

XML Connector v1.1.1 for txttools 6.0 and up

How-To

Site version: 6.0
Connector Version: 1.1.1
Document Version: 2.0.2
Author: Greg J Preece
techies@txttools.co.uk
Date: 07/10/2009

The information contained herein is the property of txttools Ltd, and may not be copied, used or disclosed in whole or in part, except with the prior written permission of txttools Ltd.

Table of Contents

XML Connector for Messaging.....	3
What is the XML Connector?.....	3
Requirements for Using the XML Messaging Connector.....	3
The XML Messaging Connector Test Page.....	4
The <Request> Element.....	5
The <Authentication> Element.....	5
Sending an SMS Message.....	6
Getting Status Updates.....	8
Retrieving Inbound Messages.....	10
Getting Account Details.....	12
Example Request and Response.....	13
XML Push API for Messaging.....	17
What is the XML Push API?.....	17
How to use the API.....	17
InboundMessage Reponse Structure in XML Push:.....	18

XML Connector for Messaging

What is the XML Connector?

The XML Connector allows web and application developers a fast, flexible and standards based method for sending and receiving SMS messages as well as status updates for sent messages. XML is delivered to the server using the standard HTTP 1.1 POST to a web form. All responses from the connector are also provided to the requesting client as XML.

The required structure of the XML, as well as that of the expected response from the connector, is described in the following section.

Requirements for Using the XML Messaging Connector

In order to send or receive messages through the XML messaging connector, you first build the required XML document, URI-encode it to ensure there are no problems during transmission, and then send the document to the XML connector over HTTP 1.1.

The complete XML structure for request **MUST** be contained in a <Request> tag and **MUST** contain an Authentication structure and one or more sub-elements. Please correctly escape any character fields using <![CDATA]> tags, and URI-encode your transmissions, to prevent parse errors. Please also note that **XML IS case sensitive** and tags should be used as shown.

NB: txttools accounts must be enabled for use with the XML connector. Contact support@txttools.co.uk if you need to have your account enabled.

The address for the XML messaging connector is:
<http://www.txttools.co.uk/connectors/XML/xml.jsp?XMLPost=>

An SSL 128 bit encrypted version of this connector is available at:
<https://www.txttools.co.uk/connectors/XML/xml.jsp?XMLPost=>

For more information on XML see the W3C XML site at:

<http://www.w3.org/XML/>

For more information on HTTP 1.1 see the relevant RFC

RFC2616: Hypertext Transfer Protocol -- HTTP/1.1
<http://www.ietf.org/rfc/rfc2616.txt>

Please note: When connecting to the XML messaging connector, we ask that you please set your User-Agent string, where possible, to be your txttools username. This enables us to identify which users are having problems connecting, and assist them if necessary.

The XML Messaging Connector Test Page

If you wish to test your constructed XML manually before using it as part of an automatic system, you can use the XML connector test page to do so. Open your chosen browser and go to:

<https://www.txttools.co.uk/connectors/XML/xml.jsp?test=true>

This will display a very basic page containing only a textbox and a submit button. If you cut and paste your XML into this box, and click the submit button, it will be run on the live site, and the result displayed onscreen.

(Mozilla Firefox users may need to click CTRL+U to open the returned XML.)

Please note: This test page operates on the live site. If you put an XML request into it asking that a message be sent, that message **will** be sent.

The <Request> Element

The <Request> element is the root tag of the XML document. All XML sent to the connector must be contained within a <Request> tag, or you will receive an error message in response.

The <Authentication> Element

Each XML structure sent to the server **MUST** contain the authentication structure. The authentication structure, contains two required elements <Username> and <Password>. These contain the username and password of the txttools account you want to send/receive messages through.

Please note: txttools accounts must be enabled for use with the XML connector. Contact support@txttools.co.uk if you need to have your account enabled.

Authentication Structure Elements:

Element	Description	Required
Authentication	Container element	YES
Username	The Username for authentication, it should contain a text string of the username provided.	YES
Password	The Password for authentication, it should contain the password provided.	YES

Example Authentication Structure:

```
<Authentication>  
  <Username><![CDATA[MyUsername]]></Username>  
  <Password><![CDATA[MyPassword]]></Password>  
</Authentication>
```

Sending an SMS Message

To send a message, you need to include one or more <Message> elements. This element should include the items necessary for message delivery. Required fields are <MessageText>, <Type> and at least one <Phone> tag.

<Message> Structure Elements:

Element	Description	Required
Message	The top level element for this structure	YES
MessageText	The text of the message to send	YES
Phone	The phone number including the international dialing prefix of each recipient should be contained in this tag. (e.g. +447777777777 or 00447777777777) Please Note the + character in a phone number can cause encoding problems and should be replaced with %2b if you are experiencing delivery problems. Alternatively the 0044.. form may be used. Please remember to use <![CDATA]> tags to encapsulate character data.	YES
Type	The type of message, 1 indicates bulk, and 2 indicates reverse charged. In most cases, you will want to use 1.	YES
MessageDate	A UTC timestamp (the time in seconds since Jan 1 1970) showing when the message should be delivered.	NO
UniqueID	A Unique ID to assign to this message. This is not used by the system internally, but is passed back in the connector's response and can be used in delivery reporting information	NO
TTL	The time to live in minutes for this message. If a message cannot be delivered, it will keep re-trying for this number of minutes before failing. Default is 1440, or 1 day.	NO
From	If you have the appropriate permission enabled on your account (contact txttools if you need this enabling), this field allows you to change the message source shown on the receiver's mobile phone. If this field does not contain a phone number, it may contain up to 11 alphanumeric characters (case sensitive).	NO

Example Message:

```
<Message>
  <MessageText><![CDATA[The Message Text]]></MessageText>
  <Phone><![CDATA[+441234123412]]></Phone>
  <Type>1</Type>
  <MessageDate>1234567890</MessageDate>
  <UniqueID><![CDATA[Just an ID]]></UniqueID>
  <From><![CDATA[GregJPreece]]></From>
</Message>
```

The resulting `<MessageStatus>` structure will contain the message ticket and initial status. It will also include the details of the message itself, so the receiving system can easily identify source messages and associate them with the tickets provided for retrieving status information.

Response:

Element	Description	Always Included
MessageStatus	The top level element for this structure	YES
Ticket	The ticket for this message issued by the server. This is used for status updates.	YES
Status	A numerical code representing the initial status of the message. These codes are listed under "Getting Status Updates".	YES
StatusMessage	The textual status information for this message. Please note, this text is subject to change without notice. Use the numerical codes to determine the status of your message.	YES
MessageText	The text of the message sent	YES
Phone	The phone number that the message was sent to (formatted to international standard, e.g. +447...)	YES
UniqueID	If you provided a UniqueID when sending the message, it will be passed back in this tag.	NO

Example Response:

```
<MessageStatus>
  <MessageText><![CDATA[The Message Text]]></MessageText>
  <Phone><![CDATA[+441234567890]]></Phone>
  <Type>1</Type>
  <MessageDate>1234567890</MessageDate>
  <UniqueID><![CDATA[Just an ID]]></UniqueID>
  <Ticket>123</Ticket>
  <Status>0</Status>
  <StatusMessage>
    <![CDATA[Queued For Delivery]]>
  </StatusMessage>
</MessageStatus>
```

Getting Status Updates

To fetch status updates for sent messages, you need to send a <RequestStatus> element in your XML request. The only valid sub-element is <Ticket> which should contain the message ticket assigned when a message was sent. You can include more than one <Ticket> element in the same <RequestStatus> block.

As with sending messages, the response will contain a numeric status code (See Status Response Codes table below) and a textual status message.

PLEASE NOTE: The textual status message may change without notice. Status updates should use the numeric status code not the status message to identify delivery status.

