačnímu serveru.

MQHTTP40001: Rezervováno

Vyhrazené

Stavový kód HTTP
`400 Bad Request`

Identifikátor URI MQHTTP40002: není platný pro transport WebSphere MQ pro protokol HTTP

Identifikátor URI uvedený v požadavku HTTP je neplatný.

Vysvětlení
Identifikátor URI uvedený v požadavku HTTP je neplatný.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Potvrďte, že formát a syntaxe uvedeného identifikátoru URI jsou správné.

MQHTTP40003: Identifikátor URI je neplatný. `@qmgr` je platný pouze na POST

Volba identifikátoru URI produktu `@qmgr` byla určena v identifikátoru URI pro požadavek HTTP, který není požadavkem **POST** .

Vysvětlení
Volba identifikátoru URI produktu `@qmgr` byla určena v identifikátoru URI pro požadavek HTTP, který není požadavkem **POST** .

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Pokoušíte-li se vložit zprávu pomocí příkazového slova **POST** , změňte požadavek HTTP na požadavek **POST** . Pokoušíte-li se získat zprávu pomocí příkazových slov **DELETE** nebo **GET** , odeberte `@qmgr` z identifikátoru URI.

MQHTTP40004: Byl zadán neplatný `Content-Type` .

Pole záhlaví `Content-Type` zadané v požadavku **POST** není kompatibilní s hodnotou záhlaví `x-msg-class` .

Vysvětlení
Pole záhlaví `Content-Type` zadané v požadavku **POST** není kompatibilní s hodnotou záhlaví `x-msg-class` .

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Změňte pole záhlaví `Content-Type` tak, aby bylo podporováno. Záhlaví `Content-Type` musí být kompatibilní s určeným polem záhlaví `x-msg-class` .

MQHTTP40005: Chybná hodnota záhlaví zprávy

Bylo zadáno podporované pole záhlaví s hodnotou, která není platná pro uvedený požadavek.

Vysvětlení
Bylo zadáno podporované pole záhlaví s hodnotou, která není platná pro uvedený požadavek.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Změňte hodnotu zadanou pro dané pole záhlaví na hodnotu, která je platná. Zkontrolujte velikost uvedené hodnoty, protože některá pole záhlaví mají hodnoty rozlišující velikost písmen.

MQHTTP40006: *Header_name* není platné záhlaví požadavku

Záhlaví, které je platné pouze ve zprávě s odezvou HTTP, bylo uvedeno ve zprávě požadavku HTTP.

Vysvětlení
Záhlaví, které je platné pouze ve zprávě s odezvou HTTP, bylo uvedeno ve zprávě požadavku HTTP.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Odeberte všechna záhlaví z požadavku HTTP, která jsou platná pouze v odezvě HTTP; například `x-msg-timestamp`.

MQHTTP40007: *Header_name* je platný pouze na ...

Záhlaví bylo určeno v požadavku HTTP, ale pole záhlaví není pro dané slovo požadavku platné.

Vysvětlení
Záhlaví bylo určeno v požadavku HTTP, ale pole záhlaví není pro dané slovo požadavku platné.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Odeberte všechna záhlaví z požadavku HTTP, která nejsou platná pro dané příkazové slovo požadavku. Například, `x-msg-encoding` je platné pro požadavky HTTP **POST** , ale není platné pro požadavky HTTP **GET** nebo HTTP **DELETE** .

MQHTTP40008: *Header_name* maximální délka je ...

Byla překročena maximální délka pro dané pole záhlaví.

Vysvětlení
Byla překročena maximální délka pro dané pole záhlaví.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Změňte hodnotu v poli záhlaví na hodnotu, která je v rozsahu povoleném pro pole záhlaví.

MQHTTP40009: Pole záhlaví *`header_field`* není platné pro ...

Pole záhlaví zadané v požadavku HTTP není podporováno poskytovatelem systému zpráv, ke kterému je připojen most produktu WebSphere MQ pro protokol HTTP.

Vysvětlení
Pole záhlaví zadané v požadavku HTTP není podporováno poskytovatelem systému zpráv, ke kterému je připojen most produktu WebSphere MQ pro protokol HTTP. K chybě dochází v případě, že je použit poskytovatel systému zpráv, který nepodporuje všechny funkce mostu produktu WebSphere MQ pro protokol HTTP.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Odeberte nepodporované záhlaví z požadavku HTTP.

MQHTTP40010: Zprávu s typem `Content-Type` *`content_type`* nelze analyzovat

Obsah požadavku HTTP není kompatibilní s typem `Content-Type` požadavku.

Vysvětlení
Obsah požadavku HTTP není kompatibilní s typem `Content-Type` požadavku. Běžnou příčinou je špatně utvořený `application/x-www-form-urlencoded` nebo `application/xml data`.

Stavový kód HTTP
`400 Bad Request`

Odpověď programátora
Opravte obsah požadavku HTTP tak, aby byl ve správném formátu pro typ `Content-Type` požadavku.

MQHTTP40301: Nemáte přístup k ...

Most produktu WebSphere MQ pro protokol HTTP se nepodařilo ověřit u zadaného místa určení.

Vysvětlení
Most produktu WebSphere MQ pro protokol HTTP se nepodařilo ověřit u zadaného místa určení.

Stavový kód HTTP
`403 Forbidden`

Odpověď programátora
Změňte vlastnosti ověření cíle tak, aby most produktu WebSphere MQ pro HTTP byl autorizován pro připojení k němu. Případně zadejte místo určení, ke kterému je most produktu WebSphere MQ pro protokol HTTP autorizován pro připojení.

MQHTTP40302: Je zakázáno ...

Most produktu WebSphere MQ pro protokol HTTP se nepodařilo připojit ke správci front.

Vysvětlení
Most produktu WebSphere MQ pro protokol HTTP se nepodařilo připojit ke správci front. Most produktu WebSphere MQ pro konfiguraci zabezpečení HTTP je nesprávný.

Stavový kód HTTP
`403 Forbidden`

Odpověď programátora
Změňte konfiguraci ověření správce front tak, aby most produktu WebSphere MQ pro protokol HTTP byl autorizován pro připojení k tomuto produktu. Případně můžete nakonfigurovat most produktu WebSphere MQ pro protokol HTTP tak, aby se připojoval ke správci front, k němuž má autorizaci pro připojení.

MQHTTP40401: Cíl *destination_name* nebyl nalezen.

Místo určení zadané v identifikátoru URI požadavku HTTP nelze nalézt pomocí mostu WebSphere MQ pro protokol HTTP.

Vysvětlení
Místo určení zadané v identifikátoru URI požadavku HTTP nelze nalézt pomocí mostu WebSphere MQ pro protokol HTTP.

Stavový kód HTTP
`404 Not found`

Odpověď programátora
Zkontrolujte, zda cíl uvedený v identifikátoru URI požadavku HTTP existuje, nebo určete alternativní místo určení.

MQHTTP40501: Metoda *method_name* není povolena

Metoda uvedená v požadavku HTTP není podporována mostem WebSphere MQ pro protokol HTTP.

Vysvětlení
Metoda uvedená v požadavku HTTP není podporována mostem WebSphere MQ pro protokol HTTP.

Stavový kód HTTP
`405 Method not allowed`

Odpověď programátora
Změňte metodu určenou v požadavku HTTP na takovou metodu, kterou podporuje most produktu WebSphere MQ pro protokol HTTP.

MQHTTP41301: Vystavená zpráva byla příliš velká pro místo určení.

Místo určení uvedené v identifikátoru URI požadavku POST protokolu HTTP nemůže přijímat zprávy, které jsou tak dlouho, jak je zpráva uvedená v požadavku HTTP.

Vysvětlení
Místo určení uvedené v identifikátoru URI požadavku POST protokolu HTTP nemůže přijímat zprávy, které jsou tak dlouho, jak je zpráva uvedená v požadavku HTTP.

Stavový kód HTTP
`413 Request entity too large`

Odpověď programátora
Zmenšete velikost zprávy uvedené v požadavku HTTP. Případně zadejte místo určení, které může podporovat zprávy požadované délky.

MQHTTP41501: Znaková sada typu média není podporována.

Znaková sada určená v poli záhlaví `Content-Type` není podporována pomocí mostu WebSphere MQ pro protokol HTTP.

Vysvětlení
Znaková sada určená v poli záhlaví `Content-Type` není podporována pomocí mostu WebSphere MQ pro protokol HTTP.

Stavový kód HTTP
`415 Unsupported media type`

Odpověď programátora
Změňte znakovou sadu pole záhlaví `Content-Type` na takovou znakovou sadu, kterou podporuje most produktu WebSphere MQ pro protokol HTTP.

MQHTTP41502: Médium-typ *`media-type`* není podporováno ...

Typ média určený v požadavku HTTP není podporován mostem WebSphere MQ pro protokol HTTP pro zadané slovo HTTP.

Vysvětlení
Typ média určený v požadavku HTTP není podporován mostem WebSphere MQ pro protokol HTTP pro zadané slovo HTTP.

Stavový kód HTTP
`415 Unsupported media type`

Odpověď programátora
Změňte typ média uvedený v požadavku HTTP na takový typ, který je podporován produktem WebSphere MQ Bridge for HTTP pro zadané příkazové slovo HTTP.

MQHTTP41503: Typ média *`media-type`* není podporován ...

Typ média určený v požadavku HTTP není podporován funkcí WebSphere MQ pro protokol HTTP pro určené pole záhlaví `x-msg-class` .

Vysvětlení
Typ média určený v požadavku HTTP není podporován funkcí WebSphere MQ pro protokol HTTP pro určené pole záhlaví `x-msg-class` .

Stavový kód HTTP
`415 Unsupported media type`

Odpověď programátora
Změňte typ média určený v požadavku HTTP na takový typ, který je podporován produktem WebSphere MQ Bridge pro protokol HTTP pro určený pole záhlaví `x-msg-class` .

MQHTTP41701: Záhlaví HTTP `Expect` není podporováno.

Most produktu WebSphere MQ pro protokol HTTP nepodporuje pole záhlaví `Expect` .

Vysvětlení
Záhlaví `Expect` bylo určeno v požadavku HTTP. Most produktu WebSphere MQ pro protokol HTTP nepodporuje pole záhlaví `Expect` .

Stavový kód HTTP
`417 Expectation failed`

Odpověď programátora
Odeberte záhlaví `Expect` z požadavku HTTP.

MQHTTP50001: Došlo k neočekávanému problému ...

V mostu WebSphere MQ pro protokol HTTP se vyskytla chyba.

Vysvětlení
V mostu WebSphere MQ pro protokol HTTP se vyskytla chyba.

Stavový kód HTTP
`500 Internal server error`

Odpověď programátora
Obraťte se na administrátora systému produktu WebSphere MQ Bridge pro protokol HTTP.

MQHTTP50201: Došlo k chybě mostu WebSphere MQ pro protokol HTTP a správce front.

Došlo k chybě mostu WebSphere MQ pro protokol HTTP a správce front.

Vysvětlení
Došlo k chybě mostu WebSphere MQ pro protokol HTTP a správce front.

Stavový kód HTTP
`502 Bad Gateway`

Odpověď programátora
Obraťte se na administrátora systému produktu WebSphere MQ Bridge pro protokol HTTP.

MQHTTP50401: Vypršel časový limit načítání zpráv.

V období časového limitu nebyla vrácena žádná zpráva odpovídající zadaným parametrům požadavku v HTTP **GET** nebo HTTP **DELETE** .

Vysvětlení
V období časového limitu nebyla vrácena žádná zpráva odpovídající zadaným parametrům požadavku v HTTP **GET** nebo HTTP **DELETE** . Návratový kód indikuje, že kdykoli během životnosti požadavku HTTP nebyla k dispozici žádná vhodná zpráva.

Stavový kód HTTP
`504 Gateway timeout`

Odpověď programátora
Pokud byla očekávána zpráva, zkontrolujte pole záhlaví požadavku HTTP, např. `x-msg-correlId` a `x-msg-msgid`. Zkontrolujte, zda je cíl určený v identifikátoru URI požadavku HTTP správný. Zkuste prodloužit čekací dobu požadavku HTTP pomocí pole záhlaví `x-msg-wait` .

MQHTTP50501: HTTP 1.1 a směrem nahoru ...

Protokol HTTP použitý v požadavku HTTP není podporován funkcí WebSphere MQ Bridge for HTTP.

Vysvětlení
Protokol HTTP použitý v požadavku HTTP není podporován funkcí WebSphere MQ Bridge for HTTP.

Stavový kód HTTP
`505 HTTP version not supported`

Odpověď programátora
Změňte požadavek HTTP tak, aby používal protokol HTTP V1.1 nebo vyšší.

Typy zpráv a mapování zpráv pro produkt WebSphere Bridge for HTTP

Produkt WebSphere MQ Bridge for HTTP podporuje čtyři třídy zpráv, TEXT, BYTES, STREAM a MAP. Třídy zpráv jsou mapovány na typy zpráv JMS a HTTP `Content-Type`.

HTTP POST

Typ zprávy, který dorazí na místo určení, závisí na hodnotě záhlaví `x-msg-class` nebo `Content-Type` požadavku HTTP. Tabulka 604 na stránce 1207 zobrazuje typ HTTP `Content-Type` , který odpovídá každé třídě `x-msg-class`. Pro nastavení typu zprávy a formátu zprávy lze použít buď pole, nebo pole. Jsou-li obě pole nastavena a jsou nastavena nekonzistentně, je vrácena hodnota `Bad Request exception` (`HTTP 400, MQHTTP20004`).

Tabulka 604. Mapování mezi `x-msg-class` a HTTP `Content-Type`

x-zpr-třída	**HTTP Content-Type**
BYTES	`application/octet-stream` `application/xml`
TEXT	`text/*`
MAP	`application/x-www-form-urlencoded` `application/xml` (volitelné)
STREAM	`application/xml` (volitelné)

Pokud je typ zprávy JMS nastaven v záhlaví MQRFH2 , je mapován v souladu s Tabulka 605 na stránce 1207.

Tabulka 605. Mapování mezi typy zpráv `x-msg-class` a JMS.

x-zpr-třída	**Typ zprávy platformy JMS**
BYTES	`jms_bytes`
TEXT	`jms_text`
MAP	`jms_map`

Tabulka 605. Mapování mezi typy zpráv x-msg-class a JMS. (pokračování)	
x-zpr-třída	**Typ zprávy platformy JMS**
STREAM	jms_stream

Typ zprávy JMS je vždy nastaven pro třídu zpráv MAP nebo STREAM. Není nastaven vždy pro třídu zpráv BYTES nebo TEXT. Má-li být pro požadavek vytvořen MQRFH2 , typ zprávy JMS je vždy nastaven. Jinak, pokud není vytvořen MQRFH2 , není nastaven žádný typ zprávy JMS. Pokud jsou vlastnosti uživatele nastaveny v požadavku pomocí záhlaví x-msg-usr , vytvoří se MQRFH2 .

Je-li nastaven typ zprávy JMS, pak je formát zprávy nastaven na MQFMT_NONE, viz Tabulka 607 na stránce 1208:

Tabulka 606. Mapování mezi formátem zpráv x-msg-class a WebSphere MQ		
x-zpr-třída	**Formát zprávy se zprávou MQRFH2 přítomná ve zprávě**	**Formát zprávy s *ne* MQRFH2 přítomnou ve zprávě**
BYTES	MQFMT_NONE	MQFMT_NONE
TEXT	MQFMT_NONE	MQFMT_STRING
MAP	MQFMT_NONE	Nemožné
STREAM	MQFMT_NONE	Nemožné

HTTP GET nebo DELETE

Načtený typ nebo formát zprávy určuje hodnotu záhlaví x-msg-class a Content-Type odezvy HTTP. Záhlaví x-msg-class je vráceno pouze v případě, že je požadováno v požadavku x-msg-headers .

Část Tabulka 607 na stránce 1208 popisuje mapování mezi třídou x-msg-class a Content-Typea typem zprávy načtenou z fronty nebo tématu.

Tabulka 607. Mapování typů zpráv na x-msg-class a Content-Type			
Formát zprávy	**Typ zprávy platformy JMS**	**x-zpr-třída**	**Content-Type**
Vše kromě MQFMT_STRING	Není	BYTES	application/octet-stream
MQFMT_STRING	Není	TEXT	text/plain
MQFMT_NONE	jms_bytes	BYTES	application/octet-stream
MQFMT_NONE	jms_text	TEXT	text/plain
MQFMT_NONE	jms_map	MAP	application/xml
MQFMT_NONE	jms_stream	STREAM	application/xml

Serializace třídy zpráv MAP a STREAM

Třídy zpráv MAP a STREAM jsou serializovány zpět klientovi v odpovědi HTTP stejným způsobem, jakým je zpráva serializována do fronty.

Pro produkt MAPjsou triplety názvu XML, typu a hodnoty kódovány jako:

```
<map>
    <elt name="elementname1" dt="datatype1">value1</elt>
    <elt name="elementname2" dt="datatype2">value2</elt>
```

```
    ...
</map>
```

STREAM je jako MAP, ale nemá názvy prvků:

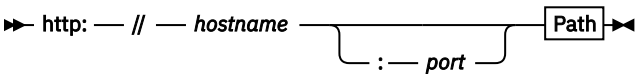
```
<stream>
    <elt dt="datatype1">value1</elt>
    <elt dt="datatype2">value2</elt>
    ...
</stream>
```

Poznámka: `datatype` je jeden z datových typů definovaných pro definování uživatelem definovaných vlastností a vypsaných v “usr: HTTP x-msg-usr -záhlaví entity” na stránce 1197. Atribut `dt="string"` je vynechán pro řetězcové prvky, protože výchozí datový typ je `string`.

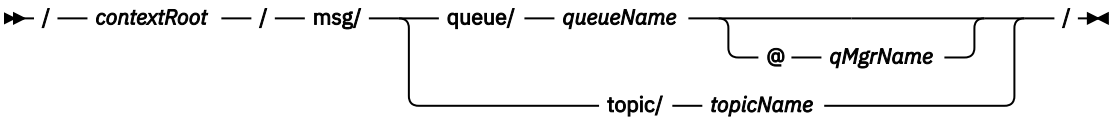
Formát identifikátoru URI

Identifikátory URI zachycené pomocí mostu produktu WebSphere MQ pro protokol HTTP.

Syntax



Path



Poznámka:

1. If a question mark (?) is used it must be substituted with `%3f`. For example, `orange?topic` should be specified as `orange%3ftopic`.
2. @*qMgrName* is only valid on an HTTP **POST**

Popis

Implementujte most produktu WebSphere MQ pro servlet HTTP na aplikační server JEE s kontextovým kořenovým adresářem adresáře *contextRoot*. Požadavky na

```
http://hostname:port/context_root/msg/queue/queueName@qMgrName
```

a

```
http://hostname:port/context_root/msg/topic/topicString
```

zachytávané produktem WebSphere MQ Bridge for HTTP.

Třídy a rozhraní .NET produktuIBM WebSphere MQ

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

MQAsyncStatus Třída .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 1210
- “Konstruktory” na stránce 1211

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.

MQAuthenticationInformationRecord Třída .NET

MQAuthenticationInformationRecord lze použít k zadání informací o ověřovateli, který má být použit v připojení klienta WebSphere MQ SSL. MQAuthenticationInformationRecord zapouzdřuje záznam ověřovacích informací, MQAIR.

Třída

```
System.Object
    |
    └ IBM.WMQ.MQAuthenticationInformationRecord
```

```
public class IBM.WMQ.MQAuthenticationInformationRecord extends System.Object;
```

- “Vlastnosti” na stránce 1211
- “Konstruktory” na stránce 1211

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();

MQDestination Třída .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" na stránce 1212
- "Metody" na stránce 1212
- "Konstruktory" na stránce 1214

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 produktu WebSphere MQ z konkrétního produktu MQQueueManager jsou synchronní. Pokud tedy provedete operaci get s čekáním, budou všechny ostatní podprocesy

používající stejný produkt MQQueueManager blokovány dalšími voláními produktu WebSphere MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu WebSphere MQ současně, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

message
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.

getMessageOptions
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 .

MaxMsgSize
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, bude zpráva ponechána 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 WebSphere 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);
```

message
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).

putMessageOptions
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.

MQEnvironment Třída .NET

Pomocí obslužného programu MQEnvironment můžete řídit, jak je volán konstruktor MQQueueManager a vybrat připojení klienta WebSphere MQ MQI. Třída MQEnvironment obsahuje vlastnosti, které řídí chování produktu WebSphere MQ.

Třída

```
System.Object
    │
    └─ IBM.WMQ.MQEnvironment
```

```
public class IBM.WMQ.MQEnvironment extends System.Object;
```

- “Vlastnosti-pouze klient” na stránce 1214
- “Vlastnosti” na stránce 1215
- “Konstruktory” na stránce 1216

Vlastnosti-pouze klient

Testuje se test produktu MQException při získávání vlastností.

public static int CertificateValPolicy {get; set;}

Nastavit, která zásada ověření certifikátu SSL/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 MQQueueManager v režimu klienta *musíte* nastavit vlastnost kanálu.

public static int FipsRequired {get; set;}

Chcete-li v produktu WebSphere MQpoužívat pouze šifrovací algoritmy standardu FIPS, zadejte hodnotu `MQC.MQSSL_FIPS_YES` . 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 počítače, na kterém je umístěn server WebSphere 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 WebSphere MQ naslouchá příchozím požadavkům na připojení. Výchozí hodnota je 1414.

public static string SSLCipherSpec {get; set;}

Nastavte hodnotu `SSLCipherSpec` na hodnotu sady CipherSpec nastavené v kanálu SVRCONN, aby bylo možné povolit zabezpečení SSL pro připojení. Výchozí hodnota je Null a SSL 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 SSL před opětovným získáním tajného klíče.

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 .NET, když 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 tak, jako byl před produktem WebSphere MQ verze 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 WebSphere 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 WebSphere MQ vytvoří 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()

MQException Třída .NET

Použijte MQException k vyhledání dokončení a kód příčiny selhání funkce WebSphere MQ . Pokud dojde k chybě produktu WebSphere 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 1217
- “Konstruktory” na stránce 1217

Vlastnosti

public int CompletionCode {get; set;}
Kód dokončení WebSphere MQ přidružený k chybě. Možné hodnoty jsou:

- `MQException.MQCC_OK`
- `MQException.MQCC_WARNING`
- `MQException.MQCC_FAILED`

public int ReasonCode {get; set;}
WebSphere Kód příčiny MQ popisující chybu.

Konstruktory

public MQException(int *completionCode*, int *reasonCode*)

completionCode
Kód dokončení WebSphere MQ .

reasonCode
Kód dokončení WebSphere MQ .

MQGetMessageOptions Třída .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 1217
- “Konstruktory” na stránce 1220

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`, `readLine`a `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.

`MQManagedObject` Třída .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 1220
- "Metody" na stránce 1221
- "Konstruktory" na stránce 1222

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 WebSphere MQ . Metoda `OpenOptions.set` je ignorována a nezpůsobuje chybu. Odběry nemají žádné `OpenOptions`.

Metody

public virtual void Close();
Vyvolá `MQException`.

Zavře objekt. Po volání `Close`nejsou 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.

selector
Celé číslo označující, na který atribut se dotazujete.

length
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`.

selectors
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.

selectors
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.

selector
Celé číslo označující, který atribut se nastavuje.

value
Řetězec, který se má nastavit jako hodnota atributu.

length
Celé číslo udávající délku požadovaného řetězce.

Konstruktory

protected MQManagedObject()
Metoda konstruktoru. Tento objekt je abstraktní základní třída, která nemůže být převedena na instanci sama.

MQMessage Třída .NET

Použijte MQMessage pro přístup k popisovači zprávy a datům pro zprávu WebSphere MQ . Produkt MQMessage zapouzdřuje zprávu produktu WebSphere 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

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 .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_NONE`pomáhá 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
   Little-endian celá čísla, jak se používají v architektuře Intel.

2. Pakovaný desetinný

   MQC.MQENC_DECIMAL_NORMAL
   packed-endian big-endian, který je používán systémem 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
   z/OS plovoucí řádové čárky.

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 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 příčinyMQRC_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 podporovány správci front z/OS . Nepoužívejte je v případě, že aplikace pravděpodobně přistupuje ke správci front systému 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 platformy .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_NORMAL`nebo 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 WebSphere 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 608. 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)`

Tabulka 608. Metody čtení a zápisu zpráv (pokračování)

Typ cíle	Podpisy metody
Double	`public double ReadDouble()` `public void WriteDouble(double value)`
Float	`public float ReadFloat()` `public void WriteFloat(float value)`
Int2	`public void WriteInt2(int value)`

Int4	`public int readDecimal4()`
	`public int ReadInt()`
	`public int ReadInt4()`
	`public void WriteInt(int value)`
	`public void WriteInt4(int value)`

Int8	`public void WriteInt8(long value)`

Long	`public long ReadDecimal8()`
	`public long ReadLong()`
	`public long ReadInt8()`
	`public void WriteLong(long value)`

Object	`public Object ReadObject()` `public void WriteObject(Object object)`

Short	`public short ReadShort()`
	`public short ReadDecimal2()`
	`public short ReadInt2()`
	`public void WriteShort(int value)`

string	`public string ReadString(int length)` `public void WriteString(string string)`
Unsigned Short	`public ushort ReadUnsignedShort()` `public ushort ReadUInt2()`

Unicode	`public string ReadLine()`
	`public char ReadChar()`
	`public void WriteChar(int value)`
	`public void WriteChars(string string)`

Tabulka 608. Metody čtení a zápisu zpráv (pokračování)	
Typ cíle	**Podpisy metody**
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.

name
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.

name
Název vlastnosti, se kterou se má shodovat.

- Všechny metody SetProperty a GetProperty generují MQException

Tabulka 609. 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);`

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);`

Tabulka 609. Metody SetProperty a GetProperty (pokračování)	
Typ	**Podpisy metody**

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);`

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.

MQProcess Třída .NET

Použijte produkt MQProcess k zadání dotazu na atributy procesu produktu WebSphere 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 1233
- “Konstruktory” na stránce 1234

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 WebSphere 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 WebSphere MQ v tomto správci front za účelem zjišťování atributů 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 .

MQPropertyDescriptor Třída .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 1236
- "Konstruktory" na stránce 1237

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 definovanou vlastnost produktu WebSphere 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; }`

Chcete-li určit úroveň podpory vyžadovanou pro vlastnosti zprávy definované produktem WebSphere MQ, nastavte volbu `Podpora` . 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 definované produktem WebSphere MQ .

MQC.MQPD_SUPPORT_REQUIRED
Podpora pro vlastnost je povinná. Pokud zprávu vložíte do správce front, který nepodporuje vlastnost definované produktem WebSphere 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 definované produktem WebSphere 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.

MQPutMessageOptions Třída .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 1237“Konstruktory” na stránce 1239

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.

.NET nepodporuje distribuční seznamy, ale `InvalidDestCount` je nastaven při otevření jedné fronty.

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.

.NET nepodporuje distribuční seznamy, ale InvalidDestCount je nastaven při otevření jedné fronty.

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 prostředí .NET podporované.

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.

.NET nepodporuje distribuční seznamy, ale InvalidDestCount je nastaven při otevření jedné fronty.

Konstruktory

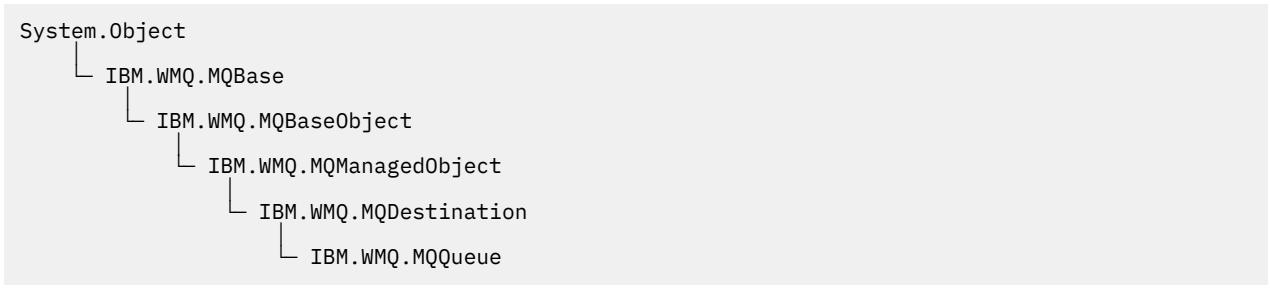
public MQPutMessageOptions();

Vytvořte nový objekt MQPutMessageOptions bez nastavení voleb a prázdnou hodnotu ResolvedQueueName a ResolvedQueueManagerName.

MQQueue Třída .NET

Pomocí produktu MQQueue můžete odesílat a přijímat zprávy a dotazy na atributy fronty produktu WebSphere MQ . Vytvořte objekt MQQueue pomocí konstruktoru nebo metody MQQueueManager.AccessProcess .

Třída




```
public class IBM.WMQ.MQQueue extends IBM.WMQ.MQDestination;
```

- “Vlastnosti” na stránce 1240
- “Metody” na stránce 1242
- “Konstruktory” na stránce 1244

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 WebSphere 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 produktu WebSphere MQ z konkrétního produktu MQQueueManager jsou synchronní. Pokud tedy provedete operaci get s čekáním, budou všechny ostatní podprocesy používající stejný produkt MQQueueManager blokovány dalšími voláními produktu WebSphere MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu WebSphere MQ současně, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

message
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.

getMessageOptions
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 .

MaxMsgSize
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, bude zpráva ponechána 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 WebSphere 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);
```

message
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).

putMessageOptions
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.

message
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).

putMessageOptions
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.

message
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.

putMessageOptions
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.

message
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.

putMessageOptions
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.

dynamicQueueName
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.

dynamicQueueName
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.

MQQueueManager Třída .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 1247
- "Metody" na stránce 1251
- "Konstruktory" na stránce 1256

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 Bridge.

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 WebSphere 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 trasování kanálu iniciátoru.

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;}
Název skupiny, kterou modul listener TCP zpracovává příchozí přenosy pro skupinu sdílení front, se musí připojit. Tato skupina se připojuje k této skupině při použití správce pracovní zátěže pro podporu dynamických služeb názvů domény (DDNS).

public int DNSWLM {get;}
Zda se má modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, registrovat ve správci pracovní zátěže pro systém DDNS.

public int IPAddressVersion {get;}
Který protokol IP (IPv4 nebo IPv6) má být použit 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;}
Časový interval (v sekundách) mezi pokusy o restartování modulu listener po selhání APPC nebo TCP/IP pomocí produktu WebSphere MQ .

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 iniciátor kanálu, vydá se příkaz SET APPC=xx příkazu z/OS .

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 se generují události SSL.

public int SSLFips {get;}
Určuje, zda mají být použity pouze algoritmy certifikované podle standardu FIPS, pokud je šifrování prováděno v produktu WebSphere MQ, nikoli kryptografickému hardwaru.

public int SSLKeyResetCount {get;}
Označuje počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL před opětovným získáním tajného klíče.

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 WebSphere MQ v tomto správci front za účelem zjišťování atributů 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.

dynamicQueueName
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. BuďtopicObject , 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.

destination
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 názvů 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 označeným názvem 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

options

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.

properties
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, je-li 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*.

type
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.

message

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).

putMessageOptions

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.

options
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`

properties

Parametr vlastností vezme řadu párů klíč/hodnota, které potlačí vlastnosti nastavené pomocí `MQEnvironment`; viz příklad “Potlačit vlastnosti MQEnvironment” na stránce 1259. 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`

channel
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, bude použit výchozí port (konfigurovaný v produktu `mqclient.ini`).

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.

V příkazovém řádku nastavte hodnotu

```
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/Hostname1(Port1),Hostname2(Por2),Hostname3(Port3)
For Example:
MQSERVER=SYSTEM.DEF.SVRCONN/TCP/fred.mq.com(5266),nick.mq.com(6566),jack.mq.com(8413)
```

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 WebSphere MQ port)
MQEnvironment.Channel  = "channel.name"; // the case sensitive
                                         //  name of the
                                         // SVR CONN channel on
                                         // the queue manager
MQQueueManager qMgr    = new MQQueueManager("MYQM");
```

Obrázek 43. 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 44. 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
                                                    // through which reconnection happens

properties.Add(MQC.CONNECTION_NAME_PROPERTY,"fred.mq.com(4789),nick.mq.com(4790)");// The list
of
                                                  // queue managers through which reconnect
happens

MQ QueueManager qmgr = new MQQueueManager("qmgrname", properties);
```

Obrázek 45. Automatické opětovné připojení klienta ke správci front

MQSubscription Třída .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 1260
- “Metody” na stránce 1260
- “Konstruktory” na stránce 1260

Vlastnosti

Přístup k vlastnostem odběru pomocí třídy MQManagedObject ; viz “Vlastnosti” na stránce 1220.

Metody

Přistupte k odběru metod Inquire, Set a Get odběru přístupu pomocí třídy MQManagedObject , viz “Metody” na stránce 1221.

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 RequestPublicationUpdateotevřete 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.

options

MQC.MQSRO_FAIL_IF_QUIESCING
Metoda selže, pokud se správce front nachází v klidovém stavu. V systému z/OS, pro aplikaci CICS nebo IMS , produkt MQC.MQSRO_FAIL_IF_QUIESCING také vynutí selhání metody, pokud je připojení 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.

MQTopic Třída .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 1261
- "Metody" na stránce 1261
- "Konstruktory" na stránce 1263

