MessageNet

Software API Overview

This document outlines the common API calls available to integrate with the MessageNet SMS gateway. Combinations of these calls can result in a comprehensive, robust 2 way SMS service enabling 3rd party solutions to leverage of the power and cost effective nature of SMS.

1. .NET/SOAP

Location: www.messagenet.com.au/dotnet

Function: LodgeSMSMessage

Description: Lodge SMS message into the MessageNet system.
Phonenumber may be nnnn;nnnn;nnnn for multiple messages

Arguments: username as string
pwd as string
phonenumber as string
phonemessage as string

Returns: <string xmlns="http://www.messagenet.com.au/dotnet">
Message sent successfully.
</string>
Anything else is error.

Example: LodgeSMSMessage(user1,password1,61412123456,Test MessageNet)

Function: LodgeSMSMessageWithReply

Description: Lodge SMS message into the MessageNet system specifying where to send any reply.
Phonenumber may be nnnn;nnnn;nnnn for multiple messages

Arguments: username as string
pwd as string
phonenumber as string
phonemessage as string
replytype as string
replypath as string

Returns: <string xmlns="http://www.messagenet.com.au/dotnet">
Message sent successfully.
</string>
Anything else is error.

Example: LodgeSMSMessageWithReply(user1,password1,61412123456,Test MessageNet,WWW,user1)
ReplyType acceptable values
EMAIL,TCP/IP,WWW,HTTP
ReplyPath acceptable values
Email address, IP address:Port, Web username, URL for POST
If specifying a URL, arguments MUST include the following
phone=#FROM#&msg=#MSG#&date_sent=#SENTDTS#&date_received=#RCVDDTS#

Function: **LodgeSMSMessageWithReplyXML**

Description: Lodge SMS message into the MessageNet system specifying where to send any reply.

Arguments: username as string
pwd as string
phonenumber as string
phonemessage as string
replytype as string
replypath as string

Returns: XML document.

```
<MESSAGENET>
<MSGID> nnnnnnnnnnnnnnnnnnn</MSGID>
<ERROR>
        <ERRORMSG></ERRORMSG>
        <ERRORNUM>n<ERRORNUM>
</ERROR>
</MESSAGENET>
```

Where n = 0 for success , > 0 if error, see ErrorMsg

Example: LodgeSMSMessageWithReplyXML (user1,password1,61412123456,Test MessageNet,HTTP,www.mydomain.com/mysms.asp?id=123)

ReplyType acceptable values
EMAIL,TCP/IP,WWW,HTTP
ReplyPath acceptable values
Email address, IP address:Port, Web username, URL for POST
If specifying a URL, arguments MUST include the following
phone=#FROM#&msg=#MSG#&date_sent=#SENTDTS#&date_received=#RCVDDTS#

Function: **GetMessageStatusXML**

Description: Get the status of a previous lodged message from.

Arguments: username as string
pwd as string
msgid as string (as returned from LodgeSMSMessageWithReplyXML

Returns: XML document.

```
<MESSAGENET>
<MSGID> nnnnnnnnnnnnnnnnnnn</MSGID>
<STATUS>xxxx</STATUS>
<ERROR>
        <ERRORMSG></ERRORMSG>
        <ERRORNUM>n<ERRORNUM>
</ERROR>
</MESSAGENET>
```

Where n = 0 for success , > 0 if error, see ErrorMsg
Where xxxx is one of SENT,ERROR,DELIV,ACTIVE,TRYMAX

Example: GetMessageStatusXML (user1,password1,MSGID)

Function: **LodgeXML**

Description: Lodge one or more messages via an XML document.
Arguments: passed XML – see appendix A for schema

XML document format is:

```
<?xml version="1.0" encoding="utf-8"?>
<messagenet>
        <userid username="?????" password="?????" />
        <senddts>15-Jun-2006 15:30</senddts>
        <reply replytype="HTTP"
replypath="http://www.yoursmsreplyURL.com.au/smsreply?vid=x&cid=y" />

        ' note in the above URL arguments vid=x&cid=y. This enables you to track
        ‘ reply messages as these arguments will be given back to your web server
        ‘ when a SMS reply message is received.
        ' MessageNet will append the follwoing to your URL
        ‘ &device=<from mobile>&dts=<dd-MMM-yyyy hh:mm:ss>&message=<txt>

        <msgblock>
                <device>614nnnnnn</device>
                <message>test message 1 &lt; 2</message>
        </msgblock>
        <msgblock>
                <device>614nnnnn</device>
                <message>test message 2</message>
        </msgblock>

</messagenet>
```

Returns: XML document.

```
        <messagenet>
        <error>
                <errormsg></errormsg>
                <errornum>n<errornum>
        </error>
        </ messagenet >
        Where n = 0 for success , > 0 if error, see ErrorMsg
```

Function: **LodgeXMLbyArg**

Description: Lodge one or more messages via an XML document.
Arguments: strdocXML as string

strdocXML format is:

```
<?xml version="1.0" encoding="utf-8"?>
<messagenet>
        <userid username="?????" password="?????" />
        <senddts>15-Jun-2006 15:30</senddts>
        <reply replytype="HTTP"
replypath="http://www.yoursmsreplyURL.com.au/smsreply?vid=x&cid=y" />

        ' note in the above URL arguments vid=x&cid=y. This enables you to track
        ‘ reply messages as these arguments will be given back to your web server
        ‘ when a SMS reply message is received.
        ' MessageNet will append the follwoing to your URL
        ‘ &device=<from mobile>&dts=<dd-MMM-yyyy hh:mm:ss>&message=<txt>

        <msgblock>
                <device>614nnnnnn</device>
                <message>test message 1 &lt; 2</message>
        </msgblock>
        <msgblock>
                <device>614nnnnn</device>
                <message>test message 2</message>
        </msgblock>

</messagenet>
```

Returns: XML document.

```
        <messagenet>
        <error>
                <errormsg></errormsg>
                <errornum>n<errornum>
        </error>
        </ messagenet >
        Where n = 0 for success , > 0 if error, see ErrorMsg
```

Function: **LodgeFAXMessageXML**

Description: Lodge 1 to 3 files for fax transmission via an XML document.
See appendix B for sample client code.

Arguments: username as string
pwd as string
faxnumber inclusive of area code as string
faxsubject as string
coverpagetext as string
Returns: XML document.

```
<messagenet>
<msgid> nnnnnnnnnnnnnnnnnnn</ msgid >
<error>
        <errormsg></ errormsg >
        <errornum>n<errornum>
</ error >
</ messagenet >
```

Where n = 0 for success , > 0 if error, see ErrorMsg

Function: **LodgeMMSMessage**

Description: Lodge MMS message into the MessageNet system
Arguments: username as string
pwd as string
phonenumber as string
mmssubject as string
mmsmessagetext as string
filename as string
mmscontentbase64 as string

Returns: <string xmlns="http://www.messagenet.com.au/dotnet">
Message sent successfully.
</string>
Anything else is error.

Example: LodgeMMSMessage(user1,pwd,61412123456,Subject,Test,fred.jpg,Base64String)

Sample VB.NET code

Add web reference to:
http://www.messagenet.com.au/dotnet/lodge.asmx?op=LodgeMMSMessage

```
Dim mms As au.com.messagenet.www.Lodge

Dim b64 As String
Dim sPic As Object
Dim strReply As String

sPic = CreateObject("ADODB.Stream")
With sPic
        .Open()
        .Type = 1
        .LoadFromFile("c:\temp\dog1.jpg")
End With

b64 = Convert.ToBase64String(sPic.read)
mms = New au.com.messagenet.www.Lodge
strReply = mms.LodgeMMSMessage("username", "pwd", "04nnnnnnnn", "Subject
text", "Msg text", "dog.jpg", b64)

mms = Nothing
sPic = Nothing

' Evaluate strReply and act accordingly

If InStr(strReply, "ERROR") > 0 Then
        ' Send Error
Else
        ' Send OK
End If
```

2. HTTP GET

Location
www.messagenet.com.au/dotNet/Lodge.asmx/LodgeSMSMessageWithReply?Username=string&Pwd=string&PhoneNumber=string&PhoneMessage=string&ReplyType=string&ReplyPath=string

Function: **LodgeSMSMessageWithReply**

Description: URL GET method for lodging SMS messages
PhoneNumber may be nnnn;nnnn;nnnn for multiple messages

Arguments: See above URL

Returns: On success

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://www.messagenet.com.au/dotnet">Message sent
successfully.</string>
```

On Error

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://www.messagenet.com.au/dotnet">Message send failed.
Unable to send to: nnnnnnn;</string>
```

Notes:

ReplyType and ReplyPath optional
ReplyType acceptable values
EMAIL,TCP/IP,WWW,HTTP
ReplyPath acceptable values
Email address, IP address:Port, Web username, URL for POST
If specifying a URL, arguments MUST include the following
phone=#FROM#&msg=#MSG#&date_sent=#SENTDTS#&date_received=#RCVDDTS#

3. HTTP POST

Location www.messagenet.com.au/dotNet/Lodge.asmx/LodgeSMSMessageWithReply

Function: **LodgeSMSMessageWithReply**

Description: URL POST method for lodging SMS messages
PhoneNumber may be nnnn;nnnn;nnnn for multiple messages

Arguments:

Username=string&Pwd=string&PhoneNumber=string&PhoneMessage=string&ReplyType=string&ReplyPath=string

Returns: On success

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://www.messagenet.com.au/dotnet">Message sent
successfully.</string>
```

On Error

```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://www.messagenet.com.au/dotnet">Message send failed.
Unable to send to: nnnnnnn;</string>
```

Notes:

ReplyType and ReplyPath optional
ReplyType acceptable values
EMAIL,TCP/IP,WWW,HTTP
ReplyPath acceptable values
Email address, IP address:Port, Web username, URL for POST
If specifying a URL, arguments MUST include the following
phone=#FROM#&msg=#MSG#&date_sent=#SENTDTS#&date_received=#RCVDDTS#

4. TCP/IP

The MessageNet SMS OCX (**MessageNetSMSControl)** enables developers to send and receive SMS messages via the MessageNet gateway using TCP/IP.

The OCX utilizes Microsoft's MSWinsock control for TCP/IP sending and receiving

MessageNetSMSControl OCX Properties:

MessageNetSMSControl1.**Username**
Username assigned by MessageNet on account setup

MessageNetSMSControl1.**Password**
Password assigned by MessageNet on account setup

MessageNetSMSControl1.**MessageNetIP**
IP address of the MessageNet SMS gateway

MessageNetSMSControl1.**MessageNetPort**
TCP/IP Port of the MessageNet SMS gateway

MessageNetSMSControl OCX Functions:

MessageNetSMSControl1.**MsgNetListen**(**LocalIP** as string, **LocalPort** as long)

Start listening for TCP/IP data containing inbound SMS message traffic from MessageNet.
The LocalIP and LocalPort MUST be accessible through your firewall via the Internet. Please check you can connect to this IP and Port using Hyperterminal.

When an SMS is received it will be displayed in the OCX's control window and it will also raise an event **SMSReceived** (**Sender** As String, **Message** As String)

MessageNetSMSControl1.**MsgNetSend**(**MobilePhone** as string, **Message** as string) as Long

Send the given message to the given mobile phone.
Return codes:

0	Username and Password have not been set prior to call
1	Message SENT OK
3	Socket connect error
4	Timeout Receiving Response from gateway
5	Message NOT sent. Check username, password & account
9	Socket connect error

MessageNetSMSControl OCX Events:

Public Event SMSReceived(Sender As String, Message As String)
An SMS was received

Public Event SendingSocketError(ErrorMsg As String)
An error has occurred on the sending socket

Public Event ReceivingSocketError(ErrorMsg As String)
An error has occurred on the receiving socket

Example:

Create a project in VB and include the MessageNetSMSControl component.
Drop the OCX on a form. You can size the component and use the interface for testing if required.

Add 2 command buttons to the form and the following code:

```
Private Sub Command1_Click()

   MessageNetSMSControl1.MsgNetListen "<your IP address>", <your port>

End Sub

Private Sub Command2_Click()

        Dim result as long
        result = MessageNetSMSControl1.MsgNetSend("0412345678","TEST OCX")
        if result = 1 then
                MsgBox “Sent!”
        Else
                MsgBox “Error”
        End if

End Sub

Private Sub Form_Load()

   MessageNetSMSControl1.Username = "<username supplied by MessageNet>”
   MessageNetSMSControl1.Password = "<password supplied by MessageNet>”

End Sub

Private Sub MessageNetSMSControl1_SMSReceived(Sender As String, Message As String)

        MsgBox “SMS received from:”  & Sender & ":" & Message

End Sub
```

5. TCP/IP RAW

Location tcp.messagenet.com.au:(7000,7001,9090)

Description: TCP/IP packet lodgement of SMS messages.
Connect to tcpsms.messagenet.com.au:(7000,7001,9090)
Lodge messages (see below)
Disconnect

Packet Structure: Field delimiter <CR>: All fields are ASCII strings

ReplyAddr<CR>
ReplyType<CR>
Username<CR>
1<CR>
2<CR>
Company<CR>
SystemName<CR>
ToName <CR>
MobileNum <CR>
GSM<CR>
0<CR>
0<CR>
0<CR>
Subject & " " & Password<CR>
Message<CR>

Sample:

```
ReplyAddr = "???@??.??.??"
ReplyType = "EMAIL"
Username = "????"       ' Assigned by MessageNet
Password = "???"        ' Assigned by MessageNet
Company = "Any Text"
SystemName = "Any Text"
ToName = "Any text"
MobileNum = "041???????"
Subject = "TCPIP TEST"
Message = "MESSAGE TEXT"
```

See ftp://www2.messagenet.com.au/api/win/ibscript.txt

Returns:

On Success
<ACK><CR>Lodge OK

On Error
<NAK><CR>Lodge Error

6. TCP/IP Reply Packet Structure

Description: This is the packet structure when a reply message is sent as a result of setting the ReplyType to TCP/IP and the ReplyPath to IP address:Port from any of the above Lodge functions.
Connect
Send
Wait for reply of ‘OK” or ‘ERROR’
Close

Packet Structure: Field delimiter <CR>: All fields are ASCII strings

<CR>
1<CR>
Senders Mobile number<CR>
To MessageNet Mobile Number<CR>
Sent DTS<CR>
Received DTS<CR>
Text

Sample:
Connect
Send
<CR>
1<CR>
61412000000<CR>
61427890900<CR>
1-Jun-2004 17:00<CR>
1-Jun-2004 17:01<CR>
Text reply message text
Receive ON or ERROR
Close

Example of 2 Way service via HTTP POST

Description: A business would like to integrate SMS into their current business system. The are required to send an SMS from an internal user to a destination users mobile phone. When the receiving party receives the SMS message, the can send a text message back to the original sender and that message will be delivered to the senders preferred user interface, e.g. Web, e-mail.

Solution: Store the senderID, destination mobile and SMS message in your local database. Have an application that reads this database and call the HTTP POST URL to lodge the message with the MessageNet gateway.

www.messagenet.com.au/dotNet/Lodge.asmx/LodgeSMSMessageWithReply

Post data:

Username=**string**&**Pwd**=**string**&**PhoneNumber**=**string**&**PhoneMessage**=**string**&**ReplyType**=**HTTP**&**ReplyPath**=**<see below customer URL>**

Customer URL:

This URL informs the MessageNet SMS server that any reply to the message will be sent back to this URL *including any arguments given in the original POST call*.

Example: (Must be URL encoded)

http://x.com/ReplySMS.asp?phone=#FROM#&message=#MSG#&mysenderid=235

The #FROM# and #MSG# are required fields and get replaced by *the MessageNet gateway server* when the POST is made back to your server, in this example x.com. Your ReplySMS.asp page should capture the arguments and determine the original sender via the mysenderid that was originally passed in the POST call. The reply message can now be delivered back to the original sender via your internal applications.

Appendix A

LodgeXML schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="messagenet">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="userid">
          <xs:complexType>
            <xs:attribute name="username" type="xs:string" use="required" />
            <xs:attribute name="password" type="xs:string" use="required" />
          </xs:complexType>
        </xs:element>
        <xs:element name="senddts" type="xs:string" />
        <xs:element name="reply">
          <xs:complexType>
            <xs:attribute name="replytype" type="xs:string" use="required" />
            <xs:attribute name="replypath" type="xs:string" use="required" />
          </xs:complexType>
        </xs:element>
        <xs:element maxOccurs="unbounded" name="msgblock">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="device" type="xs:string" />
              <xs:element name="message" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

APPENDIX B

Sample code for LodgeFAXMessageXML

Add web reference to:
http://www.messagenet.com.au/dotnet/lodge

```
Imports system.io
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click

        Dim fName As String
        fName = "c:\temp\mediarelease.doc"
        doFileUploadViaWebService(fName)

    End Sub
    Public Sub doFileUploadViaWebService(ByVal fName As String)

        Dim oFile As File                    'File object
        Dim oFileStream As FileStream        'Holds the Stream of the File

        Dim oWS As New MessageNet.Lodge      'the webservice

        Dim strReturn As String = ""         'Value returned from the webservice

        Dim XMLResult As New Xml.XmlDocument

        Dim elXML As Xml.XmlElement

        Try
            'Make sure that the file exists before trying to upload
            If File.Exists(fName) = False Then Exit Sub

            'Open the file and break it down into a file stream
            oFileStream = File.Open(fName, IO.FileMode.Open, IO.FileAccess.Read)

            'declare the byte array of the file
            Dim oFileByte(oFileStream.Length - 1) As Byte
            Dim oFileByte2(0) As Byte
            Dim oFileByte3(0) As Byte

            'break the file into bytes and place into the byte object
            oFileStream.Read(oFileByte, 0, oFileStream.Length)

            'upload the file to the webservice
            elXML = oWS.LodgeFAXMessageXML("<username>", "<pwd>", "<Fax number with
area code>", "Subject", "Cover page text.", fName, oFileByte, vbNull, oFileByte2,
vbNull, oFileByte3)

            XMLResult.LoadXml(elXML.OuterXml)

            'display the results
            If XMLResult.GetElementsByTagName("errornum").ItemOf(0).InnerXml <> "0"
Then
                Throw New Exception(elXML.OuterXml)
            Else
                MsgBox("File Uploaded Successfully")
            End If

        Catch ex As Exception
            'display errors
            MsgBox(ex.Message.ToString)
        Finally
            'cleanup
            oWS = Nothing
            oFile = Nothing
            oFileStream.Close()
        End Try
    End Sub
End Class
```