



TextAnywhere Developer Reference Guide

Web Service, HTTP and SMTP SMS Integration

TextAnywhere Limited
The Oast House
9 Brewery Court
High Street
Theale
Berkshire
RG7 5AH
UK

UK tel: 08451 221 302
Intl tel: +44 8451 221 302
Email: DevSupport@TextAnywhere.net

www.textanywhere.net

TextAnywhere Limited
TextAnywhere Developer Reference Guide
Web Service, HTTP and SMTP SMS Integration

Edition 4.8, June 2008

© 2003 - 2008 TextAnywhere Limited. All Rights Reserved.

Information in this document is subject to change without notice. Companies, names and data used in examples herein are fictional unless otherwise noted. No part of this document may be reproduced or transmitted in any form by any means, electronic or mechanical, for any purpose, without the express written permission of TextAnywhere Limited.

TextAnywhere™, TextOnline™, TextEmail™, TextAlert™, TextCampaign™, TextPremium™ and TextInbound™ are trademarks of TextAnywhere Limited. Microsoft, Windows, Windows NT and Microsoft .NET are trademarks of Microsoft Corporation. Other names are the trademarks of their respective owners.

TEXTANYWHERE LIMITED DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE INSTRUCTIONS CONTAINED IN THIS GUIDE.

IN NO EVENT SHALL TEXTANYWHERE LIMITED BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS), EVEN IF TEXTANYWHERE LIMITED HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME COUNTRIES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.

TABLE OF CONTENTS

SECTION ONE – OVERVIEW	1
1.1 INTRODUCTION	1
1.2 DOCUMENT CONTENT	1
1.3 ONLINE DEVELOPER CENTRE	2
SECTION TWO – PARAMETER DESCRIPTIONS	3
2.1 INTRODUCTION	3
2.2 PARAMETERS	3
2.3 SENDING MESSAGES TO UK LANDLINE PHONES	6
SECTION THREE –WEB SERVICE FOR SMS INTEGRATION	8
3.1 INTRODUCTION	8
3.2 SEND MESSAGE TO MULTIPLE RECIPIENTS AND RECEIVE REPLIES	8
3.3 QUERY MESSAGE DELIVERY STATUS	9
3.4 GET TEXT MESSAGE REPLIES	10
3.5 GET REPLIES TO A PARTICULAR TEXT MESSAGE	11
3.6 DELETE TEXT MESSAGE REPLIES	11
3.7 GET INBOUND TEXT MESSAGES	12
3.8 FORMAT NUMBER	13
3.9 CHECK NUMBER	13
3.10 SERVICE TEST	13
3.11 CHECK CREDITS	14
SECTION FOUR –HTTP(S) RECEIVER FOR SMS INTEGRATION	15
4.1 INTRODUCTION	15
4.2 SEND MESSAGE TO RECIPIENTS AND RECEIVE REPLIES	15
4.3 QUERY MESSAGE STATUS FOR MULTIPLE MESSAGES	17
4.4 GET TEXT MESSAGE REPLIES	18
4.5 GET REPLIES TO A PARTICULAR TEXT MESSAGE	18
4.6 DELETE TEXT MESSAGE REPLIES	19
4.7 GET INBOUND TEXT MESSAGES	19
4.8 FORMAT NUMBER	20
4.9 CHECK NUMBER	20
4.10 SERVICE TEST	21
4.11 CHECK CREDITS	21
SECTION FIVE – SMTP SERVICE	22
5.1 INTRODUCTION	22
5.2 SENDING A TEXT MESSAGE	22
5.2.1 MAIL2TXT.NET	23
5.2.2 MAIL2TXT160.NET	23
5.2.3 MAIL2TXTID.NET	24
5.2.4 MAIL2TXT160ID.NET	24
5.3 RECEIVING A REPLY	25
5.4 RECEIVING TEXTINBOUND MESSAGES AS EMAIL	26
SECTION SIX – CODE SAMPLES AND DEVELOPER CENTRE	27
6.1 INTRODUCTION	27
6.2 DEVELOPER CENTRE	27

SECTION SEVEN – SHORT CODE AND REVERSE BILLING SERVICES	28
7.1 INTRODUCTION	28
7.2 SETTING-UP YOUR SHORT CODE SERVICE	28
7.3 RECEIVING INBOUND MESSAGES	29
7.4 SENDING PREMIUM MESSAGES (OPTIONAL)	29
7.5 HTTP(S) RECEIVER AND WEB SERVICE ACCESS	30
7.5.1 ACCESSING THE HTTP(S) RECEIVER	30
7.5.2 ACCESSING THE WEB SERVICE	30
7.6 AVAILABLE METHODS	30
7.6.1 RETRIEVE INBOUND SMS	30
7.6.2 DELETE RECEIVED SMS	32
7.6.3 SEND A PREMIUM REVERSE BILLED SMS	32
7.6.4 SUBSCRIPTION SERVICE: SEND AN SMS TO YOUR SUBSCRIBER GROUP	33
7.6.5 SUBSCRIPTION SERVICE: SEND AN SMS TO MEMBERS OF YOUR GROUP	33
7.6.6 SUBSCRIPTION SERVICE: RETRIEVE LIST OF SUBSCRIBERS	34
SECTION EIGHT – REGISTRATION AREA FOR PARTNERS' CLIENTS	35
8.1 INTRODUCTION	35
8.2 USE OF THE REGISTRATION PAGE	35
APPENDIX ONE – SUPERCEDED METHODS	36
A.1 INTRODUCTION	36
A.2 SEND MESSAGE TO A SINGLE RECIPIENT	36
A.3 QUERY MESSAGE STATUS FOR SINGLE MESSAGE	36

SECTION ONE – OVERVIEW

1.1 Introduction

This *Developer Reference Guide* provides developers with the tools to access the *TextAnywhere SMS Gateway* using a *Web Service*, *HTTP(S) Receiver* or *SMTP Service*, from within their own application or web-site environment

The following functions can be performed with the integration tools:

1. Send SMS text messages
2. Receive replies to those sent messages
3. Check the delivery status of sent messages
4. Receive proactively-sent inbound text messages (in conjunction with *TextInbound*)
5. Send and receive reverse-billed, text messages (in conjunction with *TextPremium*)

TextAnywhere elected to implement *Web Service*, *HTTP* and *SMTP* approaches to integration for the following reasons:

1. To provide the widest possible support for client development environments, applications and operating platforms;
2. To ensure that no TextAnywhere software was required to be embedded within the client's application or web environment;
3. To provide a very straightforward method of sending and receiving regular and premium SMS text messages.

Through using the *Web Service*, applications can communicate with the *TextAnywhere SMS Gateway*, using the SOAP (*Simple Object Access Protocol*) specification.

The *Web Service* and *HTTP(S) Receiver* have been designed to provide a rich set of functionality to send and receive messages from both applications and web environments.

The *SMTP Service* has been provided to enable applications that currently automate the sending of emails, to be able to send out text-message content, as emails, to the *TextAnywhere SMS Gateway*. The *SMTP Service* does not support the sending and receiving of premium (reverse billed) messages.

1.2 Document content

This *Developer Reference Guide* provides developers with the tools to access the *TextAnywhere SMS Gateway* using the *Web Service*, *HTTP(S) Receiver* or *SMTP Service*, from within their own application environment

Section Two of this guide details the parameters associated with using the *Web Service* and *HTTP(S) Receiver*.

Section Three of this guide provides details on using the *Web Service* for sending and receiving regular, non-premium SMS messages, **Section Four** addresses the use of the *HTTP(S) Receiver* and **Section Five** outlines the sending of text messages through the *SMTP Service*.

A link to code samples in a variety of development environments is provided in **Section Six**.

Sending and receiving text messages to and from Short Codes (also known as *Reverse Billing* or *Premium Content services*) is addressed in **Section Seven**. The *Web Service* and *HTTP(S)* methods are described in this section.

If you are developing an application where your clients will need to register with TextAnywhere to be able to activate the text-messaging component of your application, your clients can register on a cut-back *Client Registration* page. Further details of this function are available in **Section Eight**.

1.3 Online Developer Centre

In addition to this *Developer Reference Guide*, we also offer an online resource location, known as our *Developer Centre*. From the *Developer Centre* you have access to the following:

- Detailed information on the Web Service, HTTP(S) and SMTP APIs
- Explanation of the available *Methods* and *Parameters*
- Code samples in various development environments
- Access to developer forums for peer-level discussion

Our Developer Centre will provide you with the resources you need to help you easily and swiftly integrate text messaging in to your application. We have detailed information on the APIs we provide, as well as code samples, comprehensive downloadable documentation, and access to our developer forums.

You can access the Developer Centre by clicking either [here](#) or on the link below:

<http://developer.textanywhere.net/v2/home/Default.aspx>

SECTION TWO – PARAMETER DESCRIPTIONS

2.1 Introduction

The following section describes each of the parameters necessary for invoking the *Web Service* and *HTTP(S) Receiver*, referred to within Sections Three, Four and Seven.

2.2 Parameters

- Client_ID** Your unique account identifier provided by TextAnywhere (maximum of 50 characters). You can review (but not amend) this parameter by logging on to your online TextAnywhere account, and selecting the *Administration Panel* option.
- This parameter is used, in conjunction with *Client_Pass* (below), to authenticate your message-sending request.
- Client_Pass** Your account password (maximum of 50 characters). You can review (and amend) this parameter by logging on to your online TextAnywhere account, and selecting the *Administration Panel* option.
- This parameter is used, in conjunction with *Client_ID* (above), to authenticate your message-sending request.
- Client_Ref** This is the unique message reference (maximum of 50 characters) that you will use when sending a message to enable you to later retrieve the message's delivery status.
- This is set by your application when it invokes a method that requires this parameter. To enable you to check the delivery status of a particular sent message, the *Client_Ref* you use must be unique.
- Billing_Ref** This is the billing reference for the message that you are sending (maximum of 50 characters), that helps you track the number of messages sent. This parameter is generated by you.
- It is of value when your message sending application is sending messages on behalf of multiple applications, or clients, and you wish to track each application's message-sending volume.
- At the end of any month you can run a report from your online account that will show you how many messages were sent with each of the *Billing_Ref* values that you used.
- If you are sending messages on behalf of a single application and you have no need to track messages sent by *Billing_Ref*, then just use any value that you wish for this parameter.
- Body** The message content itself, with a maximum of 160 characters.

Connection

This is the class of SMS that you elect to send the message as. There are two classes; one for testing purposes and one for live usage:

- 1 *Test*: this is to assist you in initially integrating and then testing your message-sending application.

By sending a test message with *Connection* set to 1, your application will engage with our SMS Gateway, though your message will NOT be sent beyond our gateway.

Your application will receive a valid return code, which can be queried using an appropriate method. You are not charged for sending messages to this connection.

- 2 *Enterprise – low volume use*: highest, UK-sourced message delivery quality.

When sending a single message to up to 150 recipients, set the *Connection* parameter to 2 and make a single call of the *SendSMSEx* method.

- 3 Unused.

- 4 *Enterprise – high volume use*: highest, UK-sourced message delivery quality, identical in message quality to *Enterprise – low volume*.

When sending a single message to more than 150 recipients, set the *Connection* parameter to 4, and make multiple calls of the *SendSMSEx* method, waiting for the appropriate method Return Code, before calling the method again.

Originator

The *Originator* field essentially identifies the sender of the SMS message to the message recipient.

With a conventional mobile-to-mobile SMS message this is normally the phone number of the message sender, enabling the recipient to identify the user (by recognising the number) and, possibly, reply back.

When sending a message from the Developer Toolkit, you can either send a mobile number (which can be replied to) or any 11 character alphanumeric string (perhaps a company name), which cannot be replied to.

If you do elect to send a company name, for example, then the (up to) 11 characters will appear where traditionally you would see the phone number of the sender appear.

The *OType* parameter below determines whether you are sending a numeric (phone number) or alphanumeric (company name, for example) in the *Originator* parameter.

If a mobile phone number is sent, it must be formatted without the 0 and beginning with a + and the Country Code.

For example, for a UK mobile phone number of 07836 123 456, it would be formatted as +447836123456.

- OType** This parameter determines the type of *Originator* (described above) that is being sent:
- 0 To send a mobile phone number
 - 1 To send an alphanumeric string with a maximum of 11 characters for the *Originator*
- Destination** A formatted, single phone number of the message recipient. The phone number must be formatted in the international format, beginning with a + and the Country Code, and without any leading 0.
- For example, for a UK mobile phone number of 07836 123 456, it would be formatted as +447836123456.
- Destination phone numbers can include UK and international mobile phones, as well as UK landline numbers. Please see Section 2.3 below for further information on sending text messages to UK landline phones.
- DestinationEx** One or more formatted phone numbers of the message recipients, separated by commas. The phone numbers must be formatted in the international format, beginning with a + and the Country Code, and without any leading 0.
- For example, for a UK mobile phone number of 07836 123 456, it would be formatted as +447836123456.
- Each number in this parameter has to be comma separated:
- E.g. +4477845965452,+336456232125,+4479894561232
- Destination phone numbers can include UK and international mobile phones, as well as UK landline numbers. Please see Section 2.3 below for further information on sending text messages to UK landline phones.
- SMS_Type** Please set this parameter to 0.
- Reply_Type** With this parameter you decide whether you wish to receive replies to your sent messages or not. If you wish to enable replies, you must select the method by which you wish the replies to be delivered back to your application:
- 0 *No reply*: the message will be sent with the *Originator* that you have entered and replies to the message will not be possible
 - 1 *Reply to Web Service/HTTP Connector*: your *Originator* will be replaced by a TextAnywhere reply path number, with replies going to the Web Service or HTTP Receiver
 - 2 *Reply to Email*: your *Originator* will be replaced by a TextAnywhere reply number. Replies go to the specified email in *Reply_Data*, below
 - 3 *Reply to URL*: your *Originator* will be replaced by a TextAnywhere reply number. Replies go to the specified URL in *Reply_Data*, below

Reply_Data	<p>For the value given in the <i>Reply_Type</i> field, enter the following within the <i>Reply_Data</i> field:</p> <ol style="list-style-type: none">0 Provide an empty string1 Provide an empty string2 Provide an email address for replies to be sent to3 Provide a web URL. When a reply occurs, the SMS Gateway will contact this URL as follows: <code>http://URL?originator=+447787654321&destination=+447712345678&date=5/1/2004&Time=10:30&Body=Test&ClientRef=test</code>
SMS_Encoding	This field must be set to 0 (zero)
Inbound_Number	<p>This is the phone number or Short Code to retrieve inbound regular and premium messages:</p> <p><i>TextInbound:</i> This is the phone number of a TextInbound service, used for the inbound receipt of non-premium SMS messages</p> <p><i>TextPremium:</i> This is the Short Code and Keyword combination used to identify the premium content service from which to retrieve inbound premium content messages.</p>
Destination_Group	The group of subscribers to a premium content service
Destination_Numbers	<p>One or more formatted phone numbers of the premium content message recipients, separated by commas. The phone numbers must be formatted in the international format, beginning with +44, and without any leading 0.</p> <p>For example, 07836 123 456, would be formatted as +447836123456.</p> <p>Each number in this parameter has to be comma separated: E.g. +4477845965452,+336456232125,+4479894561232</p>
IDs	An integer that uniquely identifies a received premium content message
RBID	The <i>Reverse Billing Identifier</i> which identifies the single recipient for a premium content message to be sent to. This number has been received from the consumer's network operator.

2.3 Sending messages to UK landline phones

Regular, free-to-receive, text messages can be sent to UK landline phones. Premium, reverse-billed messages cannot be sent to UK landlines.

Depending on what type of phone the recipient has, the text message will either be displayed (if the recipient has either a text home phone or the Caller Display service), or the message will be read out.

When sending messages to BT landlines, please bear in mind the following:

- 1 The recipient must have a BT landline, even though they may use a different service provider, such as TalkTalk, to make and receive phone calls. BT physically delivers the SMS to the landline and does have agreements in place with some cable operators, such as Virgin Media. The BT service will not operate on an organisation's phone system – it is aimed at the home phone where one phone number is supported on one phone line.
- 2 The number you send the message to must be formatted in its full international format. For example (01234) 567 567, must be formatted as +441234567567.
- 3 You can only send a number with the message, rather than any 11 characters. These 11 characters would, when sent to a mobile, appear where the sender's number usually appears, showing, for example, *Acme Co Ltd*. The BT landline service only supports numbers in this field.
- 4 A successful *Delivered to Handset* delivery status for your message confirms only that the message has been received by BT, not that the message has been delivered or read out on the recipient's phone.
- 5 You can only send text messages to landlines between the hours of 8.00am and 10.00pm. It is your responsibility as the sender to conform to these times.

SECTION THREE –WEB SERVICE FOR SMS INTEGRATION

3.1 Introduction

Web Service name: TextAnywhere_SMS

Access Point: http(s)://ws.textanywhere.net/TA_SMS.asmx

WSDL The XML-based Web Services Description Language (WSDL) for the TextAnywhere SMS Web Service is provided at the Access Point. The Web Service methods, properties, input and output parameters, and how to connect to the Web Service are all documented.

The following section describes the available *TextAnywhere Web Service Methods*.

Each method includes the method name, a brief description of its function, the available parameters and the return codes.

3.2 Send Message to Multiple Recipients and Receive Replies

Name: SendSMSEx

Description: This function is used to send a single text message to multiple mobile phone numbers within one function.

When sending a single message to multiple recipients, please use the following process:

1. *For up to 150 recipients:* set the *Connection* parameter to 2 and make a single call of the *SendSMSEx* method
2. *For more than 150 recipients:* set the *Connection* parameter to 4, and make multiple calls of the *SendSMSEx* method, waiting for the appropriate method Return Code, before calling the method again

SendSMSEx is also used to support the reply capability of the Web Service.

When sending a text message to one or more recipients to which a reply can be made, you stipulate the method by which you wish to receive that reply. This is achieved when sending the message.

You can receive replies through the Web Service, by email to a given email address or on a given web page URL.

By using the *Reply_Type* and *Reply_Data* parameters you stipulate how the message reply is delivered back to your application.

Parameters: *SendSMSEx(Client_ID,Client_Pass,Client_Ref,Billing_Ref,Connection,Originator,OType,DestinationEx,Body,SMS_Type,Reply_Type,Reply_Data)*

Client_ID	String
Client_Pass	String
Client_Ref	String
Billing_Ref	String
Connection	Integer
Originator	String
OType	Integer
DestinationEx	String
Body	String

SMS_Type	Integer
Reply_Type	Integer
Reply_Data	String

Returns: The *Return Code* is a comma separated string of “*destination number:integer code*” for each number to which the message is sent.

For example: +44123456789:1,+44123456789:1,+44123456789:1

The *integer code* returned refers to the following:

1	SMS Sent
2	Authentication failed: Account not found
22	Authentication failed: Your account is currently suspended
3	SMS failed
31	SMS failed: Insufficient message credits on your account
311	SMS failed: This <i>Connection</i> does not exist
32	SMS failed: <i>Originator</i> format not recognised
321	SMS failed: <i>OType</i> invalid
33	SMS failed: <i>Destination(s)</i> format not recognised
34	SMS failed: <i>Reply_Type</i> invalid or <i>Reply_Data</i> empty
35	SMS failed: <i>Client_Ref</i> too long or empty (maximum 50 characters)
36	SMS failed: <i>Billing_Ref</i> too long or empty (maximum 50 characters)
37	SMS failed: <i>Body</i> too long or empty (maximum 160 characters)
38	SMS failed: Wrong message type
39	SMS failed: Wrong message encoding

3.3 Query Message Delivery Status

Name: *SMSSStatusEx*

Description: This method is used to query the delivery status of one or more text messages sent via the *SendSMSEx* function.

Parameters: *SMSSStatusEx(Client_ID,Client_Pass,Client_Ref)*

Client_ID	String
Client_Pass	String
Client_Ref	String

Returns: The *Return Code* is a comma separated string of “*destination number:integer code*” for each number to which the message has been sent.

For example: +44123456789:45,+44123456789:45,+44123456789:45

Each *Return Code* is returned as an *Integer*.

2	Authentication failed – account not found
22	Authentication failed – account is currently suspended
4	Delivery status request failed: one or more parameters are too long or empty (maximum 50 characters)
40	Specified <i>Client_Ref</i> does not exist
41	Message being processed by TextAnywhere system
43	Delivery failure - message rejected by the sending network
45	Delivery success – message delivered to handset
46	Delivery failure – invalid number, likely to have been churned by user

47	Message in transit and being retried, handset likely to be switched off or out of service
48	Delivered to the network with no reports, message likely to have expired as handset not switched on or out of service
49	Message queued on TextAnywhere system
50	Message not sent as number could not be formatted
51	Message not sent as number contained in Opt-out List
52	Delivery failure – message expired as handset not switched on or out of service during delivery period
53	Delivery failure – message confirmed as not delivered, though reason unknown
54	Delivery status unknown – no delivery confirmation provided by delivering network
55	Delivery failure – message rejected by the delivering network
56	Delivery failure – the handset has insufficient credit to receive this message
57	Delivery failure – temporary problem locating handset
58	Delivery failure – problem with handset's operator
59	Delivery failure – problem with handset receiving the message, likely to be temporary
60	Delivery failure – message rejected by the network as it is considered to be spam
61	Delivery failure – message rejected by the network as content considered inappropriate
62	Delivery failure – billing issue with handset
63	Delivery failure – parental content bar
64	Delivery failure – age verification required

3.4 Get Text Message Replies

Name: *GetReply*

Description: This method returns the SMS text message replies to messages originally sent via the *SendSMSEx* function.

Parameters: *GetReply(Client_ID,Client_Pass)*

Client_ID	String
Client_Pass	String

Returns: The function *GetReply* returns a comma separated variable string, followed by a Carriage Return and Linefeed combination, for each message reply, as follows:

"ID","Originator","Destination","Client_Ref","Body","OriginalBody","Date","Time" & CRLF

Where:

<i>ID</i>	Unique identifier that is used within the <i>DeleteReply</i> function to delete messages that have already been retrieved. NB - You will be able to retrieve your messages at any time unless either you have deleted them or if three months has elapsed, in which case the system will have deleted the replies
<i>Originator</i>	the <i>Originator</i> field
<i>Destination</i>	the number to which the reply message was sent

<i>Client_Ref</i>	the <i>Client_Ref</i> given
<i>Body</i>	the content of the reply message
<i>OriginalBody</i>	the content of the original message sent
<i>Date</i>	the date that the message was sent
<i>Time</i>	the time that the message was sent

3.5 Get Replies to a particular Text Message

Name: *GetReplyByClientRef*

Description: This method returns the SMS text message replies to a particular text message (uniquely identified by the *Client_Ref* parameter), originally sent via the *SendSMSEx* function.

Parameters: *GetReplyByClientRef(Client_ID,Client_Pass,Client_Ref)*

<i>Client_ID</i>	String
<i>Client_Pass</i>	String
<i>Client_Ref</i>	String

Returns: The function *GetReplyByClientRef* returns a comma separated variable string, followed by a Carriage Return and Linefeed combination, for each message reply, as follows:

*"ID","Originator","Destination","Client_Ref","Body","OriginalBody",
"Date","Time" & CRLF*

Where:

<i>ID</i>	Unique identifier that is used within the <i>DeleteReply</i> function to delete messages that have already been retrieved. NB - You will be able to retrieve your messages at any time unless either you have deleted them or if three months has elapsed, in which case the system will have deleted the replies
<i>Originator</i>	the <i>Originator</i> field
<i>Destination</i>	the number to which the reply message was sent
<i>Client_Ref</i>	the <i>Client_Ref</i> given
<i>Body</i>	the content of the reply message
<i>OriginalBody</i>	the content of the original message sent
<i>Date</i>	the date that the message was sent
<i>Time</i>	the time that the message was sent

3.6 Delete Text Message Replies

Name: *DeleteReply*

Description: This method permanently deletes the given text message replies originally sent with the *SendSMSEx* function, from the TextAnywhere system.

Parameters: *DeleteReply(Client_ID,Client_Pass,"ID,ID,ID...")*

Client_ID	String
Client_Pass	String
IDs	String

For example: *DeleteReply("SW021567","password","245,456,457")*

Returns: Each *Return Code* is returned as an *Integer*.

0	Message deletion failed
1	Successful message(s) deletion

3.7 Get Inbound Text Messages

Name: *GetTextInbound*

Description: This method is used to retrieve all inbound SMS messages sent to a TextInbound number, set-up by the client. If the parameter *Inbound_Number* is an empty string then the method will return all inbound messages on all TextInbound phone numbers associated with the client. Alternatively, by entering a phone number in the *Inbound_Number* field, the messages received on that specific phone number will be retrieved.

Please be aware that once you have requested and received your inbound text messages, those text messages will be deleted from the TextAnywhere system and cannot be re-requested.

Parameters: *GetTextInbound(Client_ID,Client_Pass,Inbound_Number)*

Client_ID	String
Client_Pass	String
Inbound_Number	String

Returns: The *Return Code* is a *CSV String*.

On successful completion of this method, i.e. the SMS Gateway is able to respond, a CSV string is returned as follows:

"Sender","Destination","Date","Time","Body" (followed by Carriage Return and Line Feed)

Where:

<i>Sender</i>	the mobile phone number of the sender
<i>Destination</i>	the TextInbound number that the message was sent to
<i>Date</i>	the date that the message was sent
<i>Time</i>	the time that the message was sent
<i>Body</i>	the content of the message sent

There are three possible *Return* situations:

1. There are no inbound text messages to receive, in which case an empty string is returned;
2. A single inbound text message is returned, in the format above;

- Multiple inbound text messages are returned, in the above format, with each message delimited by a CRLF (Carriage Return and Line Feed combination).

3.8 Format Number

Name: *FormatNumber*

Description: This function is used to return a correctly formatted mobile phone number from the number given.

Parameters: *FormatNumber(Number)*

Number String

Returns: Each *Return* is a *String*.

If the *Number* can be formatted a string is returned with the formatted number.

If the *Number* can not be formatted a value of "-1" is returned.

3.9 Check Number

Name: *CheckNumber*

Description: This method is used to check the validity of a mobile phone number. It returns the mobile phone's country, correctly formatted number and the cost in *Message Credits* of sending a text message to this mobile phone.

Parameters: *CheckNumber(Number)*

Number String

A "+" must be entered before the mobile phone number.

Returns: Each *Return* is a *String*.

If correctly formatted a string is returned with the following fields:

Country
Formatted mobile phone number
Message sending cost, in credits

If incorrectly formatted a string is returned with the following field:
"*This is not a correctly formatted number*"

3.10 Service Test

Name: *ServiceTest*

Description: This method is used to query the current status of the TextAnywhere Gateway. It returns the Gateway's status and your organisation's name.

Parameters: *ServiceTest(Client_ID,Client_Pass)*

Client_ID String
Client_Pass String

Returns: The *Return Code* is a *String*.

On successful completion of this method, i.e. the Gateway is able to respond, a string is returned as follows:

"*Service Running* → <Organisation Name>"

3.11 Check Credits

Name: *CheckCredits*

Description: This method is used to query the number of *Message Credits* available on a client's pre-pay TextAnywhere account. The method returns the number of *Message Credits* as a real number.

Parameters: *CheckCredits(Client_ID, Client_Pass)*

Client_ID	String
Client_Pass	String

Returns: The *Return Code* is a *String*.

On successful completion of this method a string is returned with the number of *Message Credits* on the account, as follows:

"2249.7"

If either of the Client_ID or Client_Pass are incorrect, then the method will return:

"Couldn't find account"

If the account is found, and the account is a Credit Account (where messages are not pre-purchased), then the method will return:

"-1"

SECTION FOUR –HTTP(S) RECEIVER FOR SMS INTEGRATION

4.1 Introduction

The following section describes the available *TextAnywhere SMS HTTP(S) Receiver Methods*.

Each method includes the method name, a brief description of its function, the available parameters and the return codes. The parameters associated with each of the methods are described in Section Two.

The HTTP(S) Receiver is accessed at:

[http\(s\)://ws.textanywhere.net/HTTPRX/<method_name>.aspx?<parameter>=<value>&<etc>](http(s)://ws.textanywhere.net/HTTPRX/<method_name>.aspx?<parameter>=<value>&<etc>)

To engage the HTTP receiver you will need to replace the terms between the < and > as follows:

<method_name> the name of the desired method
<parameter> the name of the relevant parameter of the method
<value> the parameter value.

Multiple parameter/value pairs should be separated by '&'. The methods will work with GET or POST. Values should be URL encoded. We recommend the use of the POST command as the GET command can sometimes display confidential account parameters on the URL line.

For example, to send the message “hello world” to the mobile phone number (07912) 345 678, call the method *SendSMSEx* with the following parameters and values:

[http\(s\)://ws.textanywhere.net/HTTPRX/SendSMSEx.aspx?Client_ID=EX021&Client_Pass=pass&Client_Ref=001&Billing_Ref=myRef&Connection=1&Originator=me&OType=1&DestinationEx=%2b447912345678&Body=hello%20world&SMS_Type=0&Reply_Type=0](http(s)://ws.textanywhere.net/HTTPRX/SendSMSEx.aspx?Client_ID=EX021&Client_Pass=pass&Client_Ref=001&Billing_Ref=myRef&Connection=1&Originator=me&OType=1&DestinationEx=%2b447912345678&Body=hello%20world&SMS_Type=0&Reply_Type=0)

As can be seen in the example, the leading '+' in the *DestinationEx* parameter has become %2b because of the URL encoding.

You may notice that *Reply_Data* is missing. This is because the method requires that for a *Reply_Type* of 0, you must provide an empty string and for the HTTP Receiver you may completely ignore the parameter.

Remarks: ASP.NET developers should configure their *Web.Config* to have a requestEncoding of “iso8859-1”.

4.2 Send Message to Recipients and Receive Replies

Name: *SendSMSEx*

Description: This function is used to send a single text message to multiple mobile phone numbers within one function. This function is also used to support the reply capability of the HTTP Receiver.

When sending a single message to multiple recipients, please use the following process:

1. *For up to 150 recipients:* set the *Connection* parameter to 2 and make a single call of the *SendSMSEx* method

2. For more than 150 recipients: set the *Connection* parameter to 4, and make multiple calls of the *SendSMSEx* method, waiting for the appropriate method Return Code, before calling the method again

SendSMSEx is also used to support the reply capability of the HTTP Receiver.

When sending a text message to one or more recipients to which a reply can be made, you stipulate the method by which you wish to receive that reply. This is achieved when sending the message.

You can receive replies through the HTTP Receiver, by email to a given email address or on a given web page URL.

By using the *Reply_Type* and *Reply_Data* parameters you stipulate how the message reply is delivered back to your application.

Parameters: *SendSMSEx(Client_ID,Client_Pass,Client_Ref,Billing_Ref,Connection,Originator,OType,DestinationEx,Body,SMS_Type,Reply_Type,Reply_Data)*

Client_ID	String
Client_Pass	String
Client_Ref	String
Billing_Ref	String
Connection	Integer
Originator	String
OType	Integer
DestinationEx	String
Body	String
SMS_Type	Integer
Reply_Type	Integer
Reply_Data	String

Returns: The *Return Code* is a comma separated string of “*destination number:integer code*” for each number to which the message is sent.

For example: +44123456789:1,+44123456789:1,+44123456789:1

The *integer code* returned refers to the following:

- 1 SMS Sent
- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently suspended
- 3 SMS failed
- 31 SMS failed: Insufficient message credits on your account
- 311 SMS failed: This *Connection* does not exist
- 32 SMS failed: *Originator* format not recognised
- 321 SMS failed: *OType* invalid
- 33 SMS failed: *Destination(s)* format not recognised
- 34 SMS failed: *Reply_Type* invalid or *Reply_Data* empty
- 35 SMS failed: *Client_Ref* too long or empty (maximum 50 characters)
- 36 SMS failed: *Billing_Ref* too long or empty (maximum 50 characters)
- 37 SMS failed: *Body* too long or empty (maximum 160 characters)
- 38 SMS failed: Wrong message type
- 39 SMS failed: Wrong message encoding

4.3 Query Message Status for Multiple Messages

Name: *SMSStatusEx*

Description: This method is used to query the delivery status of one or more text messages sent via the *SendSMSEx* function.

Parameters: *SMSStatusEx(Client_ID, Client_Pass, Client_Ref)*

Client_ID	String
Client_Pass	String
Client_Ref	String

Returns: The *Return Code* is a comma separated string of “*destination number:integer code*” for each number to which the message has been sent.

For example: +44123456789:45,+44123456789:45,+44123456789:45

Each *Return Code* is returned as an *Integer*.

2	Authentication failed – account not found
22	Authentication failed – account is currently suspended
4	Delivery status request failed: one or more parameters are too long or empty (maximum 50 characters)
40	Specified <i>Client_Ref</i> does not exist
41	Message being processed by TextAnywhere system
43	Delivery failure - message rejected by the sending network
45	Delivery success – message delivered to handset
46	Delivery failure – invalid number, likely to have been churned by user
47	Message in transit and being retried, handset likely to be switched off or out of service
48	Delivered to the network with no reports, message likely to have expired as handset not switched on or out of service
49	Message queued on TextAnywhere system
50	Message not sent as number could not be formatted
51	Message not sent as number contained in Opt-out List
52	Delivery failure – message expired as handset not switched on or out of service during delivery period
53	Delivery failure – message confirmed as not delivered, though reason unknown
54	Delivery status unknown – no delivery confirmation provided by delivering network
55	Delivery failure – message rejected by the delivering network
56	Delivery failure – the handset has insufficient credit to receive this message
57	Delivery failure – temporary problem locating handset
58	Delivery failure – problem with handset’s operator
59	Delivery failure – problem with handset receiving the message, likely to be temporary
60	Delivery failure – message rejected by the network as it is considered to be spam
61	Delivery failure – message rejected by the network as content considered inappropriate
62	Delivery failure – billing issue with handset
63	Delivery failure – parental content bar
64	Delivery failure – age verification required

4.4 Get Text Message Replies

Name: *GetReply*

Description: This method returns the SMS text message replies to messages originally sent via the *SendSMSEx* function.

Parameters: *GetReply(Client_ID,Client_Pass)*

Client_ID String
Client_Pass String

Returns: The function *GetReply* returns a comma separated variable string, followed by a Carriage Return and Linefeed combination, for each message reply, as follows:

*"ID","Originator","Destination","Client_Ref","Body","OriginalBody",
"Date","Time" & CRLF*

Where:

ID Unique identifier that is used within the *DeleteReply* function to delete messages that have already been retrieved. NB - You will be able to retrieve your messages at any time unless either you have deleted them or if three months has elapsed, in which case the system will have deleted the replies

Originator the *Originator* field

Destination the number to which the reply message was sent

Client_Ref the *Client_Ref* given

Body the content of the reply message

OriginalBody the content of the original message sent

Date the date that the message was sent

Time the time that the message was sent

4.5 Get Replies to a particular Text Message

Name: *GetReplyByClientRef*

Description: This method returns the SMS text message replies to a particular text message (uniquely identified by the *Client_Ref* parameter), originally sent via the *SendSMSEx* function.

Parameters: *GetReplyByClientRef(Client_ID,Client_Pass,Client_Ref)*

Client_ID String
Client_Pass String
Client_Ref String

Returns: The function *GetReplyByClientRef* returns a comma separated variable string, followed by a Carriage Return and Linefeed combination, for each message reply, as follows:

*"ID","Originator","Destination","Client_Ref","Body","OriginalBody",
"Date","Time" & CRLF*

Where:

<i>ID</i>	Unique identifier that is used within the <i>DeleteReply</i> function to delete messages that have already been retrieved. NB - You will be able to retrieve your messages at any time unless either you have deleted them or if two months has elapsed, at which point they are deleted
<i>Originator</i>	the <i>Originator</i> field
<i>Destination</i>	the number to which the reply message was sent
<i>Client_Ref</i>	the <i>Client_Ref</i> given
<i>Body</i>	the content of the reply message
<i>OriginalBody</i>	the content of the original message sent
<i>Date</i>	the date that the message was sent
<i>Time</i>	the time that the message was sent

4.6 Delete Text Message Replies

Name: *DeleteReply*

Description: This method permanently deletes the given text message replies originally sent with the *SendSMSEx* function, from the TextAnywhere system.

Parameters: *DeleteReply(Client_ID,Client_Pass,"ID,ID,ID...")*

Client_ID	String
Client_Pass	String
ID	String

For example: *DeleteReply("SW021567","password","245,456,457")*

Returns: Each *Return Code* is returned as an *Integer*.

0	Message deletion failed
1	Successful message(s) deletion

4.7 Get Inbound Text Messages

Name: *GetTextInbound*

Description: This method is used to retrieve all inbound SMS messages sent to a TextInbound number, set-up by the client. If the parameter *Inbound_Number* is an empty string then the method will return all inbound messages on all TextInbound phone numbers associated with the client. Alternatively, by entering a phone number in the *Inbound_Number* field, the messages received on that specific phone number will be retrieved.

Please be aware that once you have requested and received your inbound text messages, those text messages will be deleted from the TextAnywhere system and cannot be re-requested.

Parameters: *GetTextInbound(Client_ID,Client_Pass,Inbound_Number)*

Client_ID	String
Client_Pass	String
Inbound_Number	String

Returns: The *Return Code* is a *CSV String*.

On successful completion of this method, i.e. the SMS Gateway is able to respond, a CSV string is returned as follows:

"Sender","Destination","Date","Time","Body" (followed by Carriage Return and Line Feed)

Where:

Sender the mobile phone number of the sender

Destination the TextInbound number that the message was sent to

Date the date that the message was sent

Time the time that the message was sent

Body the content of the message sent

There are three possible *Return* situations:

1. There are no inbound text messages to receive, in which case an empty string is returned;
2. A single inbound text message is returned, in the format above;
3. Multiple inbound text messages are returned, in the above format, with each message delimited by a CRLF (Carriage Return and Line Feed combination).

4.8 Format Number

Name: *FormatNumber*

Description: This function is used to return a correctly formatted mobile phone number from the number given.

Parameters: *FormatNumber(Number)*

Number	String
--------	--------

Returns: Each *Return* is a *String*.

If the *Number* can be formatted a string is returned with the formatted number.

If the *Number* can not be formatted a value of "-1" is returned.

4.9 Check Number

Name: *CheckNumber*

Description: This method is used to check the validity of a mobile phone number. It returns the mobile phone's country, correctly formatted number and the cost in *Message Credits* of sending a text message to this mobile phone.

Parameters: *CheckNumber(Number)*

Number String

A “+” must be entered before the mobile phone number.

Returns: Each *Return* is a *String*.

If correctly formatted a string is returned with the following fields:

Country
Formatted mobile phone number
Message sending cost, in credits

If incorrectly formatted a string is returned with the following field:
“*This is not a correctly formatted number*”

4.10 Service Test

Name: *ServiceTest*

Description: This method is used to query the current status of the TextAnywhere Gateway. It returns the Gateway’s status and your organisation’s name.

Parameters: *ServiceTest(Client_ID,Client_Pass)*

Client_ID String
Client_Pass String

Returns: The *Return Code* is a *String*.

On successful completion of this method, i.e. the Gateway is able to respond, a string is returned as follows:

“*Service Running* → <Organisation Name>”

4.11 Check Credits

Name: *CheckCredits*

Description: This method is used to query the number of *Message Credits* available on a client’s pre-pay TextAnywhere account. The method returns the number of *Message Credits* as a real number.

Parameters: *CheckCredits(Client_ID,Client_Pass)*

Client_ID String
Client_Pass String

Returns: The *Return Code* is a *String*.

On successful completion of this method a string is returned with the number of *Message Credits* on the account, as follows:

“2249.7”

If either of the Client_ID or Client_Pass are incorrect, then the method will return:

“*Couldn’t find account*”

If the account is found, and the account is a Credit Account (where messages are not pre-purchased), then the method will return:

“-1”

SECTION FIVE – SMTP SERVICE

5.1 Introduction

The following section provides details on how to send and receive a text message, as an email, from within an application using the *SMTP Service*.

The *SMTP Service* represents the simplest means of sending text messages from within an application. Messages are sent as emails to the *TextAnywhere SMS Gateway*, from the application.

This service does not support the enquiring of the delivery status of a message and does not have the same depth of functionality as the *Web Service* or *HTTP Receiver*.

You can also receive text messages sent to your (optional) *TextInbound* number in an email format, documented in Section 5.4.

5.2 Sending a text message

There are four TextAnywhere domains that can be used to send your email to. Each domain offers different email-to-SMS functionality.

You can limit the content of the email to a single SMS, as well as sending your chosen 11 characters (*Originator*) with the text message, rather than a phone number that is used for reply purposes.

When an email message is sent to the *TextAnywhere SMS Gateway*, the gateway verifies the email by checking the **From** address of the email with the addresses of the various users associated with the organisation's account.

If there is a match, the email is processed and the requisite text messages sent. Otherwise, a failure email is returned to the **From** address.

The text message appears on the recipient's phone with the following fields populated:

- | | |
|--------------------|--|
| Body: | The content of the email message. |
| Time stamp: | The date and time that the message was sent. |
| Originator: | A TextAnywhere reply-path phone number to enable the recipient to reply to the received text message or your 11 character Originator, depending which of the four email domains you sent the message to. |

Please note the following phone number formatting requirements:

- For messages to be delivered to UK mobile phones, the *phone-number* field should begin with a 0 and be followed by the next 10 digits. For example: 07836123456@mail2txt.net.
- For messages to be delivered to internationally-registered phones or UK landline phones, the *phone-number* field should begin with a +, and be followed by the Country Code and the remainder of the number. For example: +33123456789@mail2txt.net.
- Multiple numbers can be entered in the *To* field, delimited by semicolons (;).

The four email domains, their functionality and how to use each domain, is documented below:

5.2.1 mail2txt.net

We take the subject and body of the email and send it out as one or more text messages. Adding \$\$ after your message ensures that any email signature or disclaimer isn't sent out with your message.

We send the SMS(s) out with one of our reply-path numbers to enable any replies to be passed back to your inbox.

To send a text message, the email must be formatted as follows:

- To:** phone-number@mail2txt.net
- Cc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Bcc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Subject:** Any subject of the email will become the first line content of the text message.
- Body:** The content of the email message body will be the content of the text message, displayed after any subject body.

The body will be split in to 160 character parts (the maximum length of a text message) and will be sent as multiple messages if over 160 characters.

To ensure that any email signature or disclaimer is not sent out with your message body, enter \$\$ at the end of the message body. Upon receiving your email, the SMS Gateway will only send out the message body before the \$\$, discarding all content after the \$\$.

5.2.2 mail2txt160.net

We take the subject and as much of the body of the email as we can, and send it out as one text message only (up to a maximum of 160 characters). No need to add \$\$ after your message (though it is still supported).

We send the message out with one of our reply-path numbers to enable any replies to be passed back to your inbox.

To send a text message, the email must be formatted as follows:

- To:** phone-number@mail2txt160.net
- Cc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Bcc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Subject:** Any subject of the email will become the first line content of the text message.

Body: The content of the email message body will be the content of the text message, displayed after any subject body.

Only one SMS message will be sent, with as much of the *Subject* and *Body* of the email as can be accommodated in a single 160-character SMS message.

Entering \$\$ at the end of your email body will stop any content after the \$\$ from being sent out.

5.2.3 mail2txtid.net

This service replaces the reply-path number that we would normally send with the message, replacing it with the first 11 characters of the email's *Subject* field. So when the message arrives on the recipient's phone they see who it's from, rather than one of our reply-path numbers.

We take the body of the email and send it out as one or more text messages. Adding \$\$ after your message ensures that any email signature or disclaimer isn't sent out with your message.

To: phone-number@mail2txtid.net

Cc: This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.

Bcc: This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.

Subject: The first 11 characters of the *Subject* field will be sent as the *Originator* for the SMS. None of the *Subject* content will be sent in the body of the SMS.

If the first four characters are *Re:* , then these will be ignored and the next 11 characters will be taken.

If the *Subject* is empty, then a reply-path number will be sent instead of an alphanumeric originator.

Body: The content of the email message body will be the content of the text message.

The body will be split in to 160 character parts (the maximum length of a text message) and will be sent as multiple messages if over 160 characters.

To ensure that any email signature or disclaimer is not sent out with your message body, enter \$\$ at the end of the message body. Upon receiving your email, the SMS Gateway will only send out the message body before the \$\$, discarding all content after the \$\$.

5.2.4 mail2txt160id.net

As a combination of the above two services, we take as much of the body of the email as we can, and send it out as one text message only (up to a maximum of 160 characters). No need to add \$\$ after your message (though it is still supported).

This service also replaces the reply-path number that we would normally send with the message, replacing it with the first 11 characters of the email's *Subject* field. So when the message arrives on the recipient's phone they see who it's from, rather than one of our reply-path numbers.

- To:** phone-number@mail2txt160id.net
- Cc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Bcc:** This field is unused. If populated, the *TextAnywhere SMS Gateway* will ignore the contents.
- Subject:** The first 11 characters of the *Subject* field will be sent as the *Originator* for the SMS. None of the *Subject* content will be sent in the body of the SMS.
- If the first four characters are *Re:* , then these will be ignored and the next 11 characters will be taken.
- If the *Subject* is empty, then a reply-path number will be sent instead of an alphanumeric originator.
- Body:** The content of the email message body will be the content of the text message, displayed after any subject body.
- Only one SMS message will be sent, with as much of the *Subject* and *Body* of the email as can be accommodated in a single 160-character SMS message.
- Entering \$\$ at the end of your email body will stop any content after the \$\$ from being sent out.

5.3 Receiving a reply

The recipient of a sent message can reply back to the text message, with the message converted to an email by the *TextAnywhere SMS Gateway* and then sent back to the original sending email address, with the following format:

- From:** The message is sent from <phone-number>@mail2txt.net. The phone number is internationally formatted. If the reply is received in to a working email inbox, the email message can be replied to with the message sent to TextAnywhere as an email and then relayed to the recipient as an SMS.
- To:** The message is sent to the address of the original sender.
- Subject:** *New SMS reply message*
- Body:** The body contains the following fields and information:
1. *Sender:* the full international phone number of the sending mobile phone
 2. *Date:* the date that the message was sent
 3. *Client Ref:* an internally used, TextAnywhere message tracking code
 4. *Time:* the time that the message was sent
 5. *Body:* the content of the message
 6. *Original message:* the content of the original message

5.4 Receiving TextInbound messages as email

If you have a *TextInbound* number (the ability to receive unsolicited SMS messages sent to your own dedicated 11 digit number), you can elect to receive the SMS messages sent to that number by email. Alternatively, you can receive inbound messages by *HTTP* or the *Web Service*.

You setup your *TextInbound* number and choose your method for receiving the inbound messages from the *TextInbound* section of your account. This is accessed by selecting *TextInbound* from the *Select application* drop-down menu.

If you choose to receive your inbound SMS by email, you will be prompted to enter the email address to which the inbound messages will be relayed.

The inbound SMS messages are converted to an email by the *TextAnywhere SMS Gateway* and then sent to your given email address, with the following format:

- From:** The message is sent from textinbound@textanywhere.net
- To:** The email address you nominate
- Subject:** *New inbound SMS message*
- Body:** The email body contains the following fields and information:
1. *Sender:* the full international phone number of the sending mobile phone
 2. *Destination:* the full international phone number of your *TextInbound* number
 3. *Date:* the date that the message was sent, in the format: *dd/MM/yyyy*
 4. *Time:* the time that the message was sent, in the format: *HH:mm:ss*
 5. *Body:* the content of the message

SECTION SIX – CODE SAMPLES AND DEVELOPER CENTRE

6.1 Introduction

The following section provides access details to our *Developer Centre* where you can find code samples for the Web Service and HTTP(S) Receiver, for many of our previously-documented methods.

6.2 Developer Centre

Our Developer Centre will provide you with the resources you need to help you easily and swiftly integrate text messaging in to your application. We have detailed information on the APIs we provide, as well as code samples, comprehensive downloadable documentation, and access to our developer forums.

You can access the Developer Centre by clicking either [here](#) or on the link below:

<http://developer.textanywhere.net/v2/home/Default.aspx>

SECTION SEVEN – SHORT CODE AND REVERSE BILLING SERVICES

7.1 Introduction

The following section provides details on how to receive text messages sent in to your TextPremium *Short Code* and *Keyword* service, and, optionally, send back premium text messages that are billed at a higher rate to the message recipient. These are known as *Short Code* and *Reverse Billing* services.

To run your own *Short Code* service, you will first need to have set-up your service with TextAnywhere, see Section 7.2 below.

With a *Short Code* service, as well as receiving inbound messages sent in to your service (billed to the sender at their normal, prevailing SMS rate), you can also, optionally, send back premium text messages to the handset directly, and automatically, from within your application.

If your *Short Code* service does involve sending back premium messages, then there are two different scenarios in which you will be sending premium messages:

1. A user is entering a competition or voting on a topic, and expects to receive a **single** SMS (or possibly multiple, if stipulated in the terms of the service), by response.
2. A user is subscribing to a service where they expect to receive **multiple** reverse-billed messages. For example the user may be subscribing to a news or astrology service where they expect to receive one or two premium, reverse-billed messages each day.

TextPremium enables you to manage both types of reverse-billed application, where either you manage the interaction with your users, or you use our subscription service to automatically compile and maintain your subscription list.

When using our standard service you receive an inbound message to your *Short Code* and *Keyword*, and you send back one or more premium messages to the recipient's handset, through your application.

When using our subscription list facility, there are two methods that enable you to send your content to either all or some of the subscription list.

With our subscription list facility, the subscribers automatically receive a “*Welcome*”, non-premium text on subscribing, and are added to the subscription list for your service. A subscriber wishing to unsubscribe will be removed from the subscription list automatically when they text “*STOP*”, followed by your *Keyword* to your *Short Code*. The un-subscriber will receive automatic confirmation of their removal from your service via a non-premium text message.

Methods to send and receive premium content messages can be called by either the *HTTP(S) Receiver* or *Web Service*.

7.2 Setting-up your Short Code service

To set up your service, you will first need to register your service requirements with us from your online account, and then await confirmation and activation of your *Keyword* on an appropriate *Short Code*.

To register your service, log in to your account, click on the *TextPremium* tab, and you will arrive on the *TextPremium* home page. You will then need to register your service by completing the *New Short Code* registration form.

It is a requirement of *PhonepayPlus* (the governing body of premium rate text and voice services) that TextAnywhere retains details of each *Short Code* service a client is running.

Once your service has been registered, and we have confirmed back to you that your unique *Short Code* and *Keyword* combination has been activated, you are then in a position to integrate a *Short Code* service within your application.

7.3 Receiving inbound messages

You can elect to receive the inbound messages to your *Short Code* and *Keyword* service in one of two ways:

1. *Proactively*: have TextAnywhere automatically forward any inbound message to a nominated web address, described below. This automated approach is the most common and efficient means of receiving inbound messages
2. *Reactively*: retrieve inbound messages by polling TextAnywhere for new messages using a method invoked from either the HTTP(S) Receiver or the Web Service. The method to achieve this is described below in 7.6.1

You can configure the appropriate option from the TextPremium area of your online account,

If you choose the proactive approach, then each inbound message will be delivered to you by an *HTTP(S) GET* to your given URL, with the following variables:

<i>Originator</i> :	the full international phone number of the sending phone
<i>Destination</i> :	the <i>Short Code</i> number and <i>Keyword</i> that the message was sent to, as <i>Shortcode_Keyword</i>
<i>Body</i> :	the content of the message
<i>RBID</i> :	the <i>Reverse Billing Identifier</i> , a unique code verifying the ability to then send back a premium content message using <i>SendMTResponse</i>
<i>Date</i> :	the date that the message was received, in the format: <i>dd/MM/yyyy</i>
<i>Time</i> :	the time that the message was received, in the format: <i>HH:mm:ss</i>

7.4 Sending Premium Messages (optional)

The process for sending a premium, reverse-billed message, starts with first receiving a message sent to your *Short Code* and *Keyword*. The message sent to your service is charged to the sender at their normal, prevailing rate for sending text messages. There is no premium charge for any messages sent to your *Short Code* service.

It is the initial inbound message that provides you with the authority and associated code (through the *Reverse Billing Identifier*) to be able to send back one or more premium messages.

Having received an inbound message to your premium content service, you will then wish to send one or more premium content messages back.

Depending on your choice of service, there are different methods available to send premium content messages to your users. These methods are documented in Section 7.6.

If your service is a standard service, then you will be receiving inbound messages and you will be then deciding on the premium messages that are sent back.

If your service is a subscription service, then TextAnywhere will manage subscribers to your service and you will send premium messages back to all or some of the members of your *Subscriber Group*, for your service.

7.5 HTTP(S) Receiver and Web Service access

The methods associated with the TextAnywhere *Short Code* and *Reverse Billing* services, documented in Section 7.6 below, can be accessed from both the *HTTP(S) Receiver* and *Web Service*.

7.5.1 Accessing the HTTP(S) Receiver

The HTTP(S) Receiver is accessed at:

[http\(s\)://ws.textanywhere.net/HTTPRX/<method_name>.aspx?<parameter>=<value>&<etc>](http(s)://ws.textanywhere.net/HTTPRX/<method_name>.aspx?<parameter>=<value>&<etc>)

To engage the HTTP receiver you will need to replace the terms between the < and > as follows:

<method_name> the name of the desired method
<parameter> the name of the relevant parameter of the method
<value> the parameter value.

Multiple parameter/value pairs should be separated by '&'. The methods will work with GET or POST. Values should be URL encoded. We recommend the use of the POST command as the GET command can sometimes display confidential account parameters on the URL line.

7.5.2 Accessing the Web Service

Web Service name: TA_SMS_RB

Access Point: [http\(s\)://ws.textanywhere.net/ws_textpremium/TA_WS_RB.asmx](http(s)://ws.textanywhere.net/ws_textpremium/TA_WS_RB.asmx)

WSDL The XML-based *Web Services Description Language* (WSDL) for the TextAnywhere *Short Code* and *Reverse Billing* Web Service is provided at the Access Point. The Web Service methods, properties, input and output parameters, and how to connect to the Web Service are all documented.

7.6 Available Methods

Each method includes the method name, a brief description of its function, the available parameters and the return codes. Each method can be called from the *HTTP(S) Receiver* or *Web Service*.

7.6.1 Retrieve Inbound SMS

Name: *ReadTextMT*

Description: This method is used to proactively retrieve all inbound SMS messages sent to a TextPremium *Short Code* and *Keyword*. This method does not delete the retrieved messages.

This method is not relevant when you have configured, online, for all inbound messages to be automatically sent to a nominated URL. This automated approach is the most common and efficient means of receiving inbound messages.

Parameters: *ReadTextMT(Client_ID,Client_Pass,Inbound_Number)*

Client_ID String
Client_Pass String
Inbound_Number String

Inbound_Number should be equivalent to *Shortcode_Keyword*. For example, if your *Short Code* is 87654 and your *Keyword* is HOLIDAY, then your *Inbound_Number* would be 87654_HOLIDAY.

Returns: Received messages sent to *Inbound_Number* or an error message as XML contained within a string, are returned. XML carrying messages are structured as follows:

```
<?xml version="1.0"?>
<Messages>
  <Message ID="1">
    <SRC>447890123456</SRC>
    <DST>80000</DST>
    <RBID>00000000000</RBID>
    <TDate>04/04/04</TDate>
    <TTime>10:00</TTime>
    <Body>keyword</Body>
  </Message>
  <Message ID="...">
    ...
  </Message>
</Messages>
```

Where:

Messages: will contain one or more *Message* elements.

Message: Contains source (SRC), destination (DST), reverse billing identifier required for *SendMTRResponse* (RBID), date received (TDate), time received (TTime) and the body (Body) of the message.

TDate: is in the UK format dd/mm/yy.

TTime is in 24 hour format. Each *Message* element has an ID attribute which is used to remove the message from the server using the *DeleteTextMOMT* method.

If there is an error during the method call, the returned XML will be structured as follows:

```
<?xml version="1.0"?>
<Error>
  <Description>ERROR_DESCRIPTION</Description>
</Error>
```

The *Description* element will contain information relevant to the error which occurred.

7.6.2 Delete Received SMS

Name: *DeleteTextMOMT*

Description: This method is used to delete the specified inbound messages, previously retrieved.

Parameters: *DeleteTextMOMT(Client_ID,Client_Pass,IDs)*

Client_ID	String
Client_Pass	String
IDs	String

The *IDs* are a comma separated list of IDs returned from *ReadTextMT*.

Returns: Each *Return Code* is returned as an *Integer*.

- 0 Message deletion failed
- 1 Successful message(s) deletion
- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently stopped

7.6.3 Send a Premium Reverse Billed SMS

Name: *SendMTResponse*

Description: Sends a single reverse billed, premium content message to a single mobile phone that has previously sent an inbound message to your premium content service.

Parameters: *SendMTResponse(Client_ID,Client_Pass,Client_Ref,RBID,Body)*

Client_ID	String
Client_Pass	String
Client_Ref	String
RBID	String

The *Reverse Billing Identifier* which identifies the single recipient for the message to be sent to. **Only one RBID can be included per call.**

Body	String
------	--------

The actual message to be sent to the single mobile phone, to a maximum of 160 characters

Please note that when sending a reverse billed message it is not possible to stipulate your own *Originator* (or *Sender ID* field as it is also known). The *RBID* field has embedded within it the *Originator*, which is automatically set to the *Short Code* of the premium service.

Returns: Each *Return Code* is returned as an *Integer*.

- 1 Message successfully sent
- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently stopped
- 33 *RBID* is empty or not found
- 35 *Client_Ref* too long or empty (maximum 50 characters)
- 37 *Body* is empty, not found or longer than 160 characters
- 4 Error occurred during the sending process

7.6.4 Subscription service: Send an SMS to your Subscriber Group

Name: *SendSMSMT*

Description: This method is used to send a premium content message to the complete group of subscribers for your premium content service.

Parameters: *SendSMSMT(Client_ID,Client_Pass,Client_Ref,Destination_Group,Body)*

Client_ID	String	
Client_Pass	String	
Client_Ref	String	
Destination_Group	String	The group of subscribers to send the message to: formatted as <i>Shortcode_Keyword</i>
Body	String	The actual message to be sent to the <i>Destination_Group</i> , to a maximum of 160 characters

Returns: Each *Return Code* is returned as an *Integer*.

- 1 Message successfully sent
- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently stopped
- 33 *Destination_Group* is empty or not found
- 35 *Client_Ref* too long or empty (maximum 50 characters)
- 37 *Body* is empty, not found or longer than 160 characters
- 4 Error occurred during the sending process

7.6.5 Subscription service: Send an SMS to members of your Group

Name: *SendSMSMTSelect*

Description: This method is used to send an MT premium content message to one or more specified members of your Subscriber Group for your premium content service.

Parameters: *SendSMSMTSelect(Client_ID,Client_Pass,Client_Ref, Destination_Group, Destination_Numbers,Body)*

Client_ID	String	
Client_Pass	String	
Client_Ref	String	
Destination_Group	String	The group of subscribers to send the message to: formatted as <i>Shortcode_Keyword</i>
Destination_Numbers	String	A comma separated (without spaces) list of internationally formatted numbers (with + and the country code), to which the premium content message is to be sent
Body	String	The actual message to be sent to the <i>Destination_Group</i> , to a maximum of 160 characters

Returns: Each *Return Code* is returned as an *Integer*.

- 1 Message successfully sent
- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently stopped
- 33 *Destination_Group* or *Destination_Numbers* are empty
- 35 *Client_Ref* too long or empty (maximum 50 characters)
- 37 *Body* is empty, not found or longer than 160 characters
- 4 Error occurred during the sending process

7.6.6 Subscription service: Retrieve list of Subscribers

Name: *GetMTGroupSubscribers*

Description: This method is used to retrieve the mobile phone numbers of the current subscribers to your premium content service.

Parameters: *GetMTGroupSubscribers(Client_ID,Client_Pass,Destination_Group)*

Client_ID	String	
Client_Pass	String	
Destination_Group	String	The group of subscribers whose numbers will be retrieved: formatted <i>Shortcode_Keyword</i>

Returns: The *Return Code* is a *CSV String*.

On successful completion of this method a CSV string is returned with a list of mobile phone numbers or one of the following error codes:

- 2 Authentication failed: Account not found
- 22 Authentication failed: Your account is currently stopped
- 33 *Destination_Group* is empty or not found

SECTION EIGHT – REGISTRATION AREA FOR PARTNERS’ CLIENTS

8.1 Introduction

For TextAnywhere partners who have their clients registered with us directly, there is a dedicated registration page.

The purpose of this page is to simplify the registration process, reduce the opportunity for clients to browse other TextAnywhere services and ensure tighter integration with our partners’ websites. The partner client registration page has the following features:

- The registration process is contained on a single page.
- There are no links to other TextAnywhere services, except for viewing our *Privacy Policy* and *Terms and Conditions* documents.
- The partner’s *Reseller Code* is automatically associated with the client and not displayed on the form.
- There is no choice of application-type to be made. On completing the registration form, the client will receive a simple *Welcome* email from TextAnywhere, providing the *Client_ID* and *Client_Pass* parameters to be entered in to the partner’s application. TextAnywhere will still then perform a credit-type check on the client, before activating the account and confirming the account activation via email.
- On completing the registration (by clicking on the *Register* button), the client is returned to a given partner website page

The use of this alternative registration page is entirely optional.

8.2 Use of the registration page

To access the partner client registration page from your website, you will need to use the following URL:

<https://ws.textanywhere.net/web/OpenAccount/PartnerRegister.aspx?resellercode=YOURCODE&redirecturl=YOURURL>

The two variables that you need to pass with the URL are:

- *resellercode*: this is your unique TextAnywhere *Reseller Code*
- *redirecturl*: this is your web page that a successfully registered client will be redirected to

APPENDIX ONE – SUPERCEDED METHODS

A.1 Introduction

TextAnywhere continuously improves and enhances its services, introducing new methods and functionality for the Web Service and HTTPS Receiver.

Consequently, some methods are superseded with new methods that offer new features. However, every method, no matter how old, will always be supported through our platform.

Where we have introduced new methods, the older methods are removed from the primary part of this document, and listed in this appendix for completeness.

A.2 Send Message to a Single Recipient

Name: *SendSMS*

Description: This method is used to send one SMS text message to one mobile phone number.

To send a message to multiple recipients or to enable the recipient to reply to your sent message, use the *SendSMSEx* method, documented in 3.3 below.

Parameters: *SendSMS(Client_ID,Client_Pass,Client_Ref,Billing_Ref,Connection,Originator,OType,Destination,Body,SMS_Type,SMS_Encoding)*

A.3 Query Message Status for Single Message

Name: *SMSStatus*

Description: This method is used to query the TextAnywhere Gateway and return the message's delivery status

Parameters: *SMSStatus(Client_ID,Client_Pass,Client_Ref)*

Client_ID	String
Client_Pass	String
Client_Ref	String

Returns: Each *Return Code* is returned as an *Integer*.

2	Authentication failed – account not found
22	Authentication failed – account is currently suspended
4	Delivery status request failed: one or more parameters are too long or empty (maximum 50 characters)
40	Specified <i>Client_Ref</i> does not exist
41	Message being processed by TextAnywhere system
43	Delivery failure - message rejected by the sending network
45	Delivery success – message delivered to handset
46	Delivery failure – invalid number, likely to have been churned by user
47	Message in transit and being retried, handset likely to be switched off or out of service
48	Delivered to the network with no reports, message likely to have expired as handset not switched on or out of service
49	Message queued on TextAnywhere system
50	Message not sent as number could not be formatted

- 51 Message not sent as number contained in Opt-out List
- 52 Delivery failure – message expired as handset not switched on or out of service during delivery period
- 53 Delivery failure – message confirmed as not delivered, though reason unknown
- 54 Delivery status unknown – no delivery confirmation provided by delivering network
- 55 Delivery failure – message rejected by the delivering network
- 56 Delivery failure – the handset has insufficient credit to receive this message
- 57 Delivery failure – temporary problem locating handset
- 58 Delivery failure – problem with handset's operator
- 59 Delivery failure – problem with handset receiving the message, likely to be temporary
- 60 Delivery failure – message rejected by the network as it is considered to be spam
- 61 Delivery failure – message rejected by the network as content considered inappropriate
- 62 Delivery failure – billing issue with handset
- 63 Delivery failure – parental content bar
- 64 Delivery failure – age verification required