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 WebSphere 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);
```

message
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).

putMessageOptions
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 produktu WebSphere MQ z konkrétního produktu MQQueueManager jsou synchronní. Pokud tedy provedete operaci get s čekáním, budou všechny ostatní podprocesy používající stejný produkt MQQueueManager blokovány dalšími voláními produktu WebSphere MQ , dokud nebude provedeno volání funkce Get. Potřebujete-li více podprocesů pro přístup k produktu WebSphere MQ současně, musí každý podproces vytvořit svůj vlastní objekt MQQueueManager .

message
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.

getMessageOptions
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 .

MaxMsgSize
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, bude zpráva ponechána 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. BuďtopicObject , 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.

destination
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 názvů 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 označeným názvem 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

options

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.

properties
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. Buďtopic Object , 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.

destination
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 názvů 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 označeným názvem 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

options

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.

properties
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 .NET produktu IMQObjectTrigger

Implementujte produkt IMQObjectTrigger ke zpracování zpráv předávaných monitorem .NET v produktu **runmqdnm** .

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.

queue
Monitorovaná fronta.

message
Zpráva načtená z fronty.

param
Data předaná z `UserParameter`.

Rozhraní .NET produktu MQC

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 použít k zakódování zpráv prostředí .NET IBM WebSphere 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

Znaková sada	Popis
420	ibm420
424	ibm424
737	ibm737 /PC-řečtina
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

Znaková sada	Popis
933	ibm933
935	ibm935
937	ibm937
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 Hebrejština
1256	Windows Arabština
1257	Windows Baltské jazyky
1258	Windows Vietnamština
1381	ibm1381
1383	ibm1383
2022	JIS
5601	ksc-5601 Korejština

Znaková sada	Popis
33722	ibm33722

IBM WebSphere MQ Třídy C++

Třídy jazyka C++ produktu IBM WebSphere MQ zapouzdřují rozhraní MQI (Message Queue Interface) produktu IBM WebSphere 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 členy datové struktury WebSphere MQ (viz “Křížový odkaz C++ a MQI” na stránce 1271). Podrobný popis viz “Atributy objektů” na stránce 753.

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 Kód příčiny rozhraní API), 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 1286

- “Třída C++ ImqBinary” na stránce 1288
- “Třída C++ ImqCache” na stránce 1290
- “Třída C++ ImqChannel” na stránce 1293
- “ImqCICSBridgeTřída C++ záhlaví” na stránce 1299
- “Třída C++ ImqDeadLetterHeader” na stránce 1305
- “Třída C++ seznamu ImqDistribution” na stránce 1307
- “Třída C++ ImqError” na stránce 1309
- “Třída C++ ImqGetMessageOptions” na stránce 1310
- “Třída C++ ImqHeader” na stránce 1313
- “Třída C++ záhlaví ImqIMSBridge” na stránce 1315
- “Třída C++ ImqItem” na stránce 1318
- “Třída C++ ImqMessage” na stránce 1319
- “Třída C++ produktu ImqMessageTracker” na stránce 1326
- “Třída C++ ImqNamelist” na stránce 1329
- “Třída C++ ImqObject” na stránce 1330
- “Třída C++ ImqProcess” na stránce 1336
- “ImqPutMessageOptions Třída C++” na stránce 1337
- “Třída C++ ImqQueue” na stránce 1339
- “Třída C++ správce ImqQueue” na stránce 1350
- “Třída C++ záhlaví ImqReference” na stránce 1365
- “Třída C++ ImqString” na stránce 1368
- “Třída C++ ImqTrigger” na stránce 1372
- “Třída C++ záhlaví ImqWork” na stránce 1375

Křížový odkaz C++ a MQI

Tato kolekce témat obsahuje informace týkající se jazyka C++ pro rozhraní MQI.

Přečtěte si tyto informace spolu s produktem “Datové typy použité v rozhraní MQI” na stránce 216.

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 procedurálních rozhraní produktu WebSphere MQ . Třídy ImqBinary, ImqDistributionList a ImqString nemají žádné atributy, které spadají do této kategorie a jsou vyloučeny.

Tabulka 610. Struktura dat, třída a křížový odkaz na soubor začlenění

datová struktura	**Třída**	**Zahrnout soubor**
MQAIR	Záznam ImqAuthentication	imqair.hpp
	ImqBinary	imqbin.hpp
	ImqCache	imqcac.hpp
MQCD	ImqChannel	imqchl.hpp
MQCIH.	Záhlaví ImqCICSBridge	imqcih.hpp
MQDLH	ImqDeadLetterHeader	imqdlh.hpp
MQOR	Seznam ImqDistribution	imqdst.hpp
	ImqError	imqerr.hpp
MQGMO	ImqGetMessageOptions	imqgmo.hpp

Tabulka 610. Struktura dat, třída a křížový odkaz na soubor začlenění (pokračování)		
datová struktura	**Třída**	**Zahrnout soubor**
	ImqHeader	imqhdr.hpp
MQIIH.	Záhlaví ImqIMSBridge	imqiih.hpp
	ImqItem	imqitm.hpp
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.

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 .

Atribut	**Volání**
automatická vyrovnávací paměť	MQGET
Délka vyrovnávací paměti	MQGET

Atribut	Volání
ukazatel vyrovnávací paměti	MQGET, MQPUT
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 .

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
	MQCD	SendUserDataPtr	MQCONNX

Atribut	datová struktura	Pole	Volání
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 záhlaví ImqCICSBridge

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záhlaví ImqCICSBridge.

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
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 .

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 .

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 .

Atribut	datová struktura	Pole
stav skupiny	MQGMO	GroupStatus
volby shody	MQGMO	MatchOptions
token zprávy	MQGMO	MessageToken
volby	MQGMO	Volby
vyřešený název fronty	MQGMO	ResolvedQName
vrácená délka	MQGMO	ReturnedLength
segmentace	MQGMO	Segmentace
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 .

Atribut	datová struktura	Pole
znaková sada	MQDLH, MQIIH	CodedCharSetId
kódování	MQDLH, MQIIH	Kódování
formát	MQDLH, MQIIH	Formát
příznaky záhlaví	MQIIH, MQRMH	Příznaky

Křížový odkaz záhlaví ImqIMSBridge

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

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 .

Atribut	Volání
id struktury	MQGET

Křížový odkaz ImqMessage

Křížový odkaz na atributy, datové struktury, pole a volání pro třídu C++ ImqMessage .

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	

Atribut	datová struktura	Pole	Volání
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.

Atribut	datová struktura	Pole
Token evidence	MQMD	AccountingToken
ID korelace	MQMD	CorrelId
Zpětná vazba	MQMD	Zpětná vazba
ID skupiny	MQMD	GroupId
ID zprávy	MQMD	MsgId

Křížový odkaz ImqNamelist

Křížový odkaz na atributy, dotazy a volání pro třídu C++ ImqNamelist .

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 .

Atribut	datová struktura	Pole	Inquiry	Volání
Datum změny			MQCA_ALTERATION_DATE	MQINQ
Čas změny			MQCA_ALTERATION_TIME	MQINQ
Jméno alternativního uživatele	MQOD	AlternateUserid		

Atribut	datová struktura	Pole	Inquiry	Volání
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.

Atribut	Inquiry	Volání
ID aplikace	MQCA_APPL_ID	MQINQ
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 611. Křížový odkaz ImqPutMessageOptions

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 612. 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 vytíření klastru			MQIA_CLWL_Q_RANK	MQINQ
Priorita vytíření klastru			MQIA_CLWL_Q_PRIORITY	MQINQ
Pracovní zátěž klastru - použitá fronta			MQIA_CLWL_USEQ.	MQINQ
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

Tabulka 612. Křížový odkaz ImqQueue (pokračování)

Atribut	datová struktura	Pole	Inquiry	Volání
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
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

Tabulka 612. Křížový odkaz ImqQueue (pokračování)				
Atribut	**datová struktura**	**Pole**	**Inquiry**	**Volání**
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.

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

Atribut	datová struktura	Pole	Inquiry	Volání
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
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

Atribut	datová struktura	Pole	Inquiry	Volání
Název fronty vstupu příkazů			MQCA_COMMAND_INPUT_Q_NAME	MQINQ
Úroveň příkazů			ÚROVEŇ PŘÍKAZU MQIA_COMMAND_LEVEL	MQINQ
Řízení příkazového serveru			MQIA_CMD_SERVER_CONTROL	MQINQ
Volby připojení	MQCNO	Volby		MQCONN, MQCONNX
ID připojení	MQCNO	ConnectionId		MQCONNX
Stav připojení				MQCONN, MQCONNX, MQDISC
Značka připojení	MQCD	ConnTag		MQCONNX
Kryptografický hardware	MQSCO	CryptoHardware		MQCONNX
název fronty nedoručených zpráv			MQCA_DEAD_LETTER_Q_NAME	MQINQ
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

Atribut	datová struktura	Pole	Inquiry	Volání
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
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

Atribut	datová struktura	Pole	Inquiry	Volání
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
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.

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

Atribut	datová struktura	Pole
Zdrojový název	MQRMH	SrcNameDélka, SrcNameOffset

Křížový odkaz ImqTrigger

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

Tabulka 613. Křížový odkaz ImqTrigger

Atribut	datová struktura	Pole
ID aplikace	MQTM	ApplId
Typ aplikace	MQTM	ApplType
Data prostředí	MQTM	EnvData
Název procesu	MQTM	ProcessName
Název fronty	MQTM	QName
Data spouątěče	MQTM	TriggerData
Data uživatele	MQTM	UserData

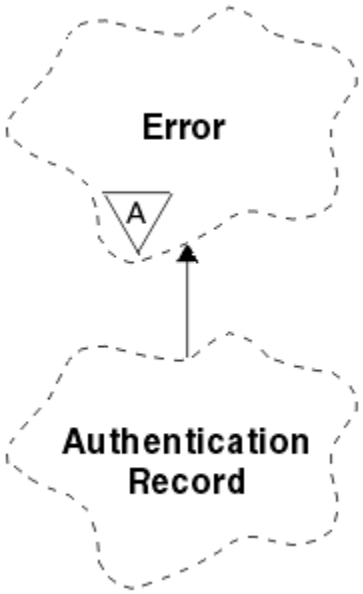
Křížový odkaz záhlaví ImqWork

Křížový odkaz na atributy, datové struktury a pole pro třídu C++ záznamu ImqAuthentication.

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 klientů SSL.



Obrázek 46. Třída záznamu ImqAuthentication

Další podrobnosti naleznete v popisu metody ImqQueueManager: :connect. Tato třída není k dispozici na platformě z/OS .

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 z prostředí *air*, 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*a nahradí existující data instance. To může zahrnovat přidělení závislého úložiště.

void clear (MQAIR * *pAir*);
Vymaže strukturu a uvolní závislé úložiště, na které odkazuje *pAir*.

Záznam ImqAuthenticationRecord * nextAuthenticationRecord () const;
Vrací **další záznam ověření**.

const ImqString & password () const;
Vrací **heslo**.

void setPassword (const ImqString & *heslo*);
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 setUserNázev (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**.

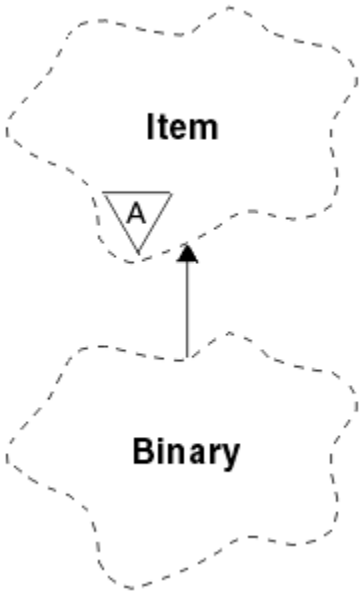
Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že nevylomí seznam autentizačních záznamů.

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 47. Třída ImqBinary

Atributy objektu

Data
Pole bajtů binárních dat. Počáteční hodnota je null.

Délka dat
Počet bajtů. Počáteční hodnota je nula.

data ukazatel
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 *length* bajtů z *data*.

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 *délka*, const char *pad* = 0);
Kopíruje až *délka* bajtů z **ukazatele dat** do *vyrovnávací paměti*. Pokud je **délka dat** nedostatečná, je zbývající prostor ve fondu *buffer* vyplněn na *pad* bajtů. Parametr *buffer* může být nula, pokud je *length* také nula. Hodnota *délka* nesmí být záporná. Pokud je úspěšný, vrací TRUE.

size_t dataLength() const ;
Vrací **datovou délku**.

ImqBoolean setDataLength(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 *délka*);
Kopíruje *délka* bajtů z *vyrovnávací paměti*. Pokud je úspěšný, vrací TRUE.

Metody objektů (chráněné)

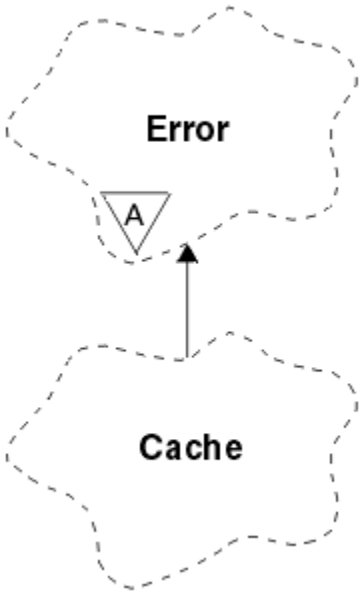
void clear();
Zmenšuje **délku dat** na nulu.

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 48. 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 1272.

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.

data ukazatel
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*** **);**
Kopíruje až **délka zprávy** bajtů dat z objektu *mezipaměť* na objekt. 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 ***posun*** **);**
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 ***byte-required*** **);**
Zajistí, že *byte-required* bude 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 ***délka*****, char * &** ***externí_vyrovnávací paměť*** **);**
Kopíruje *length* 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 pamětiza** 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 *délka*);
Identifikuje prázdnou vyrovnávací paměť uživatele, nastavení **ukazatele vyrovnávací paměti** tak, aby ukazovala na *externí-vyrovnávací paměť*, **délku vyrovnávací paměti** na *délka*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 je 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 do paměti spravované systémem spravované systémem (*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 *délka*);
Co se týče **useEmptyBuffer**, kromě toho, že **délka zprávy** je nastavena na *délka*. Pokud je úspěšný, vrací TRUE.

ImqBoolean write(const size_t *length*, const char * *external-buffer*);
Kopíruje *length* bajtů, z *external-buffer*do vyrovnávací paměti začínající na pozici **data pointer** . Po zkopírování dat se **posunutí dat** zvýší o *délka*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.

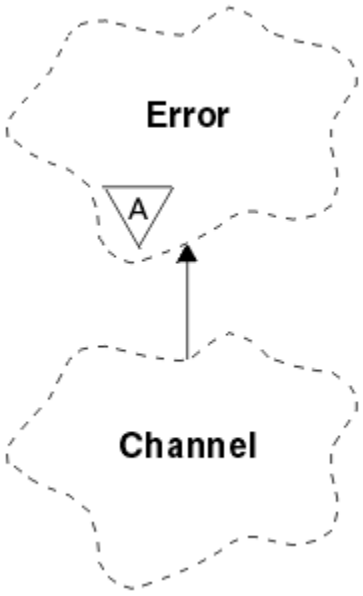
Má-li parametr **automatická vyrovnávací paměť** hodnotu TRUE, je zaručena adekvátní velikost paměti. V opačném případě nesmí být maximální hodnota **offset dat** větší než **délka vyrovnávací paměti**.

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 49. 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 podrobnosti viz popisy příkazů DEFINE CHANNEL a ALTER CHANNEL v produktu Odkaz na MQSC . Třída ImqChannel není podporována v systému z/OS.

- "Atributy objektu" na stránce 1294
- "Konstruktory" na stránce 1295
- "Metody objektů (veřejné)" na stránce 1295
- "Kódy příčin" na stránce 1299

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-prezenční interval
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.

příjem uživatelských procedur
Počet uživatelských procedur pro příjem. Počáteční hodnota je nula. Tento atribut je určen jen pro čtení.

přijímat názvy uživatelských procedur
Názvy uživatelských procedur pro příjem.

příjem uživatelských dat
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.

data uživatele 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.

názvy uživatelských procedur odeslání
Názvy uživatelských procedur odeslání.

odeslat uživatelská data
Data přidružená k uživatelským procedurám odeslání.

SSL CipherSpec
CipherSpec pro použití se zabezpečením SSL.

Typ ověření klienta SSL
Typ ověření klienta pro použití se zabezpečením SSL.

Název partnera SSL
Název partnera pro použití se zabezpečením SSL.

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 souboru *names*. Nastaví všechny *názvy* přesahující **počet výjezdu pro příjem** na prázdné řetězce. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setReceiveExitName(const char * *název* = 0);
Nastaví **názvy uživatelských procedur příjmu** na jediný *název*. *název* může být prázdný nebo 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 * *názvy* []);
Nastaví **názvy uživatelských procedur příjmu** 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 příjmu** 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 **receive user data** v *data*. Nastaví všechny *data* přesahující **počet uživatelských procedur pro příjem** 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* []);
Nastaví **příjem uživatelských dat** na *data*. Hodnota *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* []);
Nastaví **příjem uživatelských dat** na *data*. Hodnota *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 sendExitNames (const size_t *počet*, ImqString * *names* []);
Vrací kopie **názvů uživatelských procedur odeslání** v *names*. Nastaví všechny *názvy* 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* může být prázdný nebo 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 * *názvy* []);
Nastaví **názvy uživatelských procedur odeslání** na *names*. Jednotlivé hodnoty *names* nesmí být prázdné nebo mít hodnotu null. Nastaví **počet ukončení odeslání** na *počet*. 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 *počet*. 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 všechny *data* přesahující **počet uživatelských procedur odeslání** do řetězců 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* nemají hodnotu null, hodnota **počtu ukončení odeslání** musí být alespoň 1. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendUserData(const size_t *počet*, const char * *data* []);
Nastaví **odeslání uživatelských dat** na *data*. Hodnota *count* nesmí být větší než **počet ukončovacích procedur pro odeslání**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setSendUserData(const size_t *počet*, const ImqString * *data* []);
Nastaví **odeslání uživatelských dat** na *data*. Hodnota *count* nesmí být větší než **počet ukončovacích procedur pro odeslání**. Tato metoda vrací TRUE, je-li úspěšná.

Specifikace ImqString sslCipherSpecification () const;
Vrací specifikaci šifry SSL.

ImqBoolean setSslCipherSpecification(const char * *název* = 0);
Nastaví specifikaci šifry SSL. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG sslClientAuthentication () const;
Vrací typ ověřování klienta SSL.

ImqBoolean setSslClientAuthentication(const MQLONG *auth* = MQSCA_REQUIRED);
Nastaví typ ověřování klienta SSL. Tato metoda vrací TRUE, je-li úspěšná.

ImqString sslPeerName () const;
Vrátí název partnera SSL.

ImqBoolean setSslPeerName(const char * *název* = 0);
Nastaví název partnera SSL. 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**.

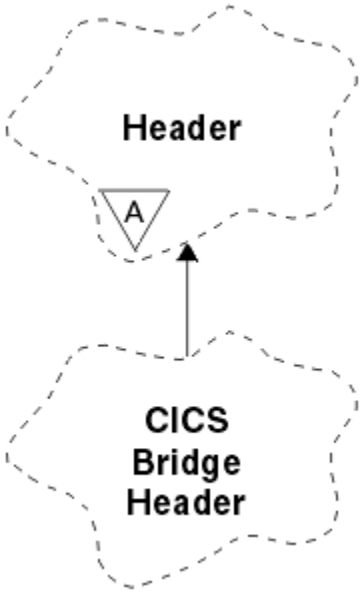
ImqBoolean setUserId (const char * *id* = 0);
Nastavuje **ID uživatele**. Tato metoda vrací TRUE, je-li úspěšná.

Kódy příčin

- CHYBA MQRC_DATA_LENGTH_ERROR
- CHYBA MQRC_ITEM_COUNT_ERROR
- MQRC_NULL_POINTER
- CHYBA MQRC_SOURCE_BUFFER_ERROR

ImqCICSBridgeTřída C++ záhlaví

Tato třída zapouzdřuje specifické vlastnosti datové struktury MQCIH.



Obrázek 50. ImqCICSBridgeTřída záhlaví

Objekty této třídy jsou používány aplikacemi, které odesílají zprávy do mostu CICS prostřednictvím produktu WebSphere MQ pro systém z/OS.

- "Atributy objektu" na stránce 1299
- "Konstruktory" na stránce 1302
- "Přetížené metody ImqItem" na stránce 1302
- "Metody objektů (veřejné)" na stránce 1302
- "Data objektu (chráněná)" na stránce 1304
- "Kódy příčin" na stránce 1304
- "Návratové kódy" na stránce 1305

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 614 na stránce 1305.

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 WebSphere 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 614 na stránce 1305.

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 produktu WebSphere 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 614 na stránce 1305.

návratový kód mostu
Návratový kód z mostu CICS . 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
Čas uvolnění zařízení mostu CICS .

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 produktu WebSphere 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 614 na stránce 1305.

Pokud **funkce** obsahuje název volání WebSphere MQ , jsou možné následující další hodnoty:

- MQCFUNC_MQCONN
- FUNKCE MQCFUNC_MQGET

- MQCFUNC_MQINQ
- MQCFUNC_NONE
- MQCFUNC_MQOPEN
- MQCFUNC_PUT
- MQCFUNC_MQPUT1

get interval čekání
Interval čekání pro volání MQGET vydané úlohou mostu CICS . 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

další identifikátor transakce
ID další transakce, která se má připojit. Pole musí mít délku MQ_TRANSACTION_ID_LENGTH.

délka výstupních dat
Délka dat COMMAREA. Počáteční hodnota je MQCODL_AS_INPUT.

formát odpovědi
Název formátu zprávy odpovědi. Počáteční hodnota je MQFMT_NONE s délkou MQ_FORMAT_LENGTH.

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.

Řídicí prvek UOW
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

ImqCICSBridgeHlavička ();
Výchozí konstruktor.

ImqCICSBridgeZáhlaví (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 & *záhlaví*);
Zkopíruje data instance z *header*, přičemž nahradí existující data instance.

Funkce MQLONG ADSDDescriptor () const;
Vrací kopii **deskriptoru ADS**.

void setADSDescriptor(const MQLONG *popisovač* = 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 dodána žádná *data* , resetuje se **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. Není-li dodána žádná *data* , resetuje **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. Není-li 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 dodána žádná *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**. *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** (odpověď na formát).

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. Není-li 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. Není-li 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

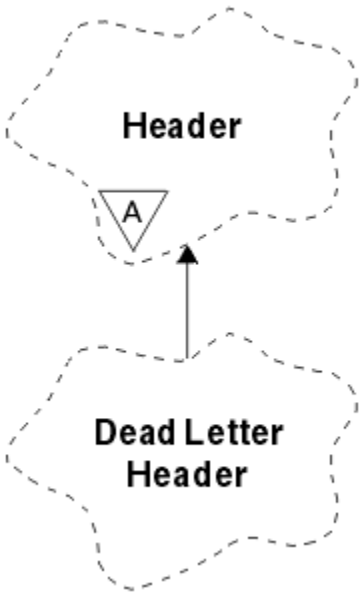
- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR
- MQRC_WRONG_VERSION

Návratové kódy

Tabulka 614. Návratové kódy třídy záhlaví ImqCICSBridge				
Návratový kód	**Funkce**	**CompCode**	**Příčina**	**Kód nestandardního konce**
MQCRC_OK				
CHYBA MQCRC_BRIDGE_ERROR			MQFB_CICS	
CHYBA MQCRC_MQ_API_ERROR	Název volání produktu WebSphere MQ	WebSphere MQ CompCode	WebSphere MQ Důvod	
MQCRC_BRIDGE_TIMEOUT	Název volání produktu WebSphere MQ	WebSphere MQ CompCode	WebSphere MQ Důvod	
CHYBA MQCRC_CICS_EXEC_ERROR	EIBFN CICS	CICS EIBRESP	CICS EIBRESP2	
MQCRC_SECURITY_ERROR	EIBFN CICS	CICS EIBRESP	CICS EIBRESP2	
MQCRC_PROGRAM_NOT_AVAILABLE	EIBFN CICS	CICS EIBRESP	CICS EIBRESP2	
MQCRC_TRANSID_NOT_AVAILABLE	EIBFN CICS	CICS EIBRESP	CICS EIBRESP2	
MQCRC_BRIDGE_ABEND				ABKÓD CICS
UKONČENÍ MQCRC_APPLICATION_ABEND				ABKÓD CICS

Třída C++ ImqDeadLetterHeader

Tato třída zapouzdřuje funkce datové struktury MQDLH.



Obrázek 51. 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

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. Nastavuje *msg* **format** na MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na stránce “Třída C++ ImqHeader” na stránce 1313 .

virtual ImqBoolean pasteIn(ImqMessage & *msg*);
Načte datovou strukturu MQDLH z vyrovnávací paměti zpráv.

Aby bylo úspěšné, ImqMessage **formátování** musí být MQFMT_DEAD_LETTER_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader na stránce “Třída C++ ImqHeader” na stránce 1313 .

Metody objektů (veřejné)

void operator = (const ImqDeadLetterHeader & *header*);
Zkopíruje data instance zkopírovaná z *header*, přičemž nahradí existující data instance.

MQLONG deadLetterReasonCode() const ;
Vrátí **deadový dopis s kódem příčiny**.

void setDeadLetterReasonCode(const MQLONG *reason*);
Nastavuje **kód příčiny smrtelného dopisu**.

ImqString destinationQueueManagerName() const ;
Vrátí **název cílového správce front**bez koncových mezer.

void setDestinationQueueManagerName(const char * *název*);
Nastaví **název cílového správce front**. Ořízne data delší než MQ_Q_MGR_NAME_LENGTH (48 znaků).

ImqString destinationQueueNázev() const ;
Vrací kopii **názvu cílové fronty**, která má být odstraněna z 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 souboru **put application name**, která byla odstraněna z případných koncových mezer.

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);
Nastavuje **typ aplikace put**.

ImqString putDate() const ;
Vrací kopii **data vložení**, která budou odstraněna z koncových mezer.

void setPutDate(const char * *date* = 0);
Nastavuje **datum vložení**. Zkrátí data delší než hodnota MQ_PUT_DATE_LENGTH (8 znaků).

ImqString putTime() const ;
Vrátí kopii souboru **put time**, která byla odstraněna z případných koncových mezer.

void setPutTime(const char * *čas* = 0);
Nastavuje **čas vložení**. Zkrátí data delší než hodnota MQ_PUT_TIME_LENGTH (8 znaků).

Data objektu (chráněná)

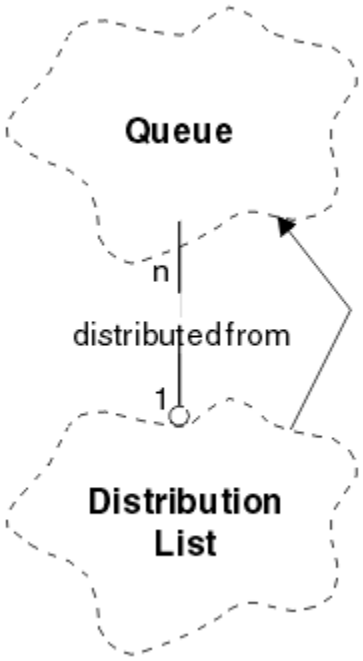
MQDLH *oqdlh*
Datová struktura MQDLH.

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 52. Třída seznamu ImqDistribution

- "Atributy objektu" na stránce 1308
- "Konstruktory" na stránce 1308
- "Metody objektů (veřejné)" na stránce 1308
- "Metody objektů (chráněné)" na stránce 1309

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 ImqDistribution(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.

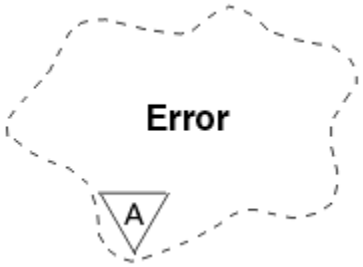
*** firstDistributedQueue() const ;**
Vrací **první distribuovanou frontu**.

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 53. Třída ImqError

- "Atributy objektu" na stránce 1309
- "Konstruktory" na stránce 1309
- "Metody objektů (veřejné)" na stránce 1309
- "Metody objektů (chráněné)" na stránce 1310
- "Kódy příčin" na stránce 1310

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 & *chyba*);
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqError & *error*);
Zkopíruje data instance z *error*, přičemž 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 *délka*);
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 *délka*);
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**.

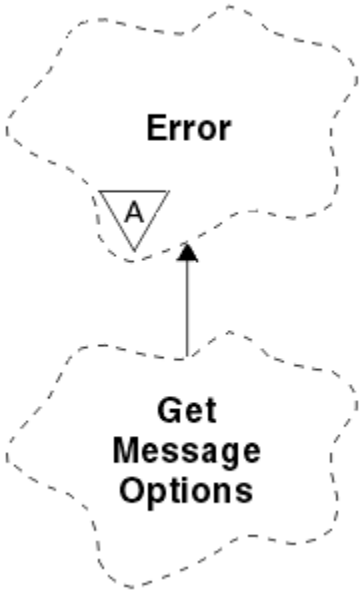
void setReasonCode(const MQLONG *kód* = 0);
Nastavuje **kód příčiny**.

Kódy příčin

- CHYBA MQRC_BUFFER_ERROR

Třída C++ ImqGetMessageOptions

Tato třída zapouzdřuje strukturu dat MQGMO



Obrázek 54. Třída ImqGetMessageOptions

- "Atributy objektu" na stránce 1310
- "Konstruktory" na stránce 1312
- "Metody objektů (veřejné)" na stránce 1312
- "Metody objektů (chráněné)" na stránce 1313
- "Data objektu (chráněná)" na stránce 1313
- "Kódy příčin" na stránce 1313

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, kterou metoda **get** třídy pozastaví a 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 **volby** neobsahují MQGMO_WAIT.

Konstruktory

ImqGetMessageOptions( );
Výchozí konstruktor.

ImqGetMessageOptions(const ImqGetMessageOptions & ***gmo*** **);**
Kopírovací konstruktor.

Metody objektů (veřejné)

void operator = (const ImqGetMessageOptions & ***gmo*** **);**
Zkopíruje data instance z *gmo*a nahradí existující data instance.

MQCHAR groupStatus() const ;
Vrátí **stav skupiny**.

void setGroupStatus(const MQCHAR ***stav*** **);**
Nastavuje **stav skupiny**.

MQLONG matchOptions() const ;
Vrací **volby shody**.

void setMatchOptions(const MQLONG ***volby*** **);**
Nastaví **volby shody**.

ImqBinary messageToken() const;
Vrací **token zprávy**.

ImqBoolean setMessageToken (const ImqBinary & ***token*** **);**
Nastaví **token zprávy**. **Délka dat** prvku *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* nenulový, musí adresovat 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.

MQLONG volby() const ;
Vrací **volby**.

void setOptions(const MQLONG ***volby*** **);**
Nastaví **volby**, včetně hodnoty **participace synchronizačního bodu** .

ImqString resolvedQueueNázev() const ;
Vrací kopii **vyřešeného názvu fronty**.

MQLONG returnedLength() const;
Vrací **vrácenou délku**.

MQCHAR segmentace() const ;
Vrátí hodnotu **segmentace**.

void setSegmentation(const MQCHAR ***hodnota*** **);**
Nastavuje **segmentaci**.

MQCHAR segmentStatus() const ;
Vrátí **stav segmentu**.

void setSegmentStav(const MQCHAR *stav*);
Nastaví **stav segmentu**.

ImqBoolean syncPointÚčast() const ;
Vrací hodnotu **Účast na synchronizačním bodu** , která má hodnotu TRUE, pokud **volby** zahrnují buď MQGMO_SYNCPOINT, nebo MQGMO_SYNCPOINT_IF_PERSISTENT.

void setSyncPointParticipation(const ImqBoolean *sync*);
Nastaví hodnotu **participace synchronizačního bodu** . Má-li volba *sync* hodnotu TRUE, změní **volby** hodnotu MQGMO_SYNCPOINT a vyloučí soubory MQGMO_NO_SYNCPOINT a MQGMO_SYNCPINT_IF_PERSISTENT. Pokud je volba *sync* FALSE, pozmění **volby** , aby zahrnuly MQGMO_NO_SYNCPOINT, a aby vyloučili jak MQGMO_SYNCPOINT, tak MQGMO_SYNCPINT_IF_PERSISTENT.

MQLONG waitInterval() const ;
Vrátí **interval čekání**.

void setWaitInterval(const MQLONG *interval*);
Nastavuje **čekací interval**.

Metody objektů (chráněné)

static void setVersionSupported(const MQLONG);
Nastavuje verzi **MQGMO** . Výchozí nastavení 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.

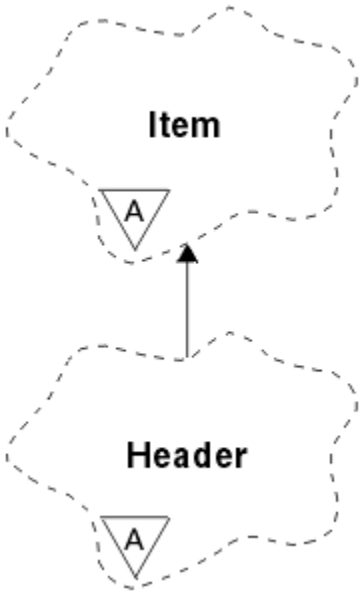
MQLONG *olVersion*
Číslo verze datové struktury MQGMO adresované pomocí *opgmo*.

Kódy příčin

- CHYBA MQRC_BINARY_DATA_LENGTH_ERROR

Třída C++ ImqHeader

Tato abstraktní třída zapouzdřuje společné funkce datové struktury MQDLH.



Obrázek 55. Třída ImqHeader

- "Atributy objektu" na stránce 1314
- "Konstruktory" na stránce 1314
- "Metody objektů (veřejné)" na stránce 1314

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 záhlaví ImqIMSBridge
- Objekt MQRMHF_LAST pro objekty třídy záhlaví ImqReference
- Objekt MQCIH_NONE pro objekty třídy záhlaví ImqCICSBridge
- 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 z *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.

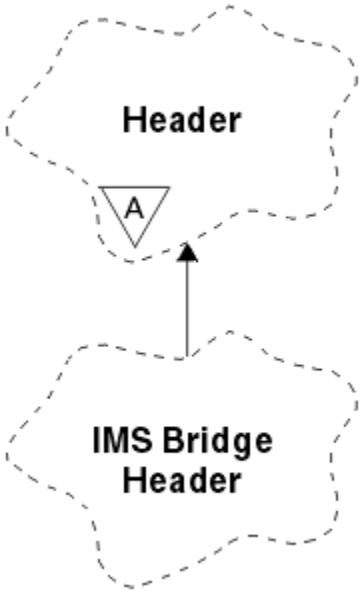
virtuální void setFormat(const char * *název* = 0);
Nastavuje **formát**vyplněný na 8 znaků s koncovými mezerami.

virtuální MQLONG headerFlags() const ;
Vrací parametry **header flags**.

virtuální void setHeaderFlags(const MQLONG *flags* = 0);
Nastaví **příznaky záhlaví**.

Třída C++ záhlaví ImqIMSBridge

Tato třída zapouzdřuje funkce datové struktury MQIIH.



Obrázek 56. ImqIMSBridgeTřída záhlaví

Objekty této třídy jsou používány aplikacemi, které odesílají zprávy na most IMS prostřednictvím produktu WebSphere MQ pro systém z/OS.

Poznámka: Hodnoty ImqHeader **znaková sada** a **kódování** musí mít výchozí hodnoty a nesmí být nastaveny na žádné jiné hodnoty.

- "Atributy objektu" na stránce 1316
- "Konstruktory" na stránce 1316
- "Přetížené metody ImqItem" na stránce 1316
- "Metody objektů (veřejné)" na stránce 1316
- "Data objektu (chráněná)" na stránce 1317
- "Kódy příčin" na stránce 1317

Atributy objektu

ověřovatel
Heslo RACF nebo přístupový lístek, délka MQ_AUTHENTICATOR_LENGTH. Počáteční hodnota je MQIAUT_NONE.

potvrdit režim
Režim vázaného zpracování. Další informace o režimech vázaného zpracování 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 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

ImqIMSBridgeHlavička ();
Výchozí konstruktor.

ImqIMSBridgeZáhlaví (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* **format** 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, **kódování** objektu *msg* musí být MQENC_NATIVE. Načtěte zprávy s MQGMO_CONVERT do MQENC_NATIVE.

Aby byla úspěšná, musí být ImqMessage **format** MQFMT_IMS.

Další podrobnosti naleznete v popisu metody nadřízené třídy.

Metody objektů (veřejné)

void operator = (const ImqIMSBridgeHeader & ***header*** **);**
Zkopíruje data instance z *header*, přičemž nahradí existující data instance.

ImqString Ověřovatel() const ;
Vrací kopii **ověřovatele**doplněnou na délku MQ_AUTHENTICATOR_LENGTH na délku MQ_AUTHENTICATOR_LENGTH.

void setAuthenticator(const char * *název*);
Nastavuje **ověřovatele**.

MQCHAR commitMode() const ;
Vrátí **režim vázaného zpracování**.

void setCommitMode(const MQCHAR *režim*);
Nastaví **režim vázaného zpracování**.

ImqString logicalTerminalPotlačit() const ;
Vrací kopii **logického terminálu přepisu**.

void setLogicalTerminalOverride(const char * *override*);
Nastaví **přepis logického terminálu**.

ImqString messageFormatServicesMapName() 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**, který je doplněn o koncové mezery až po délku MQ_FORMAT_LENGTH.

void setReplyToFormat(const char * *format*);
Nastavuje **formát odpovědi**, který je vyplněn koncovými mezerami na délku MQ_FORMAT_LENGTH.

MQCHAR securityScope() const ;
Vrací **rozsah zabezpečení**.

void setSecurityScope(const MQCHAR *rozsah*);
Nastaví **rozsah zabezpečení**.

ImqBinary transactionInstanceID() const ;
Vrací kopii **ID instance transakce**.

ImqBoolean setTransactionInstanceId(const ImqBinary & *id*);
Nastaví **ID instance transakce**. **Délka dat** prvku *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**. Hodnota *id* může být nula, což je stejné jako určení hodnoty MQITII_NONE. Je-li parametr *id* nenulový, musí adresovat proměnné 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 *oqiih*
Datová struktura MQIIH.

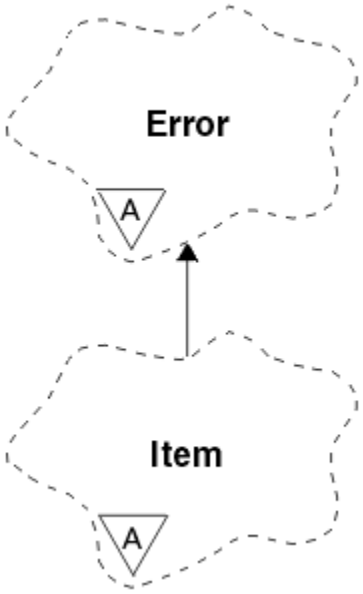
Kódy příčin

- 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 57. 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 1318
- "Konstruktory" na stránce 1318
- "Metody třídy (veřejné)" na stránce 1318
- "Metody objektů (veřejné)" na stránce 1319
- "Kódy příčin" na stránce 1319

Atributy objektu

struktura id
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*, nahradí se 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 (sub) 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.

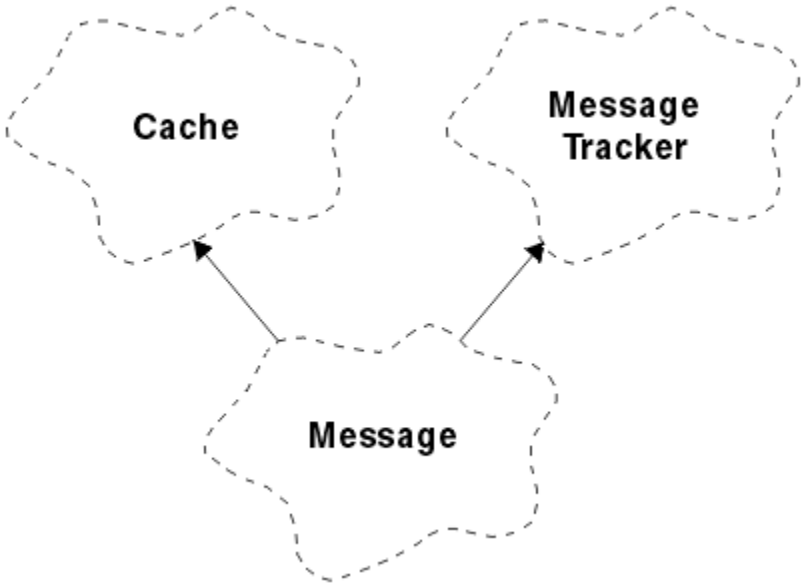
Potlačením této metody lze pracovat se specifickou podtřídou.

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 58. Třída ImqMessage

- "Atributy objektu" na stránce 1320
- "Konstruktory" na stránce 1324
- "Metody objektů (veřejné)" na stránce 1324
- "Metody objektů (chráněné)" na stránce 1326
- "Data objektu (chráněná)" na stránce 1326

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 886.

kódování
Kódování dat zprávy v počítači. Počáteční hodnota je MQENC_NATIVE.

Vypršení
Množství závislé na čase, které řídí, jak dlouho produkt WebSphere MQ uchovává nenačtenou zprávu, než ji bude zahozit. 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 397 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 408 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 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 412 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.

jméno 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 podporovány v produktu WebSphere MQ pro z/OS.

Sequence Number, 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 identifikátoru 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 ;**
Vrací TRUE, je-li **formát** stejný jako *format-to-test*.

void setFormat(const char * ***název*** **= 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 *perzistence*);
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 * *čas* = 0);
Nastavuje **čas vložení**.

ImqBoolean readItem(ImqItem & *item*);
Čte do objektu *položka* 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 * *název* = 0);
Nastaví **název správce front pro odpověď**.

ImqString replyToQueueName() const ;
Vrací kopii **názvu fronty pro odpovědi**.

void setReplyToQueueName(const char * *název* = 0);
Nastaví **název fronty pro odpovědi**.

MQLONG sestava() const ;
Vrátí **sestavu**.

void setReport(const MQLONG *sestava*);
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**.

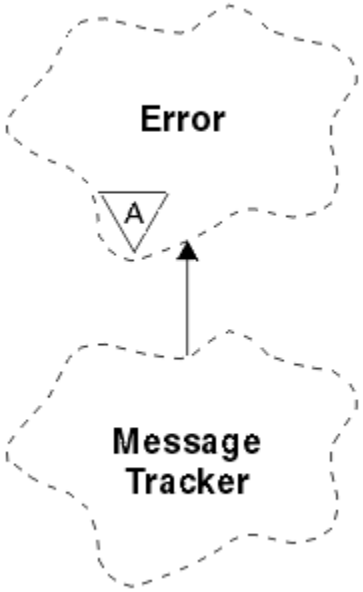
Data objektu (chráněná)

MQMD1 *oqmd*
(Pouze produkt WebSphere MQ for z/OS .) Datová struktura MQMD.

MQMD2 *oqmd*
(Platformy jiné než z/OS.) Datová struktura MQMD.

Třída C++ produktu ImqMessageTracker

Tato třída zapouzdřuje tyto atributy objektu ImqMessage nebo ImqQueue , který může být přidružen k jednomu z objektů.



Obrázek 59. Třída produktu ImqMessageTracker

Tato třída se vztahuje k voláním MQI uvedeným v části “Křížový odkaz ImqMessageTracker” na stránce 1277.

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 394 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 *tracker*a nahradí existující data instance.

ImqBinary accountingToken() const ;
Vrací kopii **účtovacího tokenu**.

ImqBoolean setAccountingToken(const ImqBinary & *token*);
Nastaví **účtovací token**. **Délka dat** *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**. *token* může být nula, což je stejné jako uvedení hodnoty MQACT_NONE. Je-li parametr *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**. Hodnota *id* může být nula, což je stejné jako určení hodnoty MQCI_NONE. Je-li parametr *id* nenulový, 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**. Hodnota *id* může být nula, což je stejné jako určení hodnoty MQGI_NONE. Je-li parametr *id* nenulový, 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.

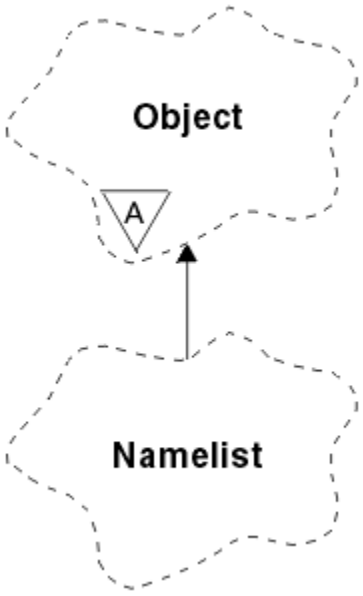
void setMessageId(const MQBYTE24 ***id*** **= 0);**
Nastavuje **ID zprávy**. Hodnota *id* může být nula, což je stejné jako uvedení MQMI_NONE. Je-li parametr *id* nenulový, musí adresovat MQ_MSG_ID_LENGTH bajtů binárních dat. Při použití předdefinovaných hodnot, jako je například MQMI_NONE, může být 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 60. Třída ImqNamelist

Tato třída se vztahuje k voláním MQI uvedeným v části “Křížový odkaz ImqNamelist” na stránce 1277.

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 *seznam*, 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.

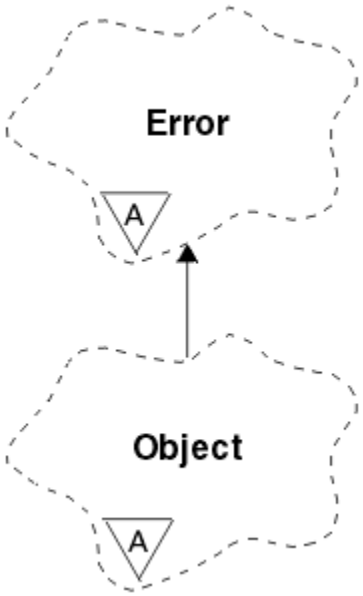
ImqString namelistName (const MQLONG ***index*** **);**
Vrátí jeden z názvů **názvů seznamů názvů** podle indexu založeného na nula bez určení možných chyb.

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 61. Třída ImqObject

Tato třída se vztahuje k voláním MQI uvedeným v části “Křížový odkaz ImqObject” na stránce 1277.

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.

zavřít volby
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 ji se **názvem správce front** , který identifikuje správce front (pravděpodobně vzdáleného) pro pojmenovanou frontu.

Popis
Popisný název (maximálně 64 znaků) správce front, fronty, seznamu názvů nebo procesu. Tento atribut je určen jen pro čtení.

Jméno
Jméno (maximálně 48 znaků) správce front, fronty, seznamu názvů nebo procesu. Počáteční hodnota je řetězec s hodnotou null. Název modelové fronty se změní po **otevření** 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
Jedná se o další objekt této třídy, v žádném konkrétním pořadí, který má stejný **odkaz 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 **volby otevření** a nepoužívejte metodu **open** . Produkt WebSphere MQ automaticky upraví **volby otevření** a automaticky se otevře, znovu otevře a zavře objekty podle potřeby. To může vést ke zbytečným operacím opětovného otevření, protože produkt WebSphere MQ používá metodu **openFor** , a to přidává **volby otevření** přírůstkově.
2. Před použitím metod, které vedou k volání MQI (viz “Křížový odkaz C++ a MQI” na stránce 1271), nastavte volbu **volby otevření** . 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).

   Použijete-li metodu **open** , ***musíte*** zajistit, aby **volby otevření** byly vhodné první. Použití metody **open** však není povinné; produkt WebSphere 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 voleb **setOpenOptions**(*lOpenOptions*) následováno příkazem **open**() nebo **openFor**(*lRequiredOpenOption*).

Poznámka:

1. Během metody **open** pro distribuční seznam je funkce MQOO_INQUIRE nahrazena MQOO_INQUIRE, protože v tomto okamžiku je jedinou platnou volbou **open option** hodnota 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 **vyřešený název správce front** a **vyřešený název fronty** 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, který 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 & *objekt*);
Kopírovací konstruktor. Hodnota **open status** bude FALSE.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrací **chování**.

void setBehavior(const MQLONG *chování* = 0);
Nastavuje **chování**.

Metody objektů (veřejné)

void operator = (const ImqObject & *objekt*);
Provede zavření, je-li to nezbytné, a zkopíruje data instance z *objektu*. Hodnota **open status** bude FALSE.

ImqBoolean alterationDate(ImqString & ***datum*** **);**
Poskytuje kopii **data změny**. Pokud je úspěšný, vrací TRUE.

ImqString alterationDate( );
Vrací **datum změny** bez uvedení možných chyb.

ImqBoolean alterationTime(ImqString & ***čas*** **);**
Poskytuje kopii **času změny**. Pokud je úspěšný, vrací TRUE.

ImqString alterationTime( );
Vrátí hodnotu **alterace time** bez uvedení možných chyb.

ImqString alternateUserId() const ;
Vrací kopii **alternativního ID uživatele**.

ImqBoolean setAlternateUserId(const char * ***id*** **);**
Nastaví **alternativní ID uživatele**. **Alternativní ID uživatele** lze nastavit pouze v případě, že hodnota **open status** je FALSE. Tato metoda vrací TRUE, je-li úspěšná.

ImqBinary alternateSecurityID () const;
Vrací kopii **alternativního ID zabezpečení.**

ImqBoolean setAlternateSecurityId(const ImqBinary & ***token*** **);**
Nastaví **alternativní ID zabezpečení**. **Alternativní ID zabezpečení** lze nastavit pouze tehdy, je-li **otevřený stav** FALSE. Délka dat *token* musí být buď nula, nebo MQ_SECURITY_ID_LENGTH. Pokud je úspěšný, vrací TRUE.

ImqBoolean setAlternateSecurityId(const MQBYTE* ***token*** **= 0);**
Nastaví **alternativní ID zabezpečení**. *token* může být nula, což je stejné jako určení MQSID_NONE. Je-li *token* nenulový, musí adresovat 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);**
Nastaví **alternativní ID zabezpečení**.

ImqBoolean close();
Nastaví **otevřený stav** na FALSE. Pokud je úspěšný, vrací TRUE.

MQLONG closeOptions() const ;
Vrací **volby zavření**.

void setCloseOptions(const MQLONG ***volby*** **);**
Nastavuje **volby zavření**.

Správce ImqQueueManager * connectionReference() const ;
Vrací **odkaz na připojení**.

void setConnectionReference(ImqQueueManager & ***manager*** **);**
Nastaví **odkaz na připojení**.

void setConnectionReference(ImqQueueManager * ***manager*** **= 0);**
Nastaví **odkaz na připojení**.

virtuální ImqBoolean popis(ImqString & ***popis*** **) = 0;**
Poskytuje kopii **popisu**. Pokud je úspěšný, vrací TRUE.

ImqString popis();
Vrací kopii **popisu** bez uvedení možných chyb.

virtual ImqBoolean název(ImqString & ***název*** **);**
Poskytuje kopii **názvu**. Pokud je úspěšný, vrací TRUE.

ImqString název();
Vrací kopii souboru **název** bez uvedení možných chyb.

ImqBoolean setName(const char * *název* = 0);
Nastavuje **název**. **Název** lze nastavit pouze tehdy, je-li **otevřený stav** FALSE, a u správce ImqQueueManager, zatímco **stav připojení** je FALSE. Pokud je úspěšný, vrací TRUE.

ImqObject * nextManagedObject() const ;
Vrátí **další spravovaný objekt**.

ImqBoolean open();
Změní **stav otevření** na TRUE otevřením objektu podle potřeby, pomocí dalších atributů **volby otevření** a **jména**. Tato metoda používá **odkaz na připojení** informace a metodu **připojení** správce ImqQueue, pokud je to nutné, aby se zajistilo, že správce ImqQueue **stav připojení** je TRUE. Vrací **otevřený stav**.

ImqBoolean openFor(const MQLONG *required-options* = 0);
Pokusí se zajistit, aby objekt byl otevřen s **otevřenými volbami**nebo **otevřenými volbami** , které zaručují chování implikované hodnotou parametru *požadované volby* .

Je-li *požadované-volby* nula, vstup je povinný a všechny vstupní volby postačuje. Takže, pokud **volby otevření** 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 nastavena v **otevřených volbách**.

Je-li parametr *required-options* nenulový, požadované volby se přidají do **voleb otevření**; pokud je *požadované-volby* některou z těchto voleb, ostatní se resetují.

Pokud se některá z **otevřených voleb** změní a objekt je již otevřen, objekt se zavře dočasně a znovu se otevře, aby bylo možné upravit **volby otevření**.

Pokud je úspěšný, vrací TRUE. Úspěch označuje, že objekt je otevřený s příslušnými volbami.

MQLONG openOptions() const ;
Vrací **volby otevření**.

ImqBoolean setOpenVolby(const MQLONG *volby*);
Nastaví **volby otevření**. **Volby otevření** lze nastavit pouze tehdy, když je **otevřený stav** FALSE. Pokud je úspěšný, vrací TRUE.

ImqBoolean openStatus() const ;
Vrátí **stav otevření**.

ImqObject * previousManagedObject() const ;
Vrací **předchozí spravovaný objekt**.

Identifikátor ImqBoolean queueManagerIdentifier (ImqString & *id*);
Poskytuje kopii identifikátoru **identifikátor správce front**. Pokud je úspěšný, vrací TRUE.

ImqString queueManagerIdentifier ();
Vrací **identifikátor správce front** bez uvedení možných chyb.

Metody objektů (chráněné)

virtual 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 nula, pokud neexistuje žádná **odkaz na připojení** , nebo pokud není správce připojen.

ImqBoolean dotázat se(const MQLONG *int-attr*, MQLONG & *hodnota*);
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* požadováno více než jedné hodnoty, kde jsou hodnoty konzistentní s ostatními pro instant, WebSphere 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*);
Nastaví celočíselný atribut produktu WebSphere MQ .

ImqBoolean set(const MQLONG *char-attr*, const char * *buffer*, const size_t *required-length*);
Nastaví znakový atribut produktu WebSphere MQ .

void setNextManagedObject(const ImqObject * *objekt* = 0);
Nastaví **další spravovaný objekt**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že se seznam spravovaných objektů nerozbije.

void setPreviousManagedObject(const ImqObject * *objekt* = 0);
Nastaví **předchozí spravovaný objekt**.

Upozornění: Tuto funkci použijte pouze v případě, že jste si jisti, že se seznam spravovaných objektů nerozbije.

Data objektu (chráněná)

MQHOBJ *ohobj*
Ovladač objektu WebSphere MQ (platný pouze, pokud 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 *omqod* lze přistupovat.

MQOD_VERSION_2
Ke všem ostatním polím v *omqod* lze přistupovat.

MQOD_VERSION_3
omqod.pmqod je ukazatel na dynamicky alokovaný, větší, MQOD. Žádná jiná pole v parametru *omqod* nejsou přístupná. Ke všem polím adresovaným pomocí *omqod.pmqod* lze přistupovat.

Poznámka: *omqod.pmqod.Version* může být menší než *omqod.Version*, což znamená, že klient WebSphere MQ MQI má více funkcí než server WebSphere MQ .

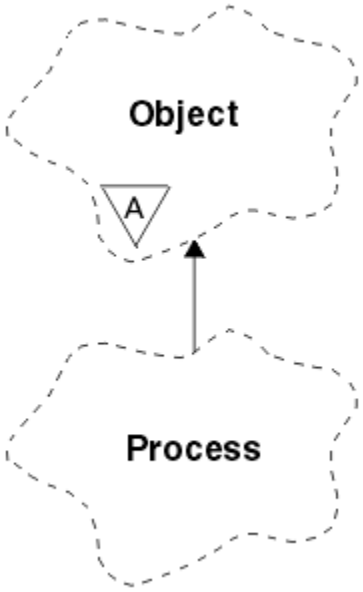
Kódy příčin

- 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 WebSphere MQ typu MQOT_PROCESS), který může být spuštěn monitorem spouštěčů.



Obrázek 62. Třída ImqProcess

- "Atributy objektu" na stránce 1336
- "Konstruktory" na stránce 1336
- "Metody objektů (veřejné)" na stránce 1337

Atributy objektu

Application ID
Identita aplikačního procesu. Tento atribut je určen jen pro čtení.

Typ aplikace
Typ procesu aplikace. Tento atribut je určen jen pro čtení.

Data prostředí
Informace o prostředí pro proces. Tento atribut je určen jen pro čtení.

Data uživatele
Uživatelská data pro proces. Tento atribut je určen jen pro čtení.

Konstruktory

ImqProcess( );
Výchozí konstruktor.

ImqProcess(const ImqProcess & ***proces*** **);**
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 *process*. 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.

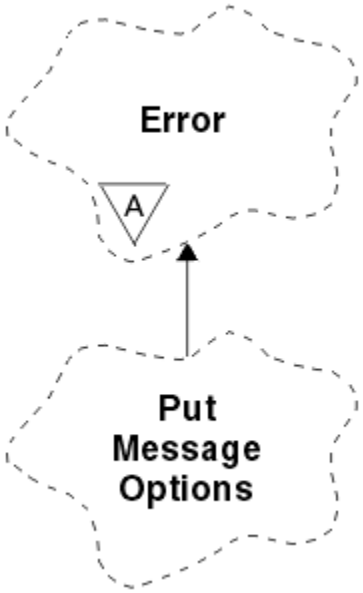
ImqString environmentData( );
Vrací **data prostředí** bez uvedení možných chyb.

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 63. Třída ImqPutMessageOptions

- "Atributy objektu" na stránce 1338
- "Konstruktory" na stránce 1338
- "Metody objektů (veřejné)" na stránce 1338
- "Data objektu (chráněná)" na stránce 1339
- "Kódy příčin" na stránce 1339

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 sledovače ImqMessagejsou převzaty z objektu ImqMessage pro libovolné pole, které *není* uvedeno.

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 ;
Vrací **odkaz kontextu**.

void setContextReference(const ImqQueue & *queue*);
Nastaví **odkaz na kontext**.

void setContextReference(const ImqQueue * *fronta* = 0);
Nastaví **odkaz na kontext**.

MQLONG volby() const ;
Vrací **volby**.

void setOptions(const MQLONG *volby*);
Nastaví **volby**, včetně hodnoty **participace synchronizačního bodu** .

MQLONG recordFields() const ;
Vrátí **pole záznamu**.

void setRecordPole(const MQLONG *pole*);
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 **participace synchronizačního bodu** , která má hodnotu TRUE, pokud volby **volby** zahrnují MQPMO_SYNCPOINT.

void setSyncPointParticipation(const ImqBoolean *sync*);
Nastaví hodnotu **participace synchronizačního bodu** . Má-li parametr *synchronizace* hodnotu TRUE, jsou volby **volby** upraveny tak, aby zahrnovaly MQPMO_SYNCPOINT a aby byly vyloučeny MQPMO_NO_SYNCPOINT. Je-li položka *sync* FALSE, volby **options** jsou upraveny tak, aby zahrnovaly MQPMO_NO_SYNCPOINT a aby vyloučily MQPMO_SYNCPOINT.

Data objektu (chráněná)

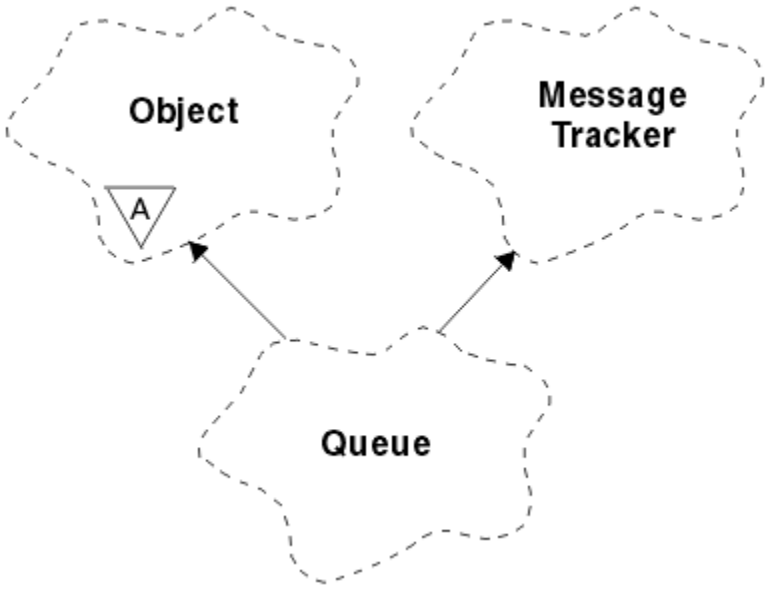
MQPMO *omqpmo*
Datová struktura MQPMO.

Kódy příčin

- MQRC_STORAGE_NOT_AVAILABLE

Třída C++ ImqQueue

Tato třída zapouzdřuje frontu zpráv (objekt WebSphere MQ typu MQOT_Q).



Obrázek 64. Třída ImqQueue

Tato třída se vztahuje k voláním MQI uvedeným v části Tabulka 612 na stránce 1279.

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.

hloubka vysoké události
Řídicí atribut pro vysoké události hloubky fronty. Tento atribut je určen jen pro čtení.

hloubka horní meze
Horní mez hloubky fronty. Tento atribut je určen jen pro čtení.

hloubka nízké události
Řídicí atribut pro události nízké hloubky fronty. Tento atribut je určen jen pro čtení.

hloubka dolní meze
Dolní mez hloubky fronty. Tento atribut je určen jen pro čtení.

maximum hloubky maximum
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 Windows, UNIXa Linux .

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.

inhibovat získat
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.

Název 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.

Rozsah
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 může být sdílená. Tento atribut je určen jen pro čtení.

paměťová třída
Paměťová třída. Tento atribut je určen jen pro čtení.

Jméno přenosové fronty
Název přenosové fronty. Tento atribut je určen jen pro čtení.

Řízení spouštěče
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 & *název*);
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 & *název*);
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 & ***čas*** **);**
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 & ***priorita*** **);**
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 & ***podpora*** **);**
Poskytuje kopii hodnoty **distribučních seznamů** . Pokud je úspěšný, vrací TRUE.

MQLONG distributionLists();
Vrátí hodnotu **distribučních seznamů** bez uvedení možných chyb.

ImqBoolean setDistributionLists(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 uvedené volby *options*. 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* vyrovnávací paměť ImqCache **automatická vyrovnávací paměť**, 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 indikováno přepsání *velikost-vyrovnávací paměti* . Pokud objekt *msg* používá ImqCache **automatic buffer**, je metoda **resizeBuffer** vyvolána na objektu *msg* před načtením zprávy a vyrovnávací paměť se dále nezvětšuje tak, aby obsáhlla větší zprávu.

ImqBoolean get(ImqMessage & ***msg*****, const size_t** ***velikost-vyrovnávací paměti*** **);**
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 & ***počet*** **);**
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 & ***název*** **);**
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, že ImqObject **volby otevření** zahrnují MQOO_OUTPUT, a (pokud *pmo* **volby** zahrnují jakékoli hodnoty MQPMO_PASS_ALLITY_CONTEXT, MQPMO_PASS_ALL_CONTEXT, MQPMO_SET_IDENTITY_CONTEXT nebo MQPMO_SET_ALL_CONTEXT), které odpovídají hodnotám MQOO_ * _CONTEXT.

Tato metoda vrací TRUE, je-li úspěšná.

Poznámka: Pokud *pmo* obsahuje **odkaz na kontext**, 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 & ***název*** **);**
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 &*název*);
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 &*název*);
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 serviceIntervalEvent(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 & *priorita*);
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 použití(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.

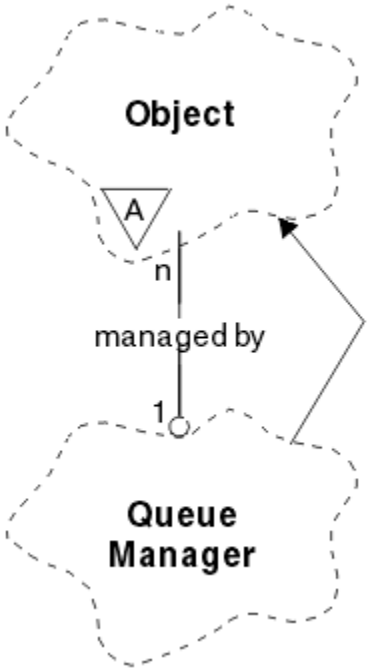
Kódy příčin

- 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 WebSphere MQ typu MQOT_Q_MGR).



Obrázek 65. Třída správce ImqQueueManager

Tato třída se vztahuje k voláním MQI uvedeným v části “Křížový odkaz správce ImqQueue” na stránce 1281. 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 (backout). Tento atribut se vzájemně vylučuje s hodnotou IMQ_EXPL_DISC_COMMIT.

IMQ_EXPL_DISC_COMMIT (1L)
Explicitní volání metody **disconnect** 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 **disconnect** , které se může vyskytnout během zničení objektu, znamená vrácení zpět. Tento atribut se vzájemně vylučuje s hodnotou IMQ_IMPL_DISC_COMMIT.

IMQ_IMPL_DISC_COMMIT (4L)
Implicitní volání metody **disconnect** , které se může vyskytnout během zničení objektu, implikuje potvrzení (výchozí nastavení). Tento atribut se vzájemně vylučuje s IMQ_IMPL_DISC_BACOUT.

V aplikacích WebSphere MQ V7.0 a vyšších jsou aplikace v jazyku C + +, které využívají implicitní připojení, nutné zadat parametr 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 metodu **begin** . Počáteční hodnota je MQBO_NONE.

událost mostu
Zda se generují události mostu IMS Bridge. 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 WebSphere 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 **connect** . 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 by měl modul listener TCP, který zpracovává příchozí přenosy pro skupinu sdílení front, připojit 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 objekt ImqObject **connection reference** adresuje tento objekt. 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) má být použit 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 klienta WebSphere MQ MQI.

počet resetování klíče
Počet nezašifrovaných bajtů odeslaných a přijatých v rámci konverzace SSL před opětovným vyjednávanstvím tajného klíče. 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 produktu WebSphere 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 na **časový limit příjmu**. 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 být operace MQOPEN sdílené fronty, ve které je název ObjectQMgrjiného správce front ve skupině sdílení front, vyřešena na otevření sdílené fronty v lokálním správci front. 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 WebSphere 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.

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 **typ 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 **název tcp** nebo může být svázán s libovolnou vybranou adresou 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 UNIX and Linux je to skutečné ID uživatele aplikace. Na platformách Windowsje ID uživatele aplikace.

Konstruktory

Správce ImqQueueManager ();
Výchozí konstruktor.

Manažer ImqQueueManager (const ImqQueueManager & ***manager*** **);**
Kopírovací konstruktor. Hodnota **stav připojení** bude FALSE.

ImqQueueManager (const char * ***název*** **);**
Nastaví objekt ImqObject **název** na *název*.

Destruktory

Když je objekt správce ImqQueuezlikvidován, je automaticky odpojen.

Metody třídy (veřejné)

statické chování MQLONG ();
Vrací **chování**.

void setBehavior(const MQLONG ***chování*** **= 0);**
Nastavuje **chování**.

Metody objektů (veřejné)

neobsazený operátor = (const ImqQueueManager & ***mgr*** **);**
Odpojí se, pokud je to nezbytné, a kopíruje data instance z *mgr*. Hodnota **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. Volba **begin options** ovlivňuje 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 **začátek voleb**.

void setBeginOptions (const MQLONG ***volby*** **= MQBO_NONE);**
Nastaví **volby začátku**.

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 & ***hodnota*** **);**
Poskytuje kopii hodnoty **automatické definice kanálu** . Pokud je úspěšný, vrací TRUE.

MQLONG channelAutoDefinition ();
Vrací 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 & ***název*** **);**
Poskytuje kopii pro název **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 se *pchannel* na 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 souboru **znakové sady**bez uvedení 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 jakýchkoli informací o možných chybách.

ImqBoolean clusterWorkloadExit (ImqString & ***název*** **);**
Poskytuje kopii **názvu uživatelské procedury pracovní zátěže klastru**. Pokud je úspěšný, vrací TRUE.

ImqString clusterWorkloadExit ();
Vrátí **název uživatelské procedury pracovní zátěže klastru** bez jakýchkoli informací o možných chybách.

ImqBoolean clusterWorkloadLength (MQLONG & ***délka*** **);**
Poskytuje kopii **délky pracovní zátěže klastru**. Pokud je úspěšný, vrací TRUE.

MQLONG clusterWorkloadLength ();
Vrátí **pracovní zátěž 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 příkazu **název vstupní fronty příkazů**. Pokud je úspěšný, vrací TRUE.

ImqString commandInputQueueName( );
Nezobrazí **název vstupní fronty příkazů** bez uvedení možných chyb.

ImqBoolean commandLevel(MQLONG & ***úroveň*** **);**
Poskytuje kopii **úrovně příkazu**. Pokud je úspěšný, vrací TRUE.

MQLONG commandLevel();
Vrátí **command level** 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 **název**, přičemž výchozí hodnotou je lokální správce front. Chcete-li se připojit ke specifickému správci front, použijte před připojením metodu ImqObject **setName** . Je-li zde **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. Informace o **odkazu na kanál** , které mají význam pouze pro připojení klienta, jsou pro připojení k serveru ignorována. Volby **connect options** 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;
Vrací **značku připojení**.

ImqBoolean setConnectionTag (const MQBYTE128 *značka* = 0);
Nastaví značku **connection tag**. Je-li značka *tag* nula, vymaže **příznak připojení**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean setConnectionZnačka (const ImqBinary & *značka*);
Nastaví značku **connection tag**. **Délka dat** *značky* musí být buď nula (k vyčištění **značky připojení**) nebo MQ_CONN_TAG_LENGTH. Tato metoda vrací TRUE, je-li úspěšná.

MQLONG connectOptions() const;
Vrací **volby připojení**.

void setConnectOptions (const MQLONG *volby* = MQCNO_NONE);
Nastaví **volby připojení**.

ImqBoolean connectionStatus() const;
Vrátí **stav připojení**.

ImqString cryptographicHardware ( );
Vrací **šifrovací hardware**.

ImqBoolean setCryptographicHardware (const char * *hardware* = 0);
Nastavuje **šifrovací hardware**. Tato metoda vrací TRUE, je-li úspěšná.

ImqBoolean deadLetterQueueName(ImqString & *název*);
Poskytuje kopii **názvu fronty nedoručených zpráv**. Pokud je úspěšný, vrací TRUE.

ImqString deadLetterQueueName( );
Vrátí kopii souboru **název fronty nedoručených zpráv**bez jakýchkoli informací o možných chybách.

ImqBoolean defaultTransmissionQueueName(ImqString & *název*);
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 je jejich **odkaz na připojení** oddělitelné. 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 **distribučních seznamů** . Pokud je úspěšný, vrací TRUE.

MQLONG distributionLists();
Vrátí hodnotu **distribučních seznamů** bez uvedení 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;
Vrací **první ověřovací záznam**.

void setFirstAuthenticationRecord (const ImqAuthenticationRecord * ***air*** **= 0);**
Nastaví **první záznam ověření**.

ImqObject * firstManagedObject () const;
Vrací **první spravovaný objekt**.

ImqBoolean inhibitEvent(MQLONG & ***událost*** **);**
Poskytuje kopii stavu povolení události **inhibit event**. Pokud je úspěšný, vrací TRUE.

MQLONG inhibitEvent();
Vrací stav povolení události **inhibit event** 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 ( );
Vrací **úložiště klíčů**.

ImqBoolean setKeyRepository (const char * ***úložiště*** **= 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 pro **lokální událost**. 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 popisovačů**. 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 ();
Nevrátí **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 & ***číslo*** **);**
Poskytuje kopii **maximálního počtu nepotvrzených zpráv**. Pokud je úspěšný, vrací TRUE.

MQLONG maximumUncommittedMessages ();
Vrátí **maximum nepotvrzených zpráv** 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 **platform**. Pokud je úspěšný, vrací TRUE.

MQLONG platform ();
Vrací **platformu** bez uvedení možných chyb.

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 události **vzdálená událost** bez uvedení možných chyb.

ImqBoolean repositoryName(ImqString & ***název*** **);**
Poskytuje kopii **názvu úložiště**. Pokud je úspěšný, vrací TRUE.

ImqString repositoryName( );
Vrací **název úložiště** bez uvedení možných chyb.

ImqBoolean repositoryNamelistName (ImqString & ***název*** **);**
Poskytuje kopii **názvu seznamu názvů úložiště**. Pokud je úspěšný, vrací TRUE.

ImqString repositoryNamelistnázev ();
Vrací kopii souboru **název 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-stop event**. Pokud je úspěšný, vrací TRUE.

MQLONG startStopEvent ();
Vrací stav povolení události **start-stop event** 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();
Vrací **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 * *objekt* **= 0);**
Nastaví **první spravovaný objekt**.

Data objektu (chráněná)

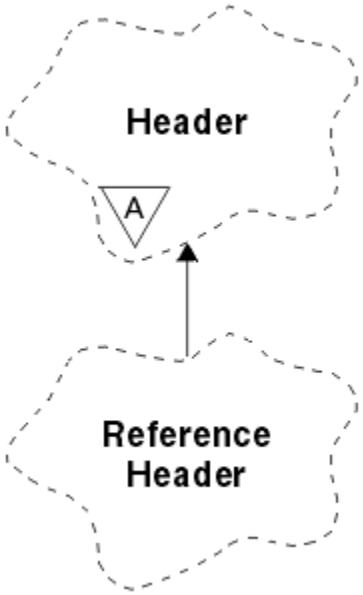
MQHCONN ***ohn ohn***
Popisovač připojení produktu WebSphere MQ (smysluplný pouze v případě, že je **stav připojení** TRUE).

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 66. Třída záhlaví ImqReference

Tato třída se vztahuje k voláním MQI uvedeným v části "Křížový odkaz záhlaví ImqReference" na stránce 1285.

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 pro **logický posun**s vysokou objednávkou. 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.

záhlaví ImqReference(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 *msg* **format** do MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader v příručce "Třída C++ ImqHeader" na stránce 1313 .

virtual ImqBoolean pasteIn(ImqMessage & ***msg*** **);**
Načte datovou strukturu MQRMH z vyrovnávací paměti zpráv.

Aby bylo úspěšné, ImqMessage **formátování** musí být MQFMT_REF_MSG_HEADER.

Další podrobnosti naleznete v popisu metody třídy ImqHeader v příručce "Třída C++ ImqHeader" na stránce 1313 .

Metody objektů (veřejné)

void operator = (const ImqReferenceHeader & *header*);
Zkopíruje data instance z *header*, přičemž nahradí existující data instance.

ImqString destinationEnvironment() const ;
Vrací kopii **cílového prostředí**.

void setDestinationEnvironment(const char * *prostředí* = 0);
Nastavuje **cílové prostředí**.

ImqString destinationName() const ;
Vrací kopii **cílového názvu**.

void setDestinationName(const char * *název* = 0);
Nastaví **název místa určení**.

ImqBinary instanceId() const ;
Vrací kopii **ID instance**.

ImqBoolean setInstanceId(const ImqBinary & *id*);
Nastavuje **ID instance**. Hodnota **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);
Nastavuje **ID instance**. Hodnota *id* může být nula, což je stejné jako určení hodnoty MQOII_NONE. Je-li parametr *id* nenulový, 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 offset**.

MQLONG logicalOffset2() const ;
Vrátí **logický posun 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 setSourceEnvironment(const char * *prostředí* = 0);
Nastavuje **zdrojové prostředí**.

ImqString sourceName() const ;
Vrací kopii **zdrojového názvu**.

void setSourceName(const char * *název* = 0);
Nastavuje **název zdroje**.

Data objektu (chráněná)

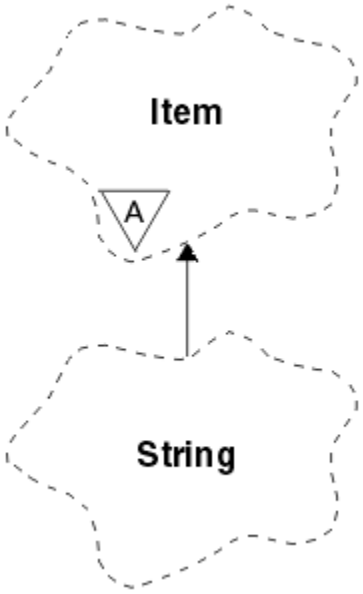
MQRMH *omqrmh*
Datová struktura MQRMH.

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 67. Třída ImqString

Použijte ImqString místo **char *** ve většině situací, kde parametr volá po **char ***.

- "Atributy objektu" na stránce 1368
- "Konstruktory" na stránce 1369
- "Metody třídy (veřejné)" na stránce 1369
- "Přetížené metody ImqItem" na stránce 1369
- "Metody objektů (veřejné)" na stránce 1369
- "Metody objektů (chráněné)" na stránce 1372
- "Kódy příčin" na stránce 1372

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 tvoří *c*.

ImqString(const char * *text*);
Znaky **znaky** jsou zkopírovány z *text*.

ImqString(const void * *buffer*, const size_t *délka*);
Kopíruje *délka* bajtů od *vyrovnávací paměti* a přiřadí je k **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 až *délka* bajtů z *zdroj-vyrovnávací paměť* do *destination-buffer*. Je-li počet znaků z *zdroj-vyrovnávací paměti* nedostatečný, zaplní zbývající prostor v *destination-buffer* znaky *pad* . *zdroj-vyrovnávací paměť* může být nula. Hodnota *destination-buffer* může být nula, pokud je *length* 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 * *zdroj-vyrovnávací paměť*, ImqError & *chybový_objekt*, const char *pad* = 0);
Kopíruje až *délka* bajtů z *zdroj-vyrovnávací paměť* do *destination-buffer*. Je-li počet znaků z *zdroj-vyrovnávací paměti* nedostatečný, zaplní zbývající prostor v *destination-buffer* znaky *pad* . *zdroj-vyrovnávací paměť* může být nula. Hodnota *destination-buffer* může být nula, pokud je *length* také nula. Jakékoli kódy chyb jsou nastaveny v *chybá-objekt*. 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. Nastavuje 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 možné 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** počínaje *ofset*. Je-li *délka* nula, vrátí zbytek **znaků**. Pokud kombinace *offset* a *length* nevytvoří odkaz v rámci **znaků**, vrátí prázdný řetězec ImqString.

void operator = (const ImqString & *řetězec*);
Zkopíruje data instance z *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 *text* 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 *číslo* k **znakům** po převodu na text.

Operátor ImqString + (const long *číslo*) const;
Vrátí výsledek připojení *číslo* k **znakům** po převodu na text.

neobsazený operátor + = (const char *c*);
Připojí *c* k **znakům**.

neobsazený operátor + = (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 & *řetězec*) const;
Porovnává **znaky** s prvky *řetězec* 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 *řetězec* pomocí metody **compare** . Výsledek je TRUE, je-li větší než a FALSE, je-li menší než nebo rovno.

Operátor ImqBoolean < = (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězec* 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 *řetězec* 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 *řetězec* pomocí metody **compare** . Vrací TRUE nebo FALSE.

ImqBoolean operator! = (const ImqString & *řetězec*) const;
Porovnává **znaky** s prvky *řetězec* 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);
Kopíruje až *délka* bajtů z **znaků** do *vyrovnávací paměti*. Pokud je počet znaků **znaků** nedostatečný, zaplní zbývající prostor ve *vyrovnávací paměti* znaky *pad* . Parametr *buffer* může být nula, pokud je *length* také nula. Pokud je úspěšný, vrací TRUE.

size_t copyOut(dlouhý & *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 všech znaků *c* , které jsou před znaky *token* . Pokud *c* není prázdný, odstraní znaky *c* , které přímo uspějí ve znacích *token* . Vrátí počet odstraněných znaků. Například řetězec zobrazený v následujícím příkladu lze rozdělit na tři tokeny 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 & *řetězec*);
Vyhledává přesnou shodu 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 & *řetězec*, size_t & *posun*);
Hledá přesnou shodu pro *řetězec* někde uvnitř **znaků** z offsetu *offset* dále. Má-li parametr *řetězec* hodnotu null, vrací hodnotu TRUE bez aktualizace *offset*. Pokud není nalezena žádná shoda, vrátí hodnotu FALSE (hodnota parametru *offset* mohla být zvýšena). Je-li nalezena shoda, vrátí hodnotu TRUE a aktualizuje *offset* na offset *řetězec* uvnitř **znaků**.

délka () velikost_t const;
Vrací **délku**.

ImqBoolean pasteIn(const double *číslo*, const char * *formát* = "%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(const long *počet*);
Připojí *číslo* k **znakům** po převodu na text. Pokud je úspěšný, vrací TRUE.

ImqBoolean pasteIn(const void * *buffer*, const size_t *délka*);
Připojí *délka* bajtů od *vyrovnávací paměti* k **znakům**a přidá poslední 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* ze **znaků** a vrátí číslo odebrané.

ImqString upperCase() const;
Vrací kopii souboru **znaků**na velká písmena.

Metody objektů (chráněné)

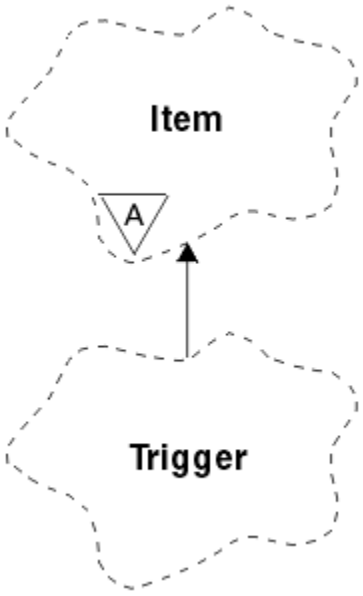
ImqBoolean assign(const ImqString & *řetězec*);
Ekvivalent ekvivalentní metodě **operator =** , ale nevirtuální. Pokud je úspěšný, vrací TRUE.

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 68. 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 WebSphere MQ budou spuštěny, když na ně budou čekat zprávy.

Příklad použití najdete v ukázkovém programu IMQSTRG.

- "Atributy objektu" na stránce 1373
- "Konstruktory" na stránce 1374
- "Přetížené metody ImqItem" na stránce 1374
- "Metody objektů (veřejné)" na stránce 1374
- "Data objektu (chráněná)" na stránce 1375
- "Kódy příčin" na stránce 1375

Atributy objektu

Application ID
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 & *spouštěč*);
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* **format** na MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem na adrese “Třída C++ ImqItem” na stránce 1318 .

virtual ImqBoolean pasteIn(ImqMessage & *msg*);
Načte datovou strukturu MQTM z vyrovnávací paměti zpráv.

Aby byla úspěšná, musí být ImqMessage **format** MQFMT_TRIGGER.

Další podrobnosti naleznete v popisu metody třídy ImqItem v příručce “Třída C++ ImqItem” na stránce 1318 .

Metody objektů (veřejné)

void operator = (const ImqTrigger & *trigger*);
Zkopíruje data instance z *triggerů*, 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*);
Nastavuje **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 procesu **název procesu**.

void setProcessName(const char * *název*);
Nastaví **název procesu**doplněný mezerami na 48 znaků.

ImqString queueName() const ;
Vrací kopii **názvu fronty**.

void setQueueName(const char * *název*);
Nastaví **název fronty**, výplň s mezerami na 48 znaků.

ImqString triggerData() const ;
Vrací kopii **aktivačních dat**.

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)

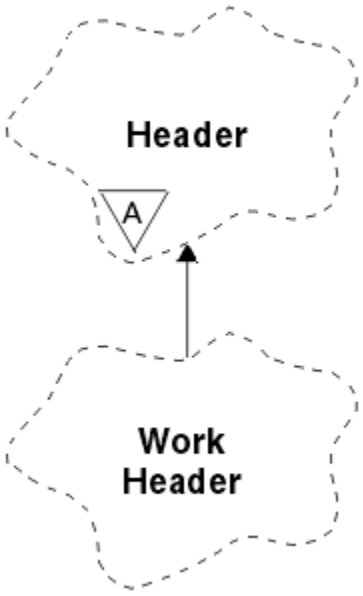
MQTM *omqtm*
Datová struktura MQTM.

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 69. Třída záhlaví ImqWork

Objekty této třídy používají aplikace, které vkládají zprávy do fronty spravované serverem z/OS Workload Manager.

- "Atributy objektu" na stránce 1376
- "Konstruktory" na stránce 1376
- "Přetížené metody ImqItem" na stránce 1376
- "Metody objektů (veřejné)" na stránce 1377
- "Data objektu (chráněná)" na stránce 1377
- "Kódy příčin" na stránce 1377

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, dále bude přesouvat existující data zprávy a nastaví *msg* **format** na 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 z *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 parametr *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 setServiceName (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

Třídy IBM WebSphere MQ pro knihovny Java

Umístění tříd produktu IBM WebSphere MQ pro knihovny Java se liší v závislosti na platformě. Uveďte toto umístění, když spustíte aplikaci.

Chcete-li zadat umístění knihoven JNI (Java Native Interface), spusťte aplikaci pomocí příkazu **java** s následujícím formátem:

```
java -Djava.library.path=library_path application_name
```

kde *cesta_knihovny* je cesta k třídám produktu WebSphere MQ pro knihovny Java, které zahrnují knihovny JNI. Tabulka 615 na stránce 1378 Zobrazuje umístění tříd produktu WebSphere MQ pro knihovny Java pro každou platformu.

Tabulka 615. Umístění tříd produktu WebSphere MQ pro knihovny Java pro každou platformu	
Platforma	**Adresář obsahující třídy WebSphere MQ pro knihovny Java**
AIX	*MQ_INSTALLATION_PATH*/java/lib (32bitové knihovny) *MQ_INSTALLATION_PATH*/java/lib64 (64 bitové knihovny)
HP-UX Linux (POWER, x86-64 a zSeries s390x platformy) Solaris (platformyx86-64 a SPARC)	*MQ_INSTALLATION_PATH*/java/lib (32bitové knihovny) *MQ_INSTALLATION_PATH*/java/lib64 (64 bitové knihovny)
Linux (platformax86)	*MQ_INSTALLATION_PATH*/java/bin
Windows	*MQ_INSTALLATION_PATH*\Java\lib (32bitové knihovny) *MQ_INSTALLATION_PATH*\Java\lib64 (64bitové knihovny)
Produkt *MQ_INSTALLATION_PATH* představuje adresář vysoké úrovně, do kterého je produkt WebSphere MQ nainstalován.	

Poznámka:

1. Na systémech AIX, HP-UX, Linux (Power platform) nebo Solaris použijte buď 32bitové knihovny, nebo 64bitové knihovny. 64bitové knihovny používejte pouze v případě, že spouštíte aplikaci v 64bitovém prostředí JVM (Java Virtual Machine) na 64bitové platformě. Jinak použijte 32bitové knihovny.
2. V systému Windowsmůžete použít proměnnou prostředí PATH k určení umístění tříd WebSphere MQ pro knihovny jazyka Java namísto zadávání jejich umístění v příkazu **java** .
3. Chcete-li používat třídy WebSphere MQ pro prostředí Java v režimu vázání v produktu IBM i, ujistěte se, že knihovna QMQMJAVA je ve vašem seznamu knihoven.

Související úlohy

Použití tříd produktu WebSphere MQ pro prostředí Java

Vlastnosti objektů IBM WebSphere MQ classes for JMS

Všechny objekty v produktu IBM WebSphere 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 WebSphere MQ classes for JMS poskytuje funkce pro nastavení a dotazování na vlastnosti objektů pomocí nástroje pro administraci produktu WebSphere MQ JMS, WebSphere MQ Explorer nebo v aplikaci. Mnohé z vlastností jsou relevantní pouze pro specifickou podmnožinu typů objektů.

Informace o způsobu použití nástroje pro administraci produktu WebSphere MQ JMS naleznete v tématu Použití nástroje pro administraci produktu WebSphere MQ JMS .

Produkt Tabulka 616 na stránce 1379 podává stručný popis každé vlastnosti a uvádí pro každou vlastnost, pro které typy objektů se používá. Typy objektů jsou identifikovány pomocí klíčových slov, viz Typy objektů JMS , kde jsou uvedeny vysvětlení těchto typů.

Čísla odkazují na poznámky na konci tabulky. Další informace najdete v tématu “Závislosti mezi vlastnostmi tříd produktu WebSphere MQ pro objekty platformy JMS” na stránce 1429.

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 616. Názvy vlastností a použitelné typy objektů

Vlastnost	**Krátký formát**	**Typ objektu**							
		CF	**QCF**	**TCF**	**Q**	**T**	**XACF**	**XAQCF**	**XATCF**
“APPLICATIONNAME” na stránce 1382	APPNAME	Y	Y	Y			Y	Y	Y
“VÝJIMKA ASYNCEXCEPTION” na stránce 1383	AEX	Y	Y	Y			Y	Y	Y
“BROKERCCDURSUBQ” na stránce 1384[1]	CCDSUB					Y			
“BROKERCCSUBQ” na stránce 1384[1]	CCSUB	Y		Y			Y		Y
“BROKERCONQ” na stránce 1385[1]	BCON	Y		Y			Y		Y
“BROKERDURSUBQ” na stránce 1385[1]	BDSUB					Y			
“BROKERPUBQ” na stránce 1386[1]	BPUB	Y		Y		Y	Y		Y
“BROKERPUBQMGR” na stránce 1386[1]	BPQM					Y			
“BROKERQMGR” na stránce 1386[1]	BQM	Y		Y			Y		Y
“BROKERSUBQ” na stránce 1387[1]	BSUB	Y		Y			Y		Y
“BROKERVER” na stránce 1387[1]	BVER	A[2]		A[2]		Y	Y		Y
“CCDTURL” na stránce 1388[3]	CCDT	Y	Y	Y			Y	Y	Y
“CCSID” na stránce 1388	CCS	Y	Y	Y	Y	Y	Y	Y	Y
“CHANNEL” na stránce 1389[3]	CHAN	Y	Y	Y			Y	Y	Y
“CLEANUP” na stránce 1389[1]	CL	Y		Y			Y		Y
“CLEANUPINT” na stránce 1390[1]	CLINT	Y		Y			Y		Y
“ConnectionNameList” na stránce 1390	CNLIST	Y	Y	Y					
“CLIENTRECONNECTOPTIONS” na stránce 1391	CROPT	Y	Y	Y					
“CLIENTRECONNECTTIMEOUT” na stránce 1392	CRT	Y	Y	Y					
“CLIENTID” na stránce 1392	CID	A[2]	Y	A[2]			Y	Y	Y
“CLONESUPP” na stránce 1392	CLS	Y		Y			Y		Y

Tabulka 616. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
"COMPHDR" na stránce 1393	HC	Y		Y			Y		Y
"COMPMSG" na stránce 1393	MC	Y	Y	Y			Y	Y	Y
"CONNOPT" na stránce 1394	CNOPT	Y	Y	Y			Y	Y	Y
"CONNTAG" na stránce 1395	CNTAG	Y	Y	Y			Y	Y	Y
"DESCRIPTION" na stránce 1395	DESC	A[2]	Y	A[2]	Y	Y	Y	Y	Y
"DIRECTAUTH" na stránce 1396	DAUTH	A[2]		A[2]					
"ENCODING" na stránce 1396	ENC				Y	Y			
"EXPIRY" na stránce 1397	EXP				Y	Y			
"FAILIFQUIESCE" na stránce 1398	FIQ	Y	Y	Y	Y	Y	Y	Y	Y
"HOSTNAME" na stránce 1398	HOST	A[2]	Y	A[2]			Y	Y	Y
"LOCALADDRESS" na stránce 1399	LA	A[2]	Y	A[2]			Y	Y	Y
"STYL MAPNAMESTYLE" na stránce 1399	MNST	Y	Y	Y			Y	Y	Y
"MAXBUFFSIZE" na stránce 1400	MBSZ	A[2]		A[2]					
"MDREAD" na stránce 1400	MDR				Y	Y			
"MDWRITE" na stránce 1401	MDW				Y	Y			
"MDMSGCTX" na stránce 1402	MDCTX				Y	Y			
"MSGBATCHSZ" na stránce 1402[1]	MBS	Y	Y	Y			Y	Y	Y
"MSGBODY" na stránce 1403	MBODY				Y	Y			
"MSGRETENTION" na stránce 1403	MRET	Y	Y				Y	Y	
"MSGSELECTION" na stránce 1404[1]	MSEL	Y		Y			Y		Y
"MULTICAST" na stránce 1404	MCAST	A[2]		A[2]		Y			
"OPTIMISTICPUBLICATION" na stránce 1405[1]	OPTPUB	Y		Y					
"OUTCOMENOTIFICATION" na stránce 1406[1]	NOTIFY	Y		Y					
"PERSISTENCE" na stránce 1406	PER				Y	Y			
"POLLINGINT" na stránce 1407[1]	PINT	Y	Y	Y			Y	Y	Y
"PORT" na stránce 1407	PORT	A[2]	Y	A[2]			Y	Y	Y
"PRIORITY" na stránce 1408	PRI				Y	Y			
"PROCESSDURATION" na stránce 1408[1]	PROCDUR	Y		Y					
"PROVIDERVERSION" na stránce 1409	PVER	Y	Y	Y			Y	Y	Y
"PROXYHOSTNAME" na stránce 1410	PHOST	A[2]		A[2]					

Tabulka 616. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
"PROXYPORT" na stránce 1411	PPORT	A[2]		A[2]					
"PUBACKINT" na stránce 1411[1]	PAI	Y		Y			Y		Y
"PUTASYNCALLOWED" na stránce 1411	PAALDOVA				Y	Y			
"QMANAGER" na stránce 1412	QMGR	Y	Y	Y	Y		Y	Y	Y
"QUEUE" na stránce 1412	QU				Y				
"READAHEADALLOWED" na stránce 1413	RAALDOVÁ				Y	Y			
"READAHEADCLOSEPOLICY" na stránce 1414	RACP				Y	Y			
"RECEIVECCSID" na stránce 1414	RCCS				Y	Y			
"RECEIVECONVERSION" na stránce 1415	RCNV				Y	Y			
"RECEIVEISOLATION" na stránce 1415[1]	RCVISOL	Y		Y					
"RECEXIT" na stránce 1415	RCX	Y	Y	Y			Y	Y	Y
"RECEXITINIT" na stránce 1416	RCXI	Y	Y	Y			Y	Y	Y
"REPLYTOSTYLE" na stránce 1416	RTOST				Y	Y			
"RESCANINT" na stránce 1417[1]	RINT	Y	Y				Y	Y	
"SECEXIT" na stránce 1418	SCX	Y	Y	Y			Y	Y	Y
"SECEXITINIT" na stránce 1418	SCXI	Y	Y	Y			Y	Y	Y
"SENDCHECKCOUNT" na stránce 1419	SCC	Y	Y	Y			Y	Y	Y
"SENDEXIT" na stránce 1419	SDX	Y	Y	Y			Y	Y	Y
"SENDEXITINIT" na stránce 1420	SDXI	Y	Y	Y			Y	Y	Y
"SHARECONVALLOWED" na stránce 1420	SCALD	Y	Y	Y			Y	Y	Y
"SPARSESUBS" na stránce 1421[1]	SSUBS	Y		Y					
"SSLCIPHERSUITE" na stránce 1421	SCPHS	Y	Y	Y			Y	Y	Y
"SSLCRL" na stránce 1421	SCRL	Y	Y	Y			Y	Y	Y
"SSLFIPSREQUIRED" na stránce 1422	SFIPS	Y	Y	Y			Y	Y	Y
"SSLPEERNAME" na stránce 1422	SPEER	Y	Y	Y			Y	Y	Y
"SSLRESETCOUNT" na stránce 1423	SRC	Y	Y	Y			Y	Y	Y
"STATREFRESHINT" na stránce 1423[1]	SRI	Y		Y			Y		Y
"SUBSTORE" na stránce 1424[1]	SS	Y		Y			Y		Y

Tabulka 616. Názvy vlastností a použitelné typy objektů (pokračování)

Vlastnost	Krátký formát	Typ objektu							
		CF	QCF	TCF	Q	T	XACF	XAQCF	XATCF
"SYNCPOINTALLGETS" na stránce 1424	SPAG	Y	Y	Y			Y	Y	Y
"TARGCLIENT" na stránce 1425	TC				Y	Y			
"TARGCLIENTMATCHING" na stránce 1425	TCM	Y	Y				Y	Y	
"TEMPMODEL" na stránce 1426	TM	Y	Y				Y	Y	
"TEMPQPREFIX" na stránce 1426	TQP	Y	Y				Y	Y	
"TEMPTOPICPREFIX" na stránce 1427	TTP	Y		Y			Y		Y
"TOPIC" na stránce 1427	TOP					Y			
"TRANSPORT" na stránce 1428	TRAN	A[2]	Y	A[2]			Y	Y	Y
"WILDCARDFORMAT" na stránce 1428	WCFMT	Y		Y			Y		Y

Poznámka:

1. Tuto vlastnost lze použít s verzí 7.0 pro třídy WebSphere MQ pro JMS, ale nemá žádný vliv na aplikaci připojenou ke správci front verze 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.

APPLICATIONNAME

Aplikace může nastavit název, který identifikuje její připojení ke správci front. Tento název aplikace se zobrazí příkazem **DISPLAY CONN MQSC/PCF** (kde se pole nazývá **APPLTAG**) nebo se zobrazí v zobrazení **Připojení aplikace** programu IBM WebSphere MQ Explorer (kde pole se nazývá **App name**).

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: APPLICATIONNAME

Krátký název nástroje pro administraci JMS: APPNAME

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setAppName ()
- MQConnectionFactory.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`.

VÝJIMKA ASYNCEXCEPTION

This property determines whether WebSphere MQ classes for JMS informs an ExceptionListener only when a connection is broken, or when any exception occurs asynchronously to a JMS API call. 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 JMS: ASYNCEXCEPTION

Krátký název nástroje pro administraci 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.

Prostředí	Hodnota
Nástroj pro administraci JMS	ALL
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_ALL = -1
Produkt WebSphere 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. V 7.5.0.8 Toto je výchozí hodnota z produktu IBM WebSphere MQ Version 7.5.0, opravná sada Fix Pack 8 (viz JMS: Změny modulu listener výjimek ve verzi 7.5).

Prostředí	Hodnota
Nástroj pro administraci JMS	SPOJENÍ PŘERUŠENO
Programový	WMQCONSTANTS.ASYNC_EXCEPTIONS_CONNECTIONBROKEN = 1
Produkt WebSphere MQ Explorer	Připojení přerušeno

Je definována následující dodatečná konstanta: V 7.5.0.8

- V produktu Version 7.5.0, Fix Pack 8: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_CONNECTIONBROKEN
- Před Version 7.5.0, Fix Pack 8: WMQCONSTANTS.ASYNC_EXCEPTIONS_DEFAULT = ASYNC_EXCEPTIONS_ALL

Související pojmy

Výjimky v třídách produktu WebSphere MQ pro službu JMS

BROKERCCDURSUBQ

Název fronty, z níž jsou načítány zprávy trvalého odběru pro ConnectionConsumer.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS: BROKERCCDURSUBQ

Krátký název nástroje pro administraci JMS: CCDSUB

Programový přístup

Metody Setter/getter

- MQTopic.setBrokerCCDurSubQueue()
- MQTopic.getBrokerCCDurSubQueue()

Hodnoty

SYSTEM.JMS.D.CC.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERCCSUBQ

Název fronty, z níž jsou načítány zprávy netrvalého odběru pro ConnectionConsumer.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: BROKERCCSUBQ

Krátký název nástroje pro administraci JMS: CCSUB

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerCCSubQueue()
- MQConnectionFactory.getBrokerCCSubQueue()

Hodnoty

SYSTEM.JMS.ND.CC.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERCONQ

Název řídicí fronty zprostředkovatele.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci služby JMS: BROKERCONQ

Krátký název nástroje pro administraci JMS: BCON

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerControlQueue()
- MQConnectionFactory.getBrokerControlQueue()

Hodnoty

SYSTEM.BROKER.CONTROL.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERDURSUBQ

Při použití tříd produktu WebSphere MQ pro platformu JMS v režimu migrace poskytovatele systému zpráv produktu WebSphere MQ tato vlastnost určuje název fronty, z níž jsou načítány zprávy trvalého odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS: BROKERDURSUBQ

Krátký název nástroje pro administraci JMS: BDSUB

Programový přístup

Metody Setter/getter

- MQTopic.setBrokerDurSubQueue()
- MQTopic.getBrokerDurSubQueue()

Hodnoty

SYSTEM.JMS.D.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Spouští se SYSTEM.JMS.D

Související pojmy
Pravidla pro výběr režimu poskytovatele systému zpráv produktu WebSphere MQ

BROKERPUBQ

Název fronty, do které jsou odesílány publikované zprávy (fronta proudu).

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: BROKERPUBQ

Krátký název nástroje pro administraci JMS: BPUB

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerPubQueue
- MQConnectionFactory.getBrokerPubQueue

Hodnoty

SYSTEM.BROKER.DEFAULT.STREAM
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERPUBQMGR

Název správce front, který vlastní frontu, do níž jsou odesílány zprávy publikované v rámci daného tématu.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS: BROKERPUBQMGR

Krátký název nástroje pro administraci JMS: BPQM

Programový přístup

Metody Setter/getter

- MQTopic.setBrokerPubQueueManager()
- MQTopic.getBrokerPubQueueManager()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERQMGR

Název správce front, v němž je zprostředkovatel spuštěn.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: BROKERQMGR

Krátký název nástroje pro administraci JMS: BQM

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerQueueManager()
- MQConnectionFactory.getBrokerQueueManager()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

BROKERSUBQ

Při použití tříd produktu WebSphere MQ pro platformu JMS v režimu migrace poskytovatele systému zpráv produktu WebSphere MQ tato vlastnost určuje název fronty, z níž jsou načítány zprávy z jiných než trvalých odběrů.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: BROKERSUBQ

Krátký název nástroje pro administraci JMS: BSUB

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerSubQueue()
- MQConnectionFactory.getBrokerSubQueue()

Hodnoty

SYSTEM.JMS.ND.SUBSCRIBER.QUEUE
Toto je výchozí hodnota.

Libovolný platný řetězec
Spouští se SYSTEM.JMS.ND

Související pojmy
Pravidla pro výběr režimu poskytovatele systému zpráv produktu WebSphere MQ

BROKERVER

Verze používaného zprostředkovatele.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, Topic, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: BROKERVER

Krátký název nástroje pro administraci JMS: BVER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setBrokerverze ()
- MQConnectionFactory.getBrokerverze ()

Hodnoty

V1
Chcete-li použít zprostředkovatele publikování/odběru produktu WebSphere MQ nebo použít zprostředkovatele produktu WebSphere MQ Integrator, WebSphere Event Broker, WebSphere Business Integration Event Broker nebo WebSphere Business Integration Message Broker v režimu kompatibility. Jedná se o výchozí hodnotu, pokud je funkce TRANSPORT nastavena na BIND nebo CLIENT.

V2
Chcete-li použít zprostředkovatele produktu WebSphere 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 JMS: CCDTURL

Krátký název nástroje pro administraci JMS: CCDT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setCCDTURL()
- MQConnectionFactory.getCCDTURL()

Hodnoty

null
Toto je výchozí hodnota.

Adresa URL (Uniform Resource Locator)

CCSID

ID kódované znakové sady, které má být použito pro připojení nebo místo určení.

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
Jedná se o výchozí hodnotu pro továrnu připojení.

1208
Jedná se o výchozí hodnotu pro místo určení.

Libovolné kladné celé číslo

CHANNEL

Název používaného kanálu připojení klienta.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: CHANNEL

Krátký název nástroje pro administraci JMS: CHAN

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setChannel()
- MQConnectionFactory.getChannel()

Hodnoty

SYSTEM.DEF.SVRCONN
Toto je výchozí hodnota.

Libovolný platný řetězec

CLEANUP

Úroveň úklidu pro úložiště odběrů BROKER či MIGRATE.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: CLEANUP

Krátký název nástroje pro administraci 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 JMS: CLEANUPINT

Krátký název nástroje pro administraci JMS: CLINT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setCleanupInterval ()
- MQConnectionFactory.getCleanupInterval ()

Hodnoty

3600000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

ConnectionNameList

Seznam názvů připojení TCP/IP. Seznam se zkouší v pořadí, jednou za každý pokus o opakování pokusu o opětovné připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: CONNECTIONNAMELIST

Krátký název nástroje pro administraci 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 JMS: CLIENTRECONECOPTIONS

Krátký název nástroje pro administraci JMS: CROPT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setClientReconnectOptions()
- MQConnectionFactory.getClientReconnectOptions()

Hodnoty

QMGR

Aplikace se může znovu připojit, avšak pouze ke stejnému správci front, ke kterému se připojovala původně.

Tuto hodnotu použijte v případě, že lze aplikaci znovu připojit, ale existuje afinita mezi třídami produktu WebSphere MQ pro aplikaci 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 k libovolnému správci fronty.

Volbu opětovného připojení použijte pouze v případě, že neexistuje žádná afinita mezi třídami produktu WebSphere 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.

Určuje, zda se aplikace automaticky znovu připojí, a to závisí na hodnotě atributu kanálu WebSphere MQ DefReconnect.

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 JMS: CLIENTRECONNECTSIMEOUT

Krátký název nástroje pro administraci 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 JMS: CLIENTID

Krátký název nástroje pro administraci JMS: CID

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setClientID ()
- MQConnectionFactory.getClientID ()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec

CLONESUPP

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 JMS: CLONESUPP

Krátký název nástroje pro administraci 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 souběžně, ale každá instance musí být spuštěna v odděleném prostředí JVM (Java Virtual Machine).

COMPHDR

Seznam technik, které lze použít pro kompresi dat záhlaví na připojení.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: COMPHDR

Krátký název nástroje pro administraci JMS: HC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setHdrCompList()
- MQConnectionFactory.getHdrCompList()

Hodnoty

ŽÁDNÉ
Toto je výchozí hodnota.

SYSTÉM
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 JMS: COMPMSG

Krátký název nástroje pro administraci JMS: MC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMsgCompList()
- MQConnectionFactory.getMsgCompList()

Hodnoty

ŽÁDNÉ
Toto je výchozí hodnota.

Seznam jedné nebo více následujících hodnot oddělených prázdnými znaky:
RLE ZLIBFAST ZLIBHIGH

CONNOPT

Řídí způsob, jakým jsou třídy WebSphere MQ pro aplikace JMS, které používají připojení vazeb, připojeny ke správci front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: CONNOPT

Krátký název nástroje pro administraci JMS: CNOPT

Programový přístup

Metody Setter/getter

- MQConnectionFactory. Volby volbysetMQConnection()
- MQConnectionFactory. Volby volbygetMQConnection()

Hodnoty

STANDARD
Povaha vazby mezi aplikací a správcem front závisí na hodnotě atributu *DefaultBindTyp* správce front. Hodnota STANDARD mapuje na WebSphere 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 produkt WebSphere 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 WebSphere MQ *ConnectOption* MQCNO_ISOLATED_BINDING.

Rychlý
Aplikace a lokální agent správce front se spouští ve stejné jednotce provedení. Tato hodnota je mapována na produkt WebSphere MQ *ConnectOption* MQCNO_FASTPATH_BINDING.

SERIALQM
Aplikace vyžaduje výlučné použití značky připojení v rámci oboru správce front. Tato hodnota je mapována na produkt WebSphere 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 produkt WebSphere 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 produkt WebSphere 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 produkt WebSphere MQ *ConnectOption* MQCNO_RESTRICT_CONN_TAG_QSG.

Další informace o volbách připojení k produktu WebSphere 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 JMS: CONNTAG

Krátký název nástroje pro administraci 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 pro administraci JMS: POPIS

Krátký název nástroje pro administraci JMS: DESC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setDescription()
- MQConnectionFactory.getDescription()

Hodnoty

null
Toto je výchozí hodnota.

Libovolný platný řetězec
`

DIRECTAUTH

Zda je ověřování SSL používáno v reálném čase připojení k danému zprostředkovateli.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: DIRECTAUTH

Krátký název nástroje pro administraci JMS: DAUTH

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setDirectAuth ()
- MQConnectionFactory.getDirectAuth ()

Hodnoty

ZÁKLADNÍ
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 JMS: ENCODING

Krátký název nástroje pro administraci JMS: ENC

Programový přístup

Metody Setter/getter

- MQDestination.setEncoding()
- MQDestination.getEncoding()

Hodnoty

Vlastnost ENCODING

Platné hodnoty, které může vlastnost 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 JMS: EXPIRY

Krátký název nástroje pro administraci 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 JMS: FAILIFQUIESCE

Krátký název nástroje pro administraci 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 JMS: HOSTNAME

Krátký název nástroje pro administraci 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. V případě připojení v reálném čase ke zprostředkovateli je tato vlastnost relevantní pouze při použití výběrového vysílání a určuje lokální síťové rozhraní, které má být použito.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: LOCALADDRESS

Krátký název nástroje pro administraci JMS: LA

Programový přístup

Metody Setter/getter

- MQConnectionFactory.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 JMS: MAPNAMESTYLE

Krátký název nástroje pro administraci JMS: MNST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMapNameStyle()
- MQConnectionFactory.getMapNameStyle()

Hodnoty

STANDARD
Bude použit standardní formát pojmenování prvků com.ibm.jms.JMSMapMessage . Jedná se o výchozí hodnotu a povoluje, aby se jako název prvku použily nepovolené identifikátory Java.

Kompatibilní
Je třeba použít starší formát pojmenování prvků com.ibm.jms.JMSMapMessage . Jako název prvku lze použít pouze platné identifikátory jazyka Java. Tento stav je nezbytný pouze v případě, že se odesílají zprávy mapování na aplikaci používající verzi produktu IBM WebSphere MQ classes for JMS starší než verze 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 JMS: MAXBUFFSIZE

Krátký název nástroje pro administraci JMS: MBSZ

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMaxBufferSize()
- MQConnectionFactory.getMaxBufferSize()

Hodnoty

1000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

MDREAD

Tato vlastnost určuje, zda aplikace JMS může extrahovat hodnoty z polí MQMD.

Použitelné objekty

Dlouhý název nástroje pro administraci JMS: MDREAD

Krátký název nástroje pro administraci JMS: MDR

Programový přístup

Metody Setter/getter

- MQDestination.setMQMDReadEnabled()
- MQDestination.getMQMDReadEnabled()

Hodnoty

NO
Při odesílání zpráv nejsou vlastnosti JMS_IBM_MQMD* 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 platformy JMS nastavit hodnoty polí MQMD.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci JMS: MDWRITE

Krátký název nástroje pro administraci JMS: MDR

Programový přístup

Metody Setter/getter

- MQDestination.setMQMDWriteEnabled()
- MQDestination.getMQMDWriteEnabled()

Hodnoty

NO
Všechny vlastnosti JMS_IBM_MQMD* jsou ignorovány a jejich hodnoty 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 JMS: MDMSGCTX

Krátký název nástroje pro administraci 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 JMS: MAXBUFFSIZE

Krátký název nástroje pro administraci JMS: MBSZ

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMsgBatchSize()
- MQConnectionFactory.getMsgBatchSize()

Hodnoty

10
Toto je výchozí hodnota.

Libovolné kladné celé číslo

MSGBODY

Určuje, zda aplikace JMS přistupuje ke zprávě MQRFH2 zprávy produktu IBM WebSphere MQ jako součást informačního obsahu zprávy.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci JMS: WMQ_MESSAGE_BODY

Krátký název nástroje pro administraci JMS: MBODY

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMessageBodyStyle()
- MQConnectionFactory.getMessageBodyStyle()

Hodnoty

Neuvedeno
Při odesílání produkt IBM WebSphere 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 WebSphere MQ classes for JMS automaticky vygeneruje záhlaví MQRFH2 a zahrne jej do zprávy produktu WebSphere MQ .

Při příjmu IBM WebSphere MQ classes for JMS nastavte vlastnosti zprávy JMS podle hodnot v záhlaví MQRFH2 (pokud existuje). Neprezentuje MQRFH2 jako část těla zprávy JMS.

MQ
Při odesílání produkt IBM WebSphere MQ classes for JMS negeneruje MQRFH2.

Při příjmu produkt IBM WebSphere MQ classes for JMS prezentuje MQRFH2 jako část těla zprávy JMS.

MSGRETENTION

Určuje, zda spotřebitel připojení uchovává nedoručené zprávy ve vstupní frontě.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory,

Dlouhý název nástroje pro administraci JMS: MSGRETENTION

Krátký název nástroje pro administraci JMS: MRET

Programový přístup

Metody Setter/getter

- MQConnectionFactory.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 třídami produktu WebSphere MQ pro službu 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 JMS: MSGSELECTION

Krátký název nástroje pro administraci JMS: MSEL

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setMessageSelection ()
- MQConnectionFactory.getMessageSelection ()

Hodnoty

CLIENT
Výběr zpráv je prováděn třídami produktu WebSphere MQ pro platformu 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 JMS: MULTICAST

Krátký název nástroje pro administraci 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 třídy produktu WebSphere MQ pro službu JMS vrátí řízení okamžitě vydavateli, který publikoval zprávu, nebo zda vrátí řízení až poté, co dokončí veškeré zpracování přidružené k volání a může ohlásit výsledek vydavateli.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: OPTIMISTICPUBLIKACE

Krátký název nástroje pro administraci JMS: OPTPUB

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setOptimisticPublication ()
- MQConnectionFactory.getOptimisticPublication ()

Hodnoty

NO
Když vydavatel publikuje zprávu, třídy WebSphere MQ pro JMS nevrací řízení vydavateli, dokud nedokončí veškeré zpracování přidružené k volání a může hlásit výsledek vydavateli. Toto je výchozí hodnota.

