

Tutorial

Creating SMS-based Applications

Part-2: *Sending and Receiving SMS in VB .NET*

By Haider Ali

May 2008

Introduction

In part-1 I gave some basics of SMS, the AT commands for SMS and then demonstrated sending/receiving SMS from HyperTerminal. Before reading this article I hope you have studied part 1 (Using HyperTerminal to send and receive SMS messages) of this series.

The SMS gateway application has two main components:

- Sending SMS from the PC.
- Reading the received SMS.

In this article I will write the first simple application that sends and receives SMS.

Messages stored in GSM/GPRS Modem or Mobile Phone Storage

Before going into the details of how to send and receive an SMS let me give some explanation of the messages stored in the mobile phone storage. The message stored in the mobile phone memory has the following four parts.

- Mobile Phone No.
- Message No.
- Status
- Message

The Mobile Phone No. is the number from which the message is received or to which the message is sent or to be sent.

The Message No. is a serial number given to each message.

The Status can be one of the following four:

STO UNSENT:	Message stored for sending but not yet sent.
STO SENT:	Message Sent.
STO UNREAD:	Message received but not yet read.
REC READ:	Message received and read.

The Message contains the actual message text.

For convenience I have created a SMSMessage class in this application. The code for the SMSMessage class is given in the following listing.

```
Public Class SMSMessage
    Public MessageNo As Integer
    Public Status As String
    Public MobileNo As String
    Public Message As String

    Public Sub New(ByVal mobileNo As String, ByVal message As String, _
        Optional ByVal MessageNo As Integer = 1,
        Optional ByVal status As String = "STO")
        Me.MessageNo = MessageNo
        Me.Status = status
        Me.MobileNo = mobileNo
        Me.Message = message
    End Sub
End Class
```

Send SMS from VB .NET application

For sending SMS I have created a simple form **frmSendSMS**. The screenshot of the form is given in figure-1.



Figure 1: Send SMS Form

This simple form contains three simple controls:

- Textbox **tbTo** contains the mobile no. to which the message has to be sent.
- Textbox **tbMessage** contains the text message that has to be sent.
- Button **btnSend** when clicked sends the message to the destination phone no.

The following code is associated with the click event of the **btnSend** button:

```
Imports System.Threading

Public Class frmSendSMS

    Private Sub btnSend_Click(ByVal sender As System.Object,
                              ByVal e As System.EventArgs)
        Handles btnSend.Click

        Dim Incoming, SubStr As String
        Dim MessageNo, N As Integer

        Using comm As IO.Ports.SerialPort =
            My.Computer.Ports.OpenSerialPort("COM4", 9600)
            comm.DtrEnable = True
            comm.Write("AT" & vbCrLf)
            comm.Write("AT+cmgf=1" & vbCrLf)

            ' chr(34)= "
            comm.Write("AT+CMGW=" & Chr(34) & MobNo & Chr(34) & vbCrLf)
```

```

        comm.Write(tbMessage & Chr(26) & vbCrLf) ' Ctrl-Z Character

        Thread.Sleep(2000)

        ' Find the Message No. in Mobile Memory
        Incoming = comm.ReadExisting
        N = Incoming.IndexOf("+CMGW:")
        N = N + 6
        SubStr = Incoming.Substring(N)
        MessageNo = Val(SubStr)

        comm.WriteLine("AT+CMSS=" & MessageNo & vbCrLf)
        Thread.Sleep(2000)

        comm.Close()
    End Using
End Sub
End Class

```

Now let me explain each line of the code.

```
Imports System.Threading
```

We need to import the System.Threading namespace in order to use Threas.sleep method.

```
Dim Incoming, SubStr As String
Dim MessageNo, N As Integer
```

Some varials that will be used in the application.

```
Using comm As IO.Ports.SerialPort =
    My.Computer.Ports.OpenSerialPort("COM4", 9600)
```

When we read/write to a serial port we need to open it for reading/writing this method is used to initialize a serial port at COM4. If your GSM/GPRS modem is installed on another port, say COM3, you should replace the port name to the actual comm port in this code.

```
comm.DtrEnable = True
```

Before reading/writig to serial port we need to enable its data transfer ready mode.

```
comm.Write("AT" & vbCrLf)
comm.Write("AT+cmgf=1" & vbCrLf)
```

These are the actual AT commands that we used to issue in HyperTerminal.

AT+CMGF=1 is the command to instruct the mobile phone to operate in text mode.

```
comm.Write("AT+CMGW=" & Chr(34) & MobNo & Chr(34) & vbCrLf)
```

AT+CMGW is used to write an SMS text message to the message storage of the mobile phone. MobNo is the mobile phone no. to which the message has to be sent. Note that the mobile phone no. has to be enclosed in double quotes for which the chr(34) is used in the code.

```
comm.Write(tbMessage & Chr(26) & vbCrLf)
```

After issuing the AT+CMGW command the serial port waits for the user input and accepts the input until Ctrl-Z is entered. In this line of code the message in the tbMessage textbox is written to the serial port and the Ctrl-Z character represented by Chr(26) is sent. This line writes the message to the message storage of the mobile phone.

Now that the message has been successfully written to the mobile phone memory, we need to send it. For sending a message from the mobile phone memory to the destination mobile no. we need to find the Message No. of the message in the mobile phone storage. This is done by the following code:

```
Incoming = comm.ReadExisting
N = Incoming.IndexOf("+CMGW:")
N = N + 6
SubStr = Incoming.Substring(N)
MessageNo = Val(SubStr)
```

When the MessageNo is identified we can use it to send the message to the destination. This is done by the following code:

```
comm.WriteLine("AT+CMSS=" & MessageNo & vbCrLf)
```

And that's it. We have successfully sent a message to the destination from our application.

Receive SMS in VB .NET

Receiving a SMS in VB .NET is actual retrieving the received SMSs in a mobile phone or GSM/GPRS modem. For this purpose I have created a simple form frmReceiveSMS whose screenshot is given as follows: