TN1010-2b: CSMail Getting Started Guide for Visual Basic Developers

Contents

1 Introduction 1
2 Using CSMail in Visual Basic 1
3 Something didn't work! 4
4 What Next? 5

1 Introduction

Getting started with a new control or library can be a frustrating experience, it often seems that most of the development effort goes into accomplishing the simplest things! You're a developer and you want to send or receive email from your application and you don't want to spend an eternity doing it – this guide is for you.

The output from the program goes to the Visual Basic Immediate Window – you might find it helpful to arrange things on screen so that you can see the immediate window while the program is running.

The example has been designed so that it that will work first time for the vast majority of developers but networks are idiosyncratic beasts and security measures and network errors crop up all the time. I've tried to anticipate the obstacles you're most likely to run into - ESMTP authentication and Proxy Servers. The cause of some other errors may be clear, or hinted at, in the error messages from the servers and your network administrator may be able to help you with these.

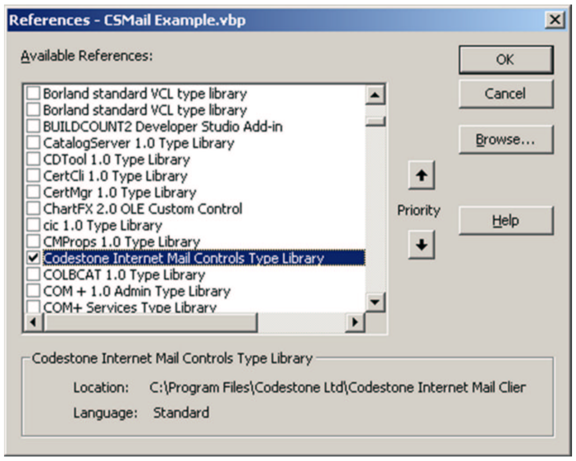
2 Using CSMail in Visual Basic

2.1 Add a reference to the library

- Fire up the Visual Basic IDE and create a new project.
- Open the project references dialog (Project|References on the main menu)
- The CSMail library is listed as 'Codestone Internet Mail Controls Type Library' - tick the checkbox and click on OK to close the project references dialog.

Visual Basic now knows about the control, auto-completion will work on the objects, properties and methods and the context sensitive help will be enabled.

Figure 1 – Add a reference to the library



References - CSMail Example.vbp

Available References:

- ☐ Borland standard VCL type library
- ☐ Borland standard VCL type library
- ☐ BUILDCOUNT2 Developer Studio Add-in
- ☐ CatalogServer 1.0 Type Library
- ☐ CDTool 1.0 Type Library
- ☐ CertCli 1.0 Type Library
- ☐ CertMgr 1.0 Type Library
- ☐ ChartFX 2.0 OLE Custom Control
- ☐ cic 1.0 Type Library
- ☐ CMProps 1.0 Type Library
- ☑ Codestone Internet Mail Controls Type Library
- ☐ COLBCAT 1.0 Type Library
- ☐ COM + 1.0 Admin Type Library
- ☐ COM+ Services Type Library

Codestone Internet Mail Controls Type Library

Location: C:\Program Files\Codestone Ltd\Codestone Internet Mail Clier

Language: Standard

2.2 *Declare SMTP and POP3 Client variables*

- Add the declarations in Figure 1 to the (General) (Declarations) sections of the main form in your new project.
- If you're reading this document on line you can probably cut and paste the code rather than typing it all!
- Change the Const declarations to values that are appropriate to you local network, the values for will be the same as those you use for your desktop email client.

Figure 2 – Declaring the variables

```
Dim MySmtp As New SMTPClient
Dim MyPop3 As New POP3Client

Const sSMTPServer = "127.0.0.1"
Const sPOP3Server = "localhost"

Const sSender = "username@localhost"
Const sRecipient = "username@localhost"

Const sPOP3User = "username"
Const sPOP3Pass = "secret"
```

2.3 *Add code to send an Email*

- Add a button to the main form of your project and add the code in Figure 3 to the button's Click event handler.
- Run the program and click on the new button.
- Congratulations! You've just sent your first email with the CSMail library

2.3.1 What to do if the program didn't work as expected

- Don't Panic – these things happen – there could be a network problem or your system administrator may have configured specific security options on your SMTP Server.
- Take a look in the Immediate window in the Visual Basic IDE and note the error messages – I've included a list of the most common errors in section 3.1.

Figure 3 – Sending an email

```
Private Sub cmdSendMail_Click()
On Error GoTo Error

Dim MyMsg As New Message

MyMsg.Subject = "Rock and Roll!"
MyMsg.To(1) = sRecipient
MyMsg.From(1) = sSender

Debug.Print"Connecting to SMTP Server."
MySmtp.Connect sSMTPServer

Debug.Print "Sending Message ."
MySmtp.SendMessage MyMsg
MySmtp.Close

Debug.Print "Message Sent!"

Exit Sub

Error:
    Debug.Print "Error #" & Err.Number
    Debug.Print "       " & Err.Description
    If Err.Number = errSMTP Then
        Debug.Print "SMTP Info"
        Debug.Print MySmtp.LastServerResponseCode
        Debug.Print MySmtp.LastServerResponseString
        End If

End Sub
```

2.4 Add code to receive email

- Add another button to the main form of your project and add the code in Figure 4 to the button's Click event handler.
- Run the program and click on the new button.
- Inspect the contents of the Immediate Window in the Visual Basic IDE – you should see a list of message senders and subjects.

2.4.1 What to do if the program didn't work as expected

- Don't Panic – these things happen – there could be a network problem or your system administrator may have configured specific security options on your POP3 Server.
- Take a look in the Immediate window in the Visual Basic IDE and note the error messages – I've included a list of the most common errors in section 3.2.

Figure 4– Receiving a email

```
Private Sub cmdReceiveMail_Click()
On Error GoTo Error

Debug.Print "Connecting to POP3 Server..."
MyPop3.Connect sPOP3Server, sPOP3User, sPOP3Pass

Debug.Print "Receiving Messages..."
MyPop3.RetrieveMessages
MyPop3.Close False
Debug.Print "Messages Received!"

For Each Msg In MyPop3.Messages
    Debug.Print Msg.From(1) & " : " & Msg.Subject
    Next

Exit Sub

Error:
    Debug.Print "Error #" & Err.Number
    Debug.Print "       " & Err.Description
    If Err.Number = errPOP Then
        Debug.Print "POP3 Info"
        Debug.Print MyPop3.LastServerResponseString
        End If

End Sub
```

3 Something didn't work!

3.1 The code in Section 2.3 (Add code to send an Email) failed

3.1.1 ErrConnect (80000205H)

- You may have mistyped the name of server in the code from Figure 2.
- The server may be down – check to see if you can connect using your standard desktop email program.
- The network may have failed – check to see if you can connect using your standard desktop email program.
- The server may be protected by a firewall – contact your network administrator and see if you can access SMTP and POP3 servers directly in the early stages of development.
- If you can't access a SMTP server directly you can add proxy support to the server. Figure 5a shows the code to add for a SOCKS 5 proxy. You should change the Dim statement to create the appropriate ProxyType (SOCK4ProxyInfo, SOCK5ProxyInfo object or SMTPProxyInfo) and change the address and authentication details appropriately.

Figure 5a –Adding SOCKS Proxy support for SMTP

```
...

MyMsg.From(1) = sSender

Dim MyProxy As New SOCK5ProxyInfo
MyProxy.ProxyAddress = "[ProxyAddress]"
MyProxy.ProxyUsername = "[ProxyUsername]"
MyProxy.ProxyPassword = "[ProxyPassword]"

MySmtp.SetProxyInfo MyProxy


Debug.Print"Connecting to SMTP Server."
MySmtp.Connect sSMTPServer

...
```

3.1.2 errSMTP (80000206H)

- EMSTP authentication may be required on the SMTP server. Figure 5b shows how to change the code to support ESMTP.
- Look in the SMTP settings on you desktop email program and find your SMTP authentication details, it is not unusual for these to be the same as your POP3 authentication details.

Figure 5b –Adding support for ESMTP

```
...

Debug.Print"Connecting to SMTP Server."
~~MySmtp.Connect sSMTPServer~~
MySmtp.ConnectESMTP sSMTPServer,"[User]","[Pass]"

...
```

3.2 *The code in Section 2.4 (Add code to receive email) failed*

3.2.1 ErrConnect (80000205H)

- You may have mistyped the name of server in the code from Figure 2.
- The server may be down – check to see if you can connect using your standard desktop email program.
- The network may have failed – check to see if you can connect using your standard desktop email program.
- The server may be protected by a firewall – contact your network administrator and see if you can access SMTP and POP3 servers directly in the early stages of development.
- If you can't access a POP3 server directly you can add proxy support to the server. Figure 5a shows the code to add for a SOCKS 5 proxy. You should change the Dim statement to create the appropriate ProxyType (SOCK4ProxyInfo, SOCK5ProxyInfo object or POP3ProxyInfo) and change the address and authentication details appropriately.

Figure 5b –Adding SOCKS Proxy support for POP3

```
...

On Error GoTo Error

Dim MyProxy As New SOCK5ProxyInfo
MyProxy.ProxyAddress = "[ProxyAddress]"
MyProxy.ProxyUsername = "[ProxyUsername]"
MyProxy.ProxyPassword = "[ProxyPassword]"

MyPOP3.SetProxyInfo MyProxy

Debug.Print "Connecting to POP3 Server..."
MyPop3.Connect sPOP3Server, sPOP3User, sPOP3Pass

...
```

4 What Next?

In this guide I've concentrated on overcoming the initial hurdles in using the CSMail Library in your application – now it's over to you to create some cool production code!

4.1 *Explore the message object*

It's quite interesting to note that, in contrast with other libraries, the Message object provides a far richer interface than the SMTP and POP3 client objects. This is one of the key design elements of the library – we designed the Message object to provide a simple yet flexible implementation of the MIME standard.

Technote TN1010-11-1 'HTML Message with Images' serves as a good primer for the message object model – even if you're not particularly interested in including graphical spectaculars in your emails!

Technote TN1010-11-1 is available at:

`http://codestone.co.uk/software/docs/csmail/tn1010-11-1.pdf`

4.2 *Advanced Server options – ESMTP and Proxy Servers*

These are features your customers are probably going to be looking for in the email features of your applications.

ESMTP provides an authentication method for SMTP sessions – see the ConnectESMTP method on page 18 of the manual and contrast it with the Connect method I've used in this example. ESMTP authentication is often required when the SMTP client and servers are located on different domains. You will probably want to provide a facility for your customers to specify their SMTP authentication details.

Proxy Servers provide a firewall between email clients and the Internet; SOCKS proxies are quite common and rather cool. The Proxy Objects Developer's reference on pages 30-34 of the manual has examples of using SOCKS proxies and the standard POP3 and SMTP proxies. You will probably want to provide a facility for your customers to specify the type of the proxy, if any, and the appropriate authentication details.

4.3 Error Handling

It is, unfortunately, the very nature of Internet enabled applications that unexpected conditions will occur, servers will go down, users will enter the wrong credentials and conscientious network administrators will apply inconvenient, but necessary, security measures.

The CSMail library uses the standard Visual Basic/Automation mechanism for notifying the developer of errors and you should handle in these in the same way as you would handle any other error in your code.

Appendix B of the manual lists the errors that the library may raise together with some notes on how you might handle them.