YES
Když vydavatel publikuje zprávu, třídy WebSphere MQ pro službu JMS vrátí řízení vydavateli bezprostředně před dokončením všech zpracování přidružených k volání a mohou ohlásit výsledek vydavateli. Třídy WebSphere MQ pro službu JMS hlásí výsledek pouze v případě, že vydavatel zprávu potvrdí.

OUTCOMENOTIFICATION

Tato vlastnost určuje, zda třídy produktu WebSphere MQ pro službu JMS okamžitě vracejí 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 JMS: OUTCOMENOTIFICATION

Krátký název nástroje pro administraci JMS: NOTIFY

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setOutcomeNotification ()
- MQConnectionFactory.getOutcomeNotification ()

Hodnoty

YES
Když odběratel potvrdí nebo potvrdí zprávu, třídy WebSphere MQ pro JMS nevracejí řízení do odběratele, dokud nedokončí všechny zpracování přidružené k volání a může oznámit výsledek odběrateli. Toto je výchozí hodnota.

NO
Když odběratel potvrdí nebo potvrdí zprávu, třídy WebSphere MQ pro službu JMS vrátí řízení odběrateli okamžitě, předtím, než dokončí všechny zpracování přidružené k volání a může oznámit výsledek odběrateli.

PERSISTENCE

Trvalost zpráv odeslaných do místa určení.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci JMS: PERSISTENCE

Krátký název nástroje pro administraci JMS: PER

Programový přístup

Metody Setter/getter

- MQDestination.setPersistence()
- MQDestination.getPersistence()

Hodnoty

Aplikace
Perzistence je definována aplikací JMS. Toto je výchozí hodnota.

QDEF
Persistence přebírá hodnotu předvolby fronty.

PERRY
Zprávy jsou trvalé.

JINÝ
Zprávy jsou přechodné.

VYSOKÁ
Další informace o použití této hodnoty naleznete v tématu Trvalá zpráva JMS .

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 JMS: POLLINGINT

Krátký název nástroje pro administraci JMS: PINT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setPollingInterval ()
- MQConnectionFactory.getPollingInterval ()

Hodnoty

5000
Toto je výchozí hodnota.

Libovolné kladné celé číslo

PORT

V případě připojení ke správci front 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 JMS: PORT

Krátký název nástroje pro administraci JMS: PORT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setPort()
- MQConnectionFactory.getPort()

Hodnoty

1414
Jedná se o výchozí hodnotu, pokud je 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 JMS: PRIORITY

Krátký název nástroje pro administraci JMS: PRI

Programový přístup

Metody Setter/getter

- MQDestination.setPriority()
- MQDestination.getPriority()

Hodnoty

Aplikace
Priorita je definována aplikací JMS. Toto je výchozí hodnota.

QDEF
Priorita přebírá hodnotu předvolby fronty.

Libovolné celé číslo v rozsahu 0 až 9
Nejnižší na nejvyšší.

PROCESSDURATION

Tato vlastnost určuje, zda odběratel zaručuje, že bude před vrácením řízení do tříd produktu WebSphere MQ pro platformu JMS rychle zpracován každou zprávu, kterou obdrží.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: PROCESSSDURATION

Krátký název nástroje pro administraci 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 rychle zpracovat všechny zprávy, které obdrží před vrácením řízení do tříd produktu WebSphere MQ pro platformu JMS.

PROVIDERVERSION

Tato vlastnost rozlišuje mezi dvěma režimy systému zpráv produktu WebSphere MQ : WebSphere MQ je normální režim a režim migrace poskytovatele systému zpráv WebSphere MQ .

Normální režim poskytovatele systému zpráv produktu WebSphere MQ používá všechny funkce správců front produktu WebSphere MQ verze 7.0 k implementaci rozhraní JMS. Tento režim se používá pouze pro připojení ke správci front produktu WebSphere MQ a může se připojit ke správcům front produktu WebSphere MQ verze 7.0 v režimu vazeb klienta nebo vazby. Tento režim je optimalizován pro použití nové funkce produktu WebSphere MQ verze 7.0 . Pokud nepoužíváte produkt WebSphere MQ Real-Time Transport, pak je použitý režim operace primárně určen hodnotou vlastnosti PROVIDERVERSION továrny připojení.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: PROVIDERVERSION

Krátký název nástroje pro administraci JMS: PVER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setProviderVerze ()
- MQConnectionFactory.getProviderVerze ()

Hodnoty

Můžete nastavit **PROVIDERVERSION** na možné hodnoty: 7, 6, nebo *nespecifikováno*. Avšak **PROVIDERVERSION** může být 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.

7
Používá normální režim poskytovatele systému zpráv WebSphere MQ .

Nastavíte-li parametr PROVIDERVERSION na hodnotu 7, bude k dispozici pouze normální režim operace poskytovatele systému zpráv WebSphere MQ . Pokud správce front, který je připojen k jiným nastavením v továrně připojení, není správcem front verze 7.0 , metoda createConnection() selže s výjimkou.

Normální režim poskytovatele systému zpráv WebSphere MQ používá funkci sdílení konverzací a počet konverzací, které lze sdílet, je řízen vlastností SHARECNV () na kanálu připojení serveru. Je-li tato

vlastnost nastavena na hodnotu 0, nelze použít normální režim poskytovatele systému zpráv produktu WebSphere MQ a metoda createConnection() selže s výjimkou.

6

Používá režim migrace poskytovatele systému zpráv produktu WebSphere MQ .

Třídy WebSphere MQ pro platformu JMS používají funkce a algoritmy dodávané s produktem WebSphere MQ verze 6.0. Chcete-li se připojit k produktu WebSphere Event Broker nebo WebSphere Message Broker s použitím produktu WebSphere MQ Enterprise Transport, je třeba tento režim použít. Pomocí tohoto režimu se můžete připojit ke správci front produktu WebSphere MQ verze 7.0 , ale nejsou použity žádné nové funkce správce front verze 7.0 , například dopředné čtení nebo kontinuální čtení.

nespecifikováno

Jedná se o výchozí hodnotu a skutečný text je "nespecifikovaný".

Továrna připojení, která byla vytvořena s předchozí verzí tříd produktu WebSphere MQ pro JMS v rozhraní JNDI, přebírá tuto hodnotu, je-li továrna připojení použita s novou verzí tříd WebSphere MQ pro JMS. Při určování použitého režimu operací se používá následující algoritmus. Tento algoritmus se používá při volání metody createConnection() a používá jiné aspekty továrny připojení k určení, zda je vyžadován normální režim poskytovatele systému zpráv produktu WebSphere MQ nebo režim migrace poskytovatele systému zpráv produktu WebSphere MQ .

- Nejprve je proveden pokus o použití běžného režimu poskytovatele systému zpráv WebSphere MQ .
- Pokud připojený správce front není produkt WebSphere MQ verze 7.0, je připojení uzavřeno a místo něj bude použit režim migrace poskytovatele systému zpráv produktu WebSphere MQ .
- Je-li vlastnost SHARECNV () na kanálu připojení serveru nastavena na hodnotu 0, je připojení uzavřeno a místo něj bude použit režim migrace poskytovatele systému zpráv produktu WebSphere MQ .
- Je-li parametr BROKERVER nastaven na hodnotu 1 nebo je zadána nová výchozí hodnota "unspecified", bude nadále používán normální režim poskytovatele systému zpráv WebSphere MQ , a proto všechny operace publikování/odběru používají nové funkce produktu WebSphere MQ V7.0 . Pokud je v režimu kompatibility použit produkt WebSphere Event Broker nebo WebSphere Message Broker (a chcete použít funkci publikování/odběru verze 6.0 namísto funkce publikování/odběru produktu WebSphere MQ verze 7), nastavte parametr PROVIDERVERSION na hodnotu 6, aby byl použit režim migrace poskytovatele systému zpráv produktu WebSphere MQ .

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 JMS: PROXYHOSTNAME

Krátký název nástroje pro administraci JMS: PHOST

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setProxyHostName()
- MQConnectionFactory.getProxyHostName()

Hodnoty

null

Název hostitele serveru proxy. Toto je výchozí hodnota.

PROXYPORT

Číslo portu, na kterém naslouchá server proxy při použití připojení v reálném čase ke zprostředkovateli prostřednictvím serveru proxy.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: PROXYPPORT

Krátký název nástroje pro administraci 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 před třídami produktu WebSphere MQ pro službu JMS vyžaduje potvrzení od zprostředkovatele.

Když snížíte hodnotu této vlastnosti, třídy WebSphere MQ pro požadavky JMS budou častěji potvrzovat, takže se výkon vydavatele sníží. Když zvýšíte hodnotu, třídy WebSphere MQ pro platformu JMS mají delší dobu na 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 JMS: PROXYPPORT

Krátký název nástroje pro administraci 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 JMS: PUTASYNCALLOWED

Krátký název nástroje pro administraci 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 tabulku definic kanálů klienta k připojení ke správci front, přečtěte si téma Použití tabulky definic kanálů klienta se třídami produktu WebSphere MQ pro platformu JMS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, Queue, XAConnectionFactory, XAQueueConnection Factory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: QMANAGER

Krátký název nástroje pro administraci JMS: QMGR

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setQueueManager ()
- MQConnectionFactory.getQueueManager ()

Hodnoty

"" (prázdný řetězec)
Jako výchozí hodnota může být libovolný řetězec.

QUEUE

Název cíle fronty JMS. Shoduje se s názvem fronty, kterou používá správce front.

Použitelné objekty

Fronta

Dlouhý název nástroje pro administraci JMS: QUEUE

Krátký název nástroje pro administraci JMS: QU

Hodnoty

Libovolný řetězec
Libovolné platné jméno fronty IBM WebSphere MQ .

Související pojmy
Pravidla pro pojmenování objektů IBM WebSphere 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 JMS: READAHEADALLOWED

Krátký název nástroje pro administraci 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 JMS: READAHEADCLOSEPOLICY

Krátký název nástroje pro administraci 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 JMS: RECEIVECCSID

Krátký název nástroje pro administraci JMS: RCCS

Programový přístup

Metody Setter/Getter

- `MQDestination.setReceiveCCSID`
- `MQDestination.getReceiveCCSID`

Hodnoty

`WMQConstants.WMQ_RECEIVE_CCSID_JVM_DEFAULT`
0 -použít prostředí JVM `Charset.defaultCharset`

1208
UTF-8

CCSID
Podporovaný identifikátor kódované znakové sady.

RECEIVECONVERSION

Cílová vlastnost, která určuje, zda má správce front provést převod dat.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci JMS: RECEIVECONVERSION

Krátký název nástroje pro administraci JMS: RCNV

Programový přístup

Metody Setter/Getter

- `MQDestination.setReceiveConversion`
- `MQDestination.getReceiveConversion`

Hodnoty

`WMQConstants.WMQ_RECEIVE_CONVERSION_CLIENT_MSG`
1 -Provést konverzi dat pouze na klientovi 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 JMS: RECEIVEISOLATION

Krátký název nástroje pro administraci 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 spouštěny za dědění.

Je možné, že bude vyžadována další konfigurace, aby produkt IBM WebSphere MQ classes for JMS mohl vyhledat uživatelské procedury pro příjem. Další informace naleznete v tématu Konfigurace tříd produktu IBM WebSphere MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: RECEXIT

Krátký název nástroje pro administraci JMS: RCX

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setReceiveExit ()
- MQConnectionFactory.getReceiveExit ()

Hodnoty

null
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 jazyce Java).
- Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru pro přijetí zprávy kanálu není zapsána v jazyce Java).

Toto je výchozí hodnota.

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 JMS: RECEXITINIT

Krátký název nástroje pro administraci JMS: RCXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setReceiveExitInit()
- MQConnectionFactory.getReceiveExitInit()

Hodnoty

null
Řetězec obsahující jednu nebo více položek 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 JMS: REPLYTOSTYLE

Krátký název nástroje pro administraci 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 WebSphere MQ verze 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 přijímat, třídy WebSphere MQ pro službu JMS prohledávají frontu WebSphere MQ pro vhodné zprávy v posloupnosti určené atributem `MsgDeliverySequence` fronty.

Jakmile třídy WebSphere MQ pro JMS najdou vhodnou zprávu a dodávají ji spotřebiteli, třídy WebSphere MQ pro platformu JMS obnoví hledání další vhodné zprávy z aktuální pozice ve frontě. Třídy WebSphere MQ pro platformu JMS nadále prohledávají frontu tímto způsobem, dokud nedosáhne konce fronty, nebo dokud nevyprší časový interval (v milisekundách) určený hodnotou této vlastnosti. V každém případě se třídy WebSphere MQ pro systém JMS vrátí na začátek fronty, aby bylo možné pokračovat ve vyhledávání, a zahájí se nový časový interval.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci JMS: RESCANINT

Krátký název nástroje pro administraci JMS: RINT

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setRescanInterval ()
- MQConnectionFactory.getRescanInterval ()

Hodnoty

5000
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 WebSphere MQ classes for JMS vyhledal uživatelské procedury zabezpečení. Další informace naleznete v tématu Konfigurace tříd produktu IBM WebSphere MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SECEXIT

Krátký název nástroje pro administraci JMS: SXC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSecurityExit ()
- MQConnectionFactory.getSecurityExit ()

Hodnoty

null
Název třídy, která implementuje rozhraní WMQSecurityExit (pro uživatelskou proceduru zabezpečení kanálu napsanou v jazyce Java).

Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru zabezpečení kanálu, která není zapsána v jazyku Java).

SECEXITINIT

Uživatelská data, která jsou předána uživatelské proceduře pro zabezpečení zprávy kanálu při volání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SECEXITINIT

Krátký název nástroje pro administraci JMS: SCXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSecurityExitInit()
- MQConnectionFactory.getSecurityExitInit()

Hodnoty

null
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 JMS: SENDCHECKCOUNT

Krátký název nástroje pro administraci 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 WebSphere MQ classes for JMS mohl vyhledat uživatelské procedury odeslání. Další informace naleznete v tématu Konfigurace tříd produktu IBM WebSphere MQ pro rozhraní JMS pro použití uživatelských procedur kanálů.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SENDEXIT

Krátký název nástroje pro administraci JMS: SDX

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSendExit ()
- MQConnectionFactory.getSendExit ()

Hodnoty

null
Jakýkoli řetězec obsahující jednu nebo více položek oddělených čárkami, kde každá položka je buď:

- Název třídy, která implementuje rozhraní WMQSendExit (pro uživatelskou proceduru pro odeslání zprávy kanálu napsané v jazyce Java).
- Řetězec ve formátu *libraryName(entryPointName)* (pro uživatelskou proceduru pro odeslání zprávy kanálu, která není zapsána v jazyce Java).
-

Toto je výchozí hodnota.

SENDEXITINIT

Uživatelská data, která jsou předána uživatelským procedurám pro odeslání zprávy kanálu při volání.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SENDEXITINIT

Krátký název nástroje pro administraci JMS: SDXI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSendExitInit()
- MQConnectionFactory.getSendExitInit()

Hodnoty

null
Výchozí hodnota může být libovolný řetězec obsahující jednu nebo více položek dat uživatele oddělených čárkami.

SHARECONVALLOWED

Tato vlastnost určuje, zda připojení klienta může sdílet svůj soket s dalšími připojeními JMS nejvyšší úrovně ze stejného procesu do stejného správce front, pokud se definice kanálu shodují.

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
Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_SHARE_CONV_ALLOWED_YES.

NO
Tato hodnota je určena pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_SHARE_CONV_ALLOWED_NO.

SPARSESUBS

Řídí zásadu načítání zpráv objektu TopicSubscriber .

Použitelné objekty

ConnectionFactory, TopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SPARSESUBS

Krátký název nástroje pro administraci JMS: SSUBS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSparseOdběry ()
- MQConnectionFactory.getSparseOdběry ()

Hodnoty

NO
Odběry přijímají 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 SSL.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SSLCIPHERSUITE

Krátký název nástroje pro administraci JMS: SCPHS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLCipherSuite ()
- MQConnectionFactory.getSSLCipherSuite ()

Hodnoty

null
Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti SSL pro objekty platformy JMS.

SSLCRL

Servery CRL, které mají zkontrolovat odvolání certifikátů SSL.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SSLCRL

Krátký název nástroje pro administraci JMS: SCRL

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLCertStores ()
- MQConnectionFactory.getSSLCertStores ()

Hodnoty

null
Seznam adres URL LDAP oddělených mezerami. Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti SSL pro objekty platformy JMS.

SSLFIPSREQUIRED

Tato vlastnost určuje, zda připojení SSL musí používat sadu CipherSuite , kterou podporuje poskytovatel IBM Java JSSE FIPS (IBMJSSEFIPS).

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.setSSLFipsVyžadováno ()
- MQConnectionFactory.getSSLFipsRequired ()

Hodnoty

NO
Připojení SSL může použít libovolnou sadu CipherSuite , která není podporována poskytovatelem IBM Java JSSE FIPS (IBMJSSEFIPS).

Toto je výchozí hodnota. V programech použijte false.

YES
Připojení SSL musí používat sadu CipherSuite , která je podporována IBMJSSEFIPS.

V programech použijte hodnotu true.

SSLPEERNAME

Pro zabezpečení SSL jde o kostru *distinguished name* , která se musí shodovat s názvem, který je poskytnut správcem front.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SSLPEERNAME

Krátký název nástroje pro administraci JMS: SPEER

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLPeerNázev ()
- MQConnectionFactory.getSSLPeerNázev ()

Hodnoty

null
Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti SSL pro objekty platformy JMS.

SSLRESETCOUNT

Pro zabezpečení SSL se jedná o celkový počet bajtů odeslaných a přijatých připojením před opětovným získáním tajného klíče, který je použit pro šifrování.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, TopicConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory, XATopicConnectionFactory

Dlouhý název nástroje pro administraci JMS: SSLRESETCOUNT

Krátký název nástroje pro administraci JMS: SRC

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSSLResetCount ()
- MQConnectionFactory.getSSLResetCount ()

Hodnoty

0
Nula nebo jakékoli kladné celé číslo menší nebo rovné 999, 999, 999. Toto je výchozí hodnota. Další informace naleznete v tématu Vlastnosti SSL pro objekty platformy 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 JMS: STATREFRESHINT

Krátký název nástroje pro administraci JMS: SRI

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setStatusRefreshInterval()
- MQConnectionFactory.getStatusRefreshInterval()

Hodnoty

6000
Libovolné kladné celé číslo může být výchozí hodnota. Další informace naleznete v tématu Vlastnosti SSL pro objekty platformy JMS.

SUBSTORE

Kde třídy produktu WebSphere MQ pro službu JMS ukládají trvalá data týkající se aktivních odběrů.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci JMS: SUBSTORE

Krátký název nástroje pro administraci JMS: SS

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSubscriptionStore ()
- MQConnectionFactory.getSubscriptionStore ()

Hodnoty

BROKER
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 JMS: SYNCPOINTALLGETS

Krátký název nástroje pro administraci JMS: SPAG

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setSyncpointAllGets()
- MQConnectionFactory.getSyncpointAllGets()

Hodnoty

Ne
Toto je výchozí hodnota.

Ano

TARGCLIENT

Tato vlastnost určuje, zda má být použit formát WebSphere MQ RFH2 k výměně informací s cílovými aplikacemi.

Použitelné objekty

Fronta, Téma

Dlouhý název nástroje pro administraci JMS: TARGCLIENT

Krátký název nástroje pro administraci JMS: TC

Programový přístup

Metody Setter/getter

- MQDestination.setTargetClient()
- MQDestination.getTargetClient()

Hodnoty

JMS
Cílem zprávy je aplikace JMS. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte WMQConstants.WMQ_CLIENT_JMS_COMPLIANT.

MQ
Cílem zprávy je aplikace WebSphere MQ , která není JMS.

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 JMS: TARGCLIENTMATCHING

Krátký název nástroje administrace platformy JMS: 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. Pokud má zpráva záhlaví MQRFH2 , vlastnost TARGCLIENT je místo toho nastavena na platformu JMS. Jedná se o výchozí hodnotu pro administrativní nástroje.

Pro programy použijte hodnotu true.

NO
Vlastnost TARGCLIENT objektu fronty odvozeného z pole záhlaví JMSReplyTo v příchozí zprávě je vždy nastavena na JMS.

Pro programy použijte hodnotu false.

TEMPMODEL

Název modelové fronty, z níž jsou vytvářeny dočasné fronty platformy JMS.

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci JMS: TEMPMODEL

Krátký název nástroje pro administraci JMS: TM

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTemporaryModel ()
- MQConnectionFactory.getTemporaryModel ()

Hodnoty

SYSTEM.DEFAULT.MODEL.QUEUE
Jako výchozí hodnota může být libovolný řetězec.

TEMPQPREFIX

Předpona, která se používá k vytvoření názvu dynamické fronty produktu WebSphere MQ .

Použitelné objekty

ConnectionFactory, QueueConnectionFactory, XAConnectionFactory, XAQueueConnectionFactory

Dlouhý název nástroje pro administraci JMS: TEMPQPREFIX

Krátký název nástroje pro administraci JMS: TQP

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTempQPrefix ()
- MQConnectionFactory.getTempQPrefix ()

Hodnoty

" " (prázdný řetězec)
Použitá předpona je `CSQ.*` na 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 WebSphere MQ (struktura MQOD), ale poslední nemezerový znak musí být hvězdička.

TEMPTOPICPREFIX

Při vytváření dočasných témat generuje služba JMS řetězec tématu ve formě '' TEMP/*TEMPTOPICPREFIX/unique_id*'', nebo je-li tato vlastnost ponechána 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 JMS: TEMPTOPICPREFIX

Krátký název nástroje pro administraci JMS: TTP

Programový přístup

Metody Setter/getter

- MQConnectionFactory.setTempTopicPrefix()
- MQConnectionFactory.getTempTopicPrefix()

Hodnoty

Jakýkoli řetězec, který není null, sestává pouze z platných znaků pro řetězec tématu WebSphere MQ . Výchozí hodnota je '' '' (prázdný řetězec).

TOPIC

Název místa určení tématu JMS. Tato hodnota je používána správcem front jako řetězec tématu publikování nebo odběru.

Použitelné objekty

Téma

Dlouhý název nástroje pro administraci JMS: TOPIC

Krátký název nástroje pro administraci JMS: TOP

Hodnoty

Libovolný řetězec
Řetězec, který tvoří platný řetězec tématu IBM WebSphere MQ . Při použití IBM WebSphere 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 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.

WILDCARDFORMAT

Tato vlastnost určuje, která verze syntaxe zástupných znaků má být použita.

Použitelné objekty

ConnectionFactory, TopicConnectionFactory, XAConnectionFactory, továrna XATopicConnection

Dlouhý název nástroje pro administraci platformy JMS: WILDCARDFORMAT

Krátký název nástroje pro administraci 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.

Závislosti mezi vlastnostmi tříd produktu WebSphere MQ pro objekty platformy 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.

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 SSL pro objekty platformy JMS

Povolte šifrování SSL (Secure Sockets Layer) pomocí vlastnosti SSLCIPHERSUITE. Charakteristiky šifrování SSL pak můžete změnit pomocí několika dalších vlastností.

Určíte-li funkci TRANSPORT (CLIENT), můžete pomocí vlastnosti SSLCIPHERSUITE povolit šifrovanou komunikaci SSL (Secure Sockets Layer). 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 SSL. Je-li v vlastnosti SSLCIPHERSUITE zadán název CipherSpec , systém JMSAdmin vydá varování a mapuje CipherSpec na ekvivalentní sadu CipherSuite. Seznam CipherSpecs rozeznaný produktem WebSphere MQ a JMSAdmin naleznete v tématu SSL CipherSpecs a CipherSuites v rozhraní JMS .

Pokud vyžadujete připojení pro použití sady CipherSuite , které je podporováno poskytovatelem IBM Java JSSE FIPS (IBMJSSEFIPS), nastavte vlastnost SSLFIPSREQUIRED na továrnu 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 názvu atributu a hodnot oddělených čárkami nebo středníky. Příklad:

```
SSLPEERNAME(CN=QMGR.*, OU=IBM, OU=WEBSPHERE)
```

Sada názvů a hodnot tvoří *rozlišující název*. Další informace o rozlišujících názvech a jejich použití s produktem WebSphere MQnaleznete v tématu Zabezpečení.

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 prostředí 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 SSL prezentovaný serverem kontrolován proti zadaným serverům CRL. Další informace o zabezpečení CRL viz Zabezpečení .

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 zahrnuje také řídicí informace odeslané a přijaté třídami produktu WebSphere MQ pro JMS.

Chcete-li například konfigurovat objekt ConnectionFactory , který lze použít k vytvoření připojení prostřednictvím kanálu MQI s povoleným SSL s použitím tajného klíče, který je znovu vyjednán po 4 MB dat, zadejte do správce JMSAdmin tento 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.

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, je-li poskytnuta, vám pomohou 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 IBM 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 (http://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: