MarshallSoft Client Mailer
Library for Delphi

Programmer's Manual

(MCM4D)

Version 5.2

April 10, 2019

This software is provided as-is.
There are no warranties, expressed or implied.

MarshallSoft Computing, Inc.
Post Office Box 4543
Huntsville AL 35815

Email: info@marshallsoft.com
Web: www.marshallsoft.com

TABLE OF CONTENTS

1 Introduction

The **MarshallSoft Client Mailer for Delphi** provides the capability to send **personalized** email to your clients or customers **directly** from your Delphi application program. The most current version of the **MarshallSoft Client Mailer for Delphi** can be found at http://www.marshallsoft.com/mcm4d.htm.

The "MarshallSoft Client Mailer for Delphi Programmer Manual" contains Delphi programming specific details such as compilers, compiling and running example programs.

The **MarshallSoft Client Mailer** (MCM) functions can be called from any 32-bit Delphi or 64-bit Delphi program. MCM4D has been tested with all 32-bit and 64-bit versions of Delphi including:

- Borland Delphi (2.0, 3.0, 4.0, 5.0. 6.0 and 7.0)
- Borland Delphi 8 for .NET
- Borland Delphi 2005 & 2006
- Borland Turbo Delphi
- Codegear Delphi 2007
- Embarcadero Delphi 2009 & 2010 and Embarcadero Delphi XE through XE10

The **MarshallSoft Client Mailer** runs under 32-bit and 64-bit Windows through Windows 10. The **MarshallSoft Client Mailer DLLs (MCM32.DLL/MCM64.DLL)** can also be used from any language (Visual C++, .NET, Visual Basic, VB.NET, ACCESS, EXCEL, VBA, Visual FoxPro, dBASE, Xbase, etc.) capable of calling the Windows API. Win32 and Win64 DLLs are provided.

Since all programming language versions of MCM (C/C++, Visual Basic, Delphi, FoxPro, dBase, and XBase) use the identical DLLs (MCM32.DLL & MCM64.DLL), once one programming language version is purchased, the **MarshallSoft Client Mailer** can be used with all supported languages.

Regardless of the programming language used, both the email letter to be sent (text or HTML) and the list of recipients are regular ASCII text files. These letter files can be created with any text editor (or HTML editor for HTML files). The recipient list file can be created with either a text editor, exported from a database (such as Access, FoxPro, dBase, Xbase++) or a spreadsheet (such as Excel), or created by a program such as VBA code in Access.

See Section 3 for details on the **SendMail** example program and Section 4 for details on the **GetReply** example program

1.1 MCM Features

The features of the **MarshallSoft Client Mailer (MCM)** include:

- Send personalized email **directly** from an application, programmable database or spreadsheet.
- Uses macro substitution strings to personalize outgoing letters (Also called email merge).
- Can send plain text, rich text, or HTML formatted letters.
- Supports multiple attachments.
- Can handle arbitrarily large lists of recipients.
- Detects duplicate email addresses.
- Supports multiple lists of recipients that should always be skipped.
- Supports ASCII, UTF8, and ISO_8859 character sets.
- Supports WIN_1250, WIN_1252, and WIN_1255 character sets.
- Can use up to 32 connections simultaneously for fast email delivery.
- Can process "undeliverable mail" replies.
- Can process multiple user replies ("REMOVE", "CONFIRM", etc)
- Has extensive error detection and logging capability.
- Includes **SendMail** example program (with source) to send email.
- Includes **GetReply** example program (with source) to read replies.
- Includes **TestMCM** example program (with source) to verify installation and provide version information.
- Implemented as a standard Windows DLL, which will work with all versions of Windows.
- Supports both 32-bit and 64-bit Windows and includes Win32 and Win64 DLLs.
- Works with 32-bit and 64-bit Windows through Windows 10.
- License includes one year of technical support and downloadable updates for the MCM DLL.
- Royalty free distribution (no run-time fees) with your compiled application
- License covers all programming languages (MCM4C, MCM4D, MCM4VB, MCM4FP, MCM4DB, MCM4XB).
- Supports all 32-bit and 64-bit versions of Borland /CodeGear/Embarcadero Delphi, from Delphi 2 through Delphi 8, Delphi 2005 through Delphi 2010, Embarcadero Delphi XE through X10 and Delphi Turbo.

1.2 Documentation Set

There are five manuals in Adobe PDF format for the **MarshallSoft Client Mailer.**

- **Tutorial Manual**: Introduces the basic functionality and overview of the **MarshallSoft Client Mailer.**
- **Servers Manual**: Covers background information on SMTP & POP3 servers.
- **User Manual**: Covers information that is not programming language specific (letter & list preparation, program logic, purchasing, performance, SSL, etc.).
- **Reference Manual**: Contains details for each individual MCM function specific for each programming language (Delphi, C/C++, VB, etc.).
- **Programmer Manual**: Contains programming language (Delphi, C/C++, etc.) specific information such as compiling and running example programs.

The manuals can be found in the DOCS subdirectory in the **MarshallSoft Client Mailer** file structure when it is installed.

- **Tutorial Manual**: mcm_tutorial.pdf or online at
  http://www.marshallsoft.com/mcm_tutorial.pdf.
- **Servers Manual**: mcm_servers.pdf or online at
  http://www.marshallsoft.com/mcm_servers.pdf.
- **User Manual**: mcm_users.pdf or online at
  http://www.marshallsoft.com/mcm_users.pdf.
- **Reference Manuals** mcm4d_reference.pdf or
  online at http://www.marshallsoft.com/mcm4d_reference.pdf.
- **Programmer Manuals** mcm4d_programmer.pdf or
  online at http://www.marshallsoft.com/mcm4d_programmer.pdf.

Links to the above manuals can also be found online on the **MarshallSoft Client Mailer for Delphi** product page

http://www.marshallsoft.com/mcm4d.htm

1.3 Filenames

Filenames follow the following convention:

- Recipient list filenames begin with "List" and end with ".txt"
- Text letter filenames begin with "Letter" and end with ".txt"
- HTML letter filenames begin with "Letter" and end with ".htm"
- HTML template filenames begin with "Template" and with ".htm"
- Background tile filenames begin with "Tile" and end with ".jpg"
- Image filenames begin with "Image" and end with ".jpg"

1.4 MCM Files

- **mcm32.dll** 32-bit MCL Dynamic Link Library (DLL) file
- **mcm64.dll** 64-bit MCL Dynamic Link Library (DLL) file
- **mcm32.pas** 32-bit Delphi function declaration & constants file
- **mcm64.pas** 64-bit Delphi function declaration & constants file

1.5 KeyCode and Edition Key

The MCM DLLs, MCM32.DLL and MCM64.DLL, each have a keycode encoded within them. The keycode is a 9 or 10-digit decimal number (unless it is 0), and will be found in the file KEYCODE.PAS. The keycode for the evaluation version is 0. The developer will receive a new key code after registering. The KEYCODE (NOT your Customer Id) is passed to **mcmAttach**.

Note that starting with Version 5.0 the Edition Code is no longer used.

1.6 Installation

(1) Before installation of **MCM4D**, your Delphi compiler should already be installed on your system and tested.

(2) Unzip MCM4D52.ZIP using any Windows unzip program.

(3) Run the installation program SETUP.EXE which will install all MCM4D files, including copying MCM32.DLL & MCM64.DLL to the Windows directory. Note that no DLL registration is required.

After SETUP is run, the MCM4D files are copied to the directory specified (default \MCM4D). Three sub-directories are created, as follows:

```
DOCS - Documentation files
APPS - Examples program and files
DLLS - MCM32.DLL & MCM64.DLL
```

1.7 Uninstalling

Uninstalling MCM4D is very easy.

First, run UNINSTAL.BAT (or UINSTALL.BAT), which will delete MCM32.DLL/MCM64.DLL from the Windows directory, typically C:\WINDOWS for Windows XP through Windows 10. Next delete the MCM4D project directory created when MCM4D was installed.

1.8 Purchase

Refer to the MCM User’s Manual (**`mcm_users.pdf`** or online at http://www.marshallsoft.com/mcm_users.pdf) for detailed purchasing information.

MCM4C can be purchased at

http://www.marshallsoft.com/order.htm.

1.9 Adding MCM to a Project

32-bit

If you are running Delphi for Win32 (Delphi 2 and above), copy MCM32.PAS to the same directory as your application program.. Also copy the files KEYCODE.PAS to this same directory. You will find these files in the APPS directory (folder) created when you ran SETUP, usually C:\MCM4D\APPS.

Next, add a reference to the files copied above to your "uses" clause in your application program. For example,

```
uses
  mcm32, keycode,...
```

You can leave 'keycode' out above if you put your numerical keycode value (found in `KEYCODE.PAS`) directly into the call to `mcmAttach`. For example,

```
  {pass the key code }
  Code := mcmAttach(123456789,0, ...) < 0 then …
```

Lastly, add a reference to `MCM32` to your project file (*.DPR). For example,

```
Uses
  mcm32 in 'mcm32.pas', ...
```

64-bit

If you are creating a 64-bit executable with Delphi (Delphi XE), copy MCM64.PAS to the same directory as your application program.

Also copy the files KEYCODE.PAS to this same directory. You will find these files in the APPS directory (folder) created when you ran SETUP, usually C:\MCM4D\APPS.

Next, add a reference to the files copied above to your "uses" clause in your application program. For example,

```
Uses
  mcm64, keycode,...
```

You can leave 'keycode' out above if you put your numerical keycode value (found in `KEYCODE.PAS`) directly into the call to `mcmAttach`. For example,

```
  {pass the key code }
  Code := mcmAttach(123456789,0, ...) < 0 then …
```

Lastly, add a reference to `MCM64` to your project file (*.DPR). For example,

```
Uses
  mcm64 in 'mcm64.pas', ...
```

2 MCM Units

There are two MCM units: MCM32.PAS and MCM64.PAS

2.1 MCM32.PAS

The MCM32.PAS unit contains MCM constants and function declarations necessary to access MCM32.DLL. Note that:

1. Strings are always declared as AnsiStrings.
2. Only integers and PAnsiChar (pointer) variables can be passed directly to MCM functions.

All functions in the MCM32.PAS unit begin with "mcm".

2.2 MCM64.PAS

The MCM64.PAS unit contains MCM constants and function declarations necessary to access MCM64.DLL. Note that:

3. Strings are always declared as AnsiStrings.
4. Only integers and PAnsiChar (pointer) variables can be passed directly to MCM functions.

All functions in the MCM64.PAS unit begin with "mcm".

3 Delphi Versions

Applications written with Delphi link with the same identical DLL's as for applications written in all other supported languages, such as C/C++ and Visual Basic.

Beginning with Delphi 2003, Delphi has two "personalities": (1) Win32 Delphi and (2) Delphi for .NET. Win32 Delphi programs are a continuation of the Delphi language as seen in earlier versions of Delphi. Delphi .NET is a version of Delphi designed to use the Microsoft .NET Framework.

3.1 Delphi 1

Delphi version 1 generates Win16 code. MCM32.DLL is a 32-bit DLL and cannot be called from 16-bit application programs such as those compiled by Delphi 1.

3.2 Delphi 2

Delphi version 2 and above generates WIN32 code. Therefore, applications written using Delphi 2 will link with MCM32.DLL. Strings can be much larger than 255 bytes.

Delphi 2 seems to have a problem with some of the string functions. Although the default is "large strings", some of the string functions (such as StrPas) copy only 255 bytes.

3.3 Delphi 3, 4, 5, and 6.

There are no known Delphi problems impacting our example programs in Delphi version 3 and above. Applications written using Delphi 3 and above will link with MCM32.DLL

3.4 Delphi 7.

Beginning in Delphi 7, the filename of a unit must match the unit name.

3.5 Delphi 2005 through XE-7

Delphi 2005 through Delphi XE are Borland/Codegear/Embarcadero’s latest Delphi product with support for both Win32 and Microsoft .NET Framework. Application programs compiled with Delphi 2005, 2006, 2007, 2009, 2010, and XE through XE-7 can create 32-bit applications using MCM32.DLL. Application programs compiled with XE2 and above can create 64-bit applications using MCM64.DLL.

4 TestMCM.Example Program

The **TestMCM** example program verifies that Windows can find MCM32.DLL at runtime, and it displays the MCM version and related information.

The project filename is Test_Prj.dpr

5 SendMail. Example Program

The **SendMail** example program sends a personalized letter to each recipient. The logic is as follows:

```
Step  1: Read configuration parameters from SendMail.ini
Step  2: Attach MarshallSoft Client Mailer
Step  3: Open the letter to be sent
Step  4: Open the list of recipients
Step  5: Open skip lists
Step  6: Connect to SMTP server
         LOOP: Step 7: Merge letter with next recipient
               Step 8: Send the mail
Step  9: Wait for all channels to complete
Step 10: Close connection on all channels
```

If you are not familiar with the above logic, read the MCM Tutorial, **mcm_tutorial.pdf,** and the MCM User's Manual, **mcm_users.pdf**.

The **SendMail** example program is very straight-forward and extensively documented. Open the SendMail program (**Send_Pgm.pas)** and look through the code.

To merge mail, but not actually send it, check the box

```
[X] Merge Only
```

once the "Send" pane is displayed.

If your security software scans outgoing email, you may need to disable this feature since some security software does not handle multiple connections correctly.

The project filename is SendMail_Prj.dpr

6 GetReply. Example Program

The **GetReply** example program reads server and recipient replies. The logic is as follows:

```
Step 1: Read configuration parameters from GetReply.ini
Step 2: Attach MarshallSoft Client Mailer
Step 3: Open bounced email file
Step 4: Open reply files
Step 5: Connect to the POP3 server
        LOOP: Step 6: Read next reply
Step 7: Close connection to server
```

If you are not familiar with the above logic, read the MCM Tutorial, **mcm_tutorial.pdf,** and the MCM User's Manual, **mcm_users.pdf**.

The **GetReply** example program is very straight-forward and extensively documented. Open the GetReply program (**GetReply_Pgm.pas)** and look through the code.

Delete Codes

GetReply divides incoming email into three mutually exclusive classes:

1. Failure messages ("bounced email") from the server.
2. User reply message (e.g., "REMOVE email-address").
3. Everything else.

As each email is read when mcmReadReply is called

```
   function mcmReadReply(SubjectBuffer, BufferSize, DeleteCode)
```

the 'DeleteCode' argument specifies if an email is to be deleted after it is read.

DeleteCode	Action
1	Delete if message is a failure message
2	Delete if message is a recipient reply message
4	Delete if message is not one of the above.

The above DeleteCodes can be added. For example, pass 3 to specify that failure messages (DeleteCode = 1) and reply messages (DeleteCode = 2) are to be deleted, but nothing else.

The project filename is GetReply_Prj.dpr

7 MakeStyled. Example Program

The **MakeStyled** example program reads a text email file and creates a basic HTML formatted email file.

The filename of the HTML formatted file will be the (input) text filename with ".htm" concatenated to the end of it.

The project filename is Styled_Prj.dpr

7 Revision History

Version 1.0: September 28, 2010.

- The official release of MCM version 1.0 for Delphi.

Version 2.0: January 11, 2011

- Added MCM_GET_TIME_STAMP to mcmGetString()
- Call mcmOpenList (NIL) or pass empty string to rewind recipient list.
- mcmOpenLetter, mcmOpenList, mcmOpenBounce, mcmOpenReply, mcmOpenSkip allow both filenames and pathnames to be passed.
- Added mcmSetString(MCM_SET_CC_ADDRESS) & mcmSetString(MCM_SET_BCC_ADDRESS)
- Added MCM_GET_AUTH_VERS_MAJOR, MCM_GET_AUTH_VERS_MINOR, & MCM_GET_CID
- Added MCM_SET_DUPLICATE_DETECT and MCM_SET_SERVER_TIMEOUT.
- The DLL runs indefinitely (does not stop working after one year).
- Maintenance updates are free.
- Activation Code Server allows up to 3 concurrent users with one activation file.

Version 3.0: May 19, 2011

- Removed requirement for activation code!
- Added mcmUtility function.
- Maximum channels increased to 32
- Fixed problem: GetReply handles email addresses with trailing whitespace
- Added MCM_GET_CUSTOMER_ID
- Added MCM_GET_ALLOWED_CHANNELS
- Added MCM_GET_ALLOWED_LIST_SIZE
- Added MCM_GET_ALLOWED_SKIP_FILES
- Added MCM_GET_ALLOWED_REPLY_FILES
- Added MCM_GET_EDITION
- Added MCM_GET_REGISTRATION
- Allow tilde ~ and carot ^ as delimiters in recipient list file.

Version 3.1: January 19, 2012

- Added "UNSOLICITED BULK EMAIL" as bounce subject when receiving mail.
- Increased (macro delimiter) field size from 64 to 256 characters.
- Added MCM_ALLOW_EMPTY_FIELDS. Allows recipient list fields to be empty.
- Allow insertion of files by use of @@filename in recipient list file.
- Set letter substitution delimiter to backquote or backslash with mcmSetInteger(MCM_SET_MACRO_DELIMITER, delimiter).
- Example: mcmSetInteger(MCM_SET_MACRO_DELIMITER, '`').
- Added mcmSetProxySSL(), which allows MCM to connect to servers requiring SSL.
- Added new merge code MCM_STAT_WITH_EMPTY_MACRO_STRING macro (1st list line) is empty.
- Added new merge code MCM_STAT_WITH_EMPTY_LIST_STRING entry in recipient list is empty.
- Added new stat code MCM_STAT_WITH_EMPTY_LIST_STRING counts "MCM_STAT_WITH_EMPTY_LIST_STRING" errors.

Version 4.0: October 11, 2012

- Added macro delimiters tilde ~, caret ^, and backquote `.
- Ignore Tab chars on list macro line unless tab is the delimiter character.
- Added MCM_GET_LAST_EMAIL_SENT to mcmGetString().
- Moved letter headers to separate header file (e.g.: letter.hdr).
- Added mcmOpenHeader() function to open header file.
- Allow comments (line starting with #) in header section of letter header files.
- Changed default macro delimiter from % (percent) to ` (backquote)
- Allow recipient list to have a single entry (no delimiters)
- Added "Header: string" to headers. Used to specify a user header.
- Fixed problem in mcmRelease - was sometimes releasing SEE prematurely.
- Option to automatically load (letter) header file: mcmSetInteger(MCM_AUTO_LOAD_HEADER_FILE, 1)

Version 5.0: October 31, 2014

- Added 64-bit DLL (MCM64.DLL).
- Added project files through Delphi XE7.
- Fixed problem in which not all reply strings (i.e., "REMOVE") were being detected.
- Fixed problem in which comments were counted as recipients in recipient list.
- Added MCM_SMTP_ONLY and MCM_POP3_ONLY error messages
- Increased MRG_STRING_SIZE from 5000 to 65536 (64K) characters.
- Removed edition codes (no limit on list size).
- Added MCM_GET_NBR_EMAILS_DELETED to mcmGetInteger().
- Fixed problem in which incoming emails were not deleted as requested.
- Added MCM_ADD_BOUNCE_STRING to mcmSetString (adds 'bounce' subject)
- Added mcmGetString2()
- Added MCM_GET_BOUNCE_STRING to mcmGetString2()
- Fixed attachment problem when using two lists (1st w/attachment, 2nd w/o)
- Updated to SEE version 7.2.6

Version 5.1: October 6, 2016

- Fixed buffer overflow problem.
- Strings exceeding limits in recipient list are detected.
- Fixed problem with accumulating CC addresses.
- Updated to SEE version 7.4.1
- Additional details written to log file
- Added MCM_SET_RCPT_TRACE_FILE to mcmSetString() to generate file of server commands & responses.
- Added MCM_GET_LAST_RESPONSE to mcmGetString() to get last email server response.
- mcmRelease forces all files closed.
- Allow vertical bar '|' to specify multiple addresses in recipients file, as for example
- `mailAddress, CCAddress, Attachments`
- `mike@mike4.com, pam@mike4.com|lauren@mike4.com, widget_red.txt`
- Time of day added to "Today is" string written to log file
- Added new HTML templates.
- Added function mcmSearch() that searches skip files for a specified string.

Version 5.2: April 10, 2019

- Warnings written if MCM_LOG_OFF was selected.
- Added diagnostics to merge code.
- Added MCM_NOT_TEXT_FILE error code (as @@ files are expected to be text files).
- Fixed problem in mcmRelease in which slot 0 not freed.
- Allow '#' comments after entries in recipient macro list file.
- Updated see32.obj and see64.obj to SEE version 8.0.4.
- Writes SEE version & build to MCM log file.
- Added mcmMakeStyled() function that creates an HTML file from an ASCII text file.
- Added MakeStyled example program that calls function mcmMakeStyled().
- Added MCM_CANNOT_CREATE_FILE error code.