[MSMQ 概述](http://www.cnblogs.com/danielWise/archive/2011/02/28/1966808.html)

2011-02-28 11:33 by DanielWise, 8088 阅读, 3 评论, [收藏](http://www.cnblogs.com/danielWise/archive/2011/02/28/1966808.html), [编辑](http://i.cnblogs.com/EditPosts.aspx?postid=1966808)

**1) MSMQ概述**

MSMQ 表示微软消息队列服务。MSMQ 可以工作在在线或者离线场景，并提供异步编程功能。如果客户端离线，MSMQ将会是最合适的方法，这是因为服务端不需要等待客户端读取数据并向服务端返回确认。

**(2) 确定MSMQ 是否已经安装**

通过在运行窗口执行"Services"，然后找到Message Queuing. 如果没有就说明MSMQ 没有安装。  
  
**(3) MSMQ 安装**  
  
控制面板 -> 添加/删除Windows 组件 -- 选择消息队列 - 下一步

这将在你的系统中安装MSMQ，然后你可以通计算机管理来进行确认。

控制面板 -> 管理工具 -> 计算机管理 -> 服务和应用 -> 消息队列，

你将看到出栈队列，私有队列，系统队列，触发器。  
  
**(4) 消息类型**

MSMQ 支持两种类型的消息: XML 和二进制, 下面的例子分别使用XML的二进制消息。  
  
**(5) MSMQ 架构(命名空间集成关系)**  
  
System   
  Messaging  
    Message  
    MessageQueue  
    MessageEnumerator  
    MessageType  
    MessagePriority  
    ...  
  
**MSMQ 示例程序**

示例 1 (使用 XmlMessageFormatter)

|  |
| --- |
| static void Main(string[] args)  {      MessageQueue messageQueue = null;      if (MessageQueue.Exists(@".\Private$\MyQueues"))      {          messageQueue = new MessageQueue(@".\Private$\MyQueues");          messageQueue.Label = "Testing Queue";      }      else      {          messageQueue = MessageQueue.Create(@".\Private$\MyQueues");          messageQueue.Label = "Newly Created Queue";      }      messageQueue.Send("First ever Message is sent to MSMQ", "Title");        messageQueue.Formatter = new XmlMessageFormatter(new string[] { "System.String" });      //iterating the queue contents      foreach (Message msg in messageQueue)      {          string readMessage = msg.Body.ToString();          Console.WriteLine(readMessage);          //process message      }      //after all processing delete the messages      messageQueue.Purge();      Console.ReadKey();  } |

示例 2 (使用 BinaryMessageFormatter)

|  |
| --- |
| class Program      {          static void Main(string[] args)          {              CreateQueue(@".\Private$\ImageQueue");              SendMessage();              ReceiveMessage();              Console.ReadKey();          }            public static void CreateQueue(string queuePath)          {              try              {                  if (!MessageQueue.Exists(queuePath))                  {                      MessageQueue.Create(queuePath);                  }                  else                  {                      Console.WriteLine(queuePath + " already exists.");                  }              }              catch(MessageQueueException e)              {                  Console.WriteLine(e.Message);              }          }            //Send an image to a queue, using the BinaryMessageFormatter.          public static void SendMessage()          {              try              {                  //Create new bitmap.                  //File must be in \bin\debug or \bin\release folder                  //Or a full path to its location should be given                    MessageQueue myQueue = new MessageQueue(@".\Private$\ImageQueue");                  Image myImage = Bitmap.FromFile("MyImage.jpg");                  Message msg = new Message(myImage, new BinaryMessageFormatter());                  myQueue.Send(msg);              }              catch (Exception e)              {                  Console.WriteLine(e.Message);              }          }            //Receive a message that contains an image.          public static void ReceiveMessage()          {              try              {                  MessageQueue myQueue = new MessageQueue(@".\Private$\ImageQueue");                  myQueue.Formatter = new BinaryMessageFormatter();                  Message myMessage = myQueue.Receive();                  Bitmap myImage = (Bitmap)myMessage.Body;                  myImage.Save("NewImage.jpg", ImageFormat.Jpeg);              }              catch (Exception e)              {                  Console.WriteLine(e.Message);              }          }      } |

在这个例子中我们将一个JPG图片文件存储到MSMQ队列，它可以在接下来的步骤中被接收然后使用。

运行这个程序并确认是否"NewImage.Jpg" 文件在Debug或者Release 文件夹中被创建。  
  
  
希望这篇文章可以给你一个简要的关于MSMQ的了解。