Status Request Structure:

Element	Description	Required
RequestStatus	Top level element for this structure	YES
Ticket	The message ticket that was provided when the message was sent	YES

Example Status Request:

```
<RequestStatus>
  <Ticket>123</Ticket>
</RequestStatus>
```

Response Structure:

Element	Description	Always Included
MessageStatus	Top level element for this structure	YES
Ticket	The message ticket that was provided when a message was sent	YES
Status	The numeric status code for this message.	YES
StatusMessage	The textual status information for this message. <i>N.B this text is subject to change.</i>	YES
Phone	The phone number the message was sent to.	YES

Example Response:

```
<MessageStatus>
  <Ticket>123</Ticket>
  <Status>5</Status>
  <StatusMessage>
    <![CDATA[Delivered To Handset]]>
  </StatusMessage>
  <Phone><![CDATA[+441234567890]]></Phone>
</MessageStatus>
```

Status Codes:

Status Code	Meaning
-1	Message failed at txttools due to insufficient message credits.
0	Message is in the queue at txttools, waiting to be sent (usually immediately)
1	Message has been delivered to the aggregator upstream and is on its way
2	A fatal error occurred during delivery. The message cannot be delivered.
3	Message received by phone network.
4	Delivery failed. The message has not been delivered, but the network will continue to retry until the message expires.
5	Message received by handset.
6+	A fatal error occurred during delivery. The message cannot be delivered.

Retrieving Inbound Messages

The RetrieveInbound structure is used to request any messages on the server that have been queued for delivery to your account.

Retrieve Inbound Request Structure:

Element	Description	Required
RetrieveInbound	Top level element for this structure	YES
RetrieveSince	A UTC timestamp (seconds since Jan 1 1970) that messages should be retrieved from. By default, no timestamp is set, and the latest messages are retrieved instead.	OPTIONAL
RetrieveType	Two possible values - "ALL" gets all messages, whereas "UNREAD" only fetches messages which have not already been read. ("UNREAD" was the only behaviour option in the previous version of the connector, and is the current default option as a result.)	OPTIONAL
RetrieveNumber	The number of messages to fetch from the system. (Maximum 50, default 50.)	OPTIONAL

Please Note: Unread messages are marked as read once they have been fetched through the XML connector.

Example Retrieve Inbound Request:

```
<RetrieveInbound>
  <RetrieveSince>1234567890</RetrieveSince>
  <RetrieveType><![CDATA[UNREAD]]></RetrieveType>
  <RetrieveNumber>10</RetrieveNumber>
</RetrieveInbound>
```

While the <RetrieveType> and <RetrieveNumber> are fairly self explanatory, it is worth taking a moment to fully explain the effects of <RetrieveSince>. Normally, if you specify only <RetrieveType> and/or <RetrieveNumber>, the system will search backwards in time from the present for messages – i.e. you will get the most recently received messages.

However, if you specify a timestamp in <RetrieveSince>, the system will search for messages from that timestamp **forwards**. So if you used the example above, the system would return the first 10 messages that arrived after the 1234567890 timestamp.

This is done so that users accessing the API via an automated system can make a message request, store the time at which they make the request, and then supply that timestamp as a parameter the next time they request messages – “get everything since my last check.”

InboundMessage Reponse Structure:

Element	Description	Required
InboundMessage	Top level element for this structure	YES
Ticket	The internal ticket for this message	YES
MessageText	The text of the message.	YES
Phone	The phone number of the sender	YES
Date	A UTC timestamp showing when the message was received at txttools.	YES
Destination	The phone number that the message was sent to.	YES

Example Response:

```
<InboundMessage>
  <Ticket>1278</Ticket>
  <MessageText><![CDATA[Example]]></MessageText>
  <Phone><![CDATA[+4471234567890]]></Phone>
  <Date>1234567890</Date>
  <Destination><![CDATA[88020]]></Destination>
</InboundMessage>
```

If the system detects that there are more messages in the set to be retrieved, then it will also return a <MessagesLeftInSet> element