

HowTo SMTP in C#

by Randy Charles Morin

This series of articles is written to show the user how to write TCP/IP based client applications using C# on Microsoft's new dotNET framework. This is the first article in this series.

The first killer applications on the Internet were email and netnews. Email on the Internet was developed using two simple Internet protocols, SMTP and POP3. The first two articles in this series, I'll present to you two classes for implementing SMTP and POP3 clients. In the third article in this series, I'll present to you one class for implementing a NNTP client.

The SMTP or Simple Mail Transfer Protocol is described in RFC 821 [<http://www.ietf.org/rfc/rfc0821.txt>]. This application protocols is used to send email over the Internet. A few years back, I wrote two articles on sending SMTP messages in C++ [<http://www.kbcafe.com/articles/smtp.html>] [<http://www.kbcafe.com/articles/html.smtp.html>]. This article shows how to do the same programming in Microsoft's new C# language.

The dotNET framework already contains an SMTP class in the System.Web.Mail namespace called SmtMail. This class is sufficient for sending email over the Internet and I would not suggest that the class I'm presenting in this article is any better or worse. Let's just say that it is different. If you can get away with using the dotNET SmtMail class, then I suggest you do just that. The only advantage of my class is that it is open source and let me suggest that the SmtMail class in dotNET has a few more features.

My motivation in writing this article is not to try and write a better SMTP class, but rather to show how to write TCP/IP based clients in C#. And here goes.

I started by writing a custom exception for my C# Smt class. Whenever my C# class encounters faults, I'll throw an instance of this exception. The exception class is presented in Listing 1.

Listing Error! Bookmark not defined.: Exception Class

```
public class SmtException : System.Exception
{
    private string message;
    public SmtException(string str)
    {
        message = str;
    }
    public string What()
    {
        return message;
    }
};
```

Our class will be inherited from the System.Net.Socket.TcpClient class in the dotNET framework. This class provides all the base functionality that is required to do TCP/IP programming. Our class declaration is presented in Listing 2.

Listing Error! Bookmark not defined.: Class Declaration

```
public class Smt : System.Net.Sockets.TcpClient
```

Our class will have eight data members. The from data member is the sender of the email. The to, cc and bcc data members represent the recipients of the email. The subject data member is the subject of the email. Either bodyText or bodyHtml will represent the body of the email. The server data member represents the SMTP server where our client is sending the message. The data member definitions are presented in Listing 3.

Listing Error! Bookmark not defined.: Data Members

```
public class Smtplib : System.Net.Sockets.TcpClient
{
    public string from = null;
    public ArrayList to;
    public ArrayList cc;
    public ArrayList bcc;
    public string subject = null;
    public string bodyText = null;
    public string bodyHtml = null;
    public string server = null;
}
```

The to, cc and bcc data members are arrays, in order to allow multiple recipients of all three types. All three arrays must be created in the constructor of our class. The constructor of our class is presented in Listing 4.

Listing Error! Bookmark not defined.: Constructor

```
public Smtplib()
{
    to = new ArrayList();
    cc = new ArrayList();
    bcc = new ArrayList();
}
```

The Send method does all the work in our class. Once you've set the subject, body, from and recipient data members, you simply call the Send method and the SMTP communication is initiated. I won't explain the details of the SMTP protocol here, but rather encourage the user to step through the code, read the SMTP RFC and read my previous articles on SMTP. The Send method is presented in Listing 5.

Listing Error! Bookmark not defined.: Send Method

```
public void Send()
{
    string message;
    string response;

    Connect(server, 25);
    response = Response();
    if (response.Substring(0, 3) != "220")
    {
        throw new SmtplibException(response);
    };

    message = "HELO me\r\n";
    Write(message);
    response = Response();
    if (response.Substring(0, 3) != "250")
    {
        throw new SmtplibException(response);
    }

    message = "MAIL FROM:<" + from + ">\r\n";
    Write(message);
    response = Response();
    if (response.Substring(0, 3) != "250")

```

```
{
    throw new SmtException(response);
}

foreach ( string address in to )
{
    try
    {
        message = "RCPT TO:<" + address + ">\r\n";
        Write(message);
        response = Response();
        if (response.Substring(0, 3) != "250")
        {
            throw new SmtException(response);
        }
    }
    catch( SmtException e)
    {
        System.Console.WriteLine("{0}", e.What());
    }
}

foreach ( string address in cc )
{
    try
    {
        message = "RCPT TO:<" + address + ">\r\n";
        Write(message);
        response = Response();
        if (response.Substring(0, 3) != "250")
        {
            throw new SmtException(response);
        }
    }
    catch( SmtException e)
    {
        System.Console.WriteLine("{0}", e.What());
    }
}

foreach ( string address in bcc )
{
    try
    {
        message = "RCPT TO:<" + address + ">\r\n";
        Write(message);
        response = Response();
        if (response.Substring(0, 3) != "250")
        {
            throw new SmtException(response);
        }
    }
    catch( SmtException e)
    {
        System.Console.WriteLine("{0}", e.What());
    }
}

message = "DATA\r\n";
Write(message);
response = Response();
if (response.Substring(0, 3) != "354")
{
    throw new SmtException(response);
}

message = "Subject: " + subject + "\r\n";
foreach ( string address in to )
{
    message += "To: " + address + "\r\n";
}
}
```

```

foreach ( string address in cc )
{
    message += "Cc: " + address + "\r\n";
}

message += "From: " + from + "\r\n";
if (bodyHtml.Length > 0)
{
    message += "MIME-Version: 1.0\r\n"
        +"Content-Type: text/html;\r\n"
        +"    charset=\"iso-8859-1\"\r\n";
    message += "\r\n" + bodyHtml;
}
else
{
    message += "\r\n" + bodyText;
};
message += "\r\n.\r\n";
Write(message);
response = Response();
if (response.Substring(0, 3) != "250")
{
    throw new SmtException(response);
}

message = "QUIT\r\n";
Write(message);
response = Response();
if (response.IndexOf("221") == -1)
{
    throw new SmtException(response);
}
}
}

```

The Send method uses three methods, the Connect, Write and Response methods. The Connect method is inherited from the TcpClient class and establishes a TCP connection between our client and a TCP server. The other two methods are implemented in our class and described in the next paragraphs.

The Write method writes data to the socket. The Write method is presented in Listing 6.

Listing Error! Bookmark not defined.: Write Method

```

public void Write(string message)
{
    System.Text.ASCIIEncoding en = new System.Text.ASCIIEncoding() ;

    byte[] WriteBuffer = new byte[1024] ;
    WriteBuffer = en.GetBytes(message) ;

    NetworkStream stream = GetStream() ;
    stream.Write(WriteBuffer,0,WriteBuffer.Length);
}

```

It must be remembered that C#'s native string type is not based on the ASCII characters, but rather is based on Unicode. Thus in order to send messages on SMTP we must first convert the string input parameter to an array of ASCII bytes. We use the ASCIIEncoding class in dotNET to make this transformation. Then we retrieve the socket stream using the GetStream method inherited from the TcpClient class and write the bytes to the stream.

The Response method receives data from the socket. The Response method is presented in Listing 7.

Listing Error! Bookmark not defined.: Response Method

```
public string Response()
{
    System.Text.ASCIIEncoding enc = new System.Text.ASCIIEncoding();
    byte []serverbuff = new Byte[1024];
    NetworkStream stream = GetStream();
    int count = stream.Read( serverbuff, 0, 1024 );
    if (count == 0)
    {
        return "";
    }
    return enc.GetString( serverbuff, 0, count );
}
```

The Response method begins by retrieving the bytes from the stream and then converts the incoming ASCII bytes to our native C# string type.

Now that we have an SMTP class, you might be wondering how you use it. To send an SMTP message, you simply instantiate an instance of the class, set the appropriate data members and call the Send method.

Listing Error! Bookmark not defined.: Usage

```
static void Main(string[] args)
{
    try
    {
        kbcafe.Smtp smtp = new kbcafe.Smtp();
        smtp.server = "smtp.xxx.com";
        smtp.from = "yyy@xxx.com (zzz)";
        smtp.subject = "Hello World";
        smtp.bodyHtml = "<HTML><BODY>Hello World</BODY></HTML>";
        smtp.to.Add("yyy@xxx.com");
        smtp.Send();
    }
    catch( kbcafe.SmtpException e)
    {
        System.Console.WriteLine("{0}", e.What());
    }
}
```

If you reuse this code, replace the xxx, yyy and zzz parameters by valid values. A valid from address may be "randy@kbcafe.com (Randy Charles Morin)". Please don't use this address, I get enough emails already.

About the Author

Randy Charles Morin is the Chief Architect of SportMarkets Development from Toronto, Canada and lives with his wife, Bernadette and two kids, Adelaine and Brayden in Brampton, Canada. He is the author of the KB Cafe.com website [<http://www.kbcafe.com>], many programming books and many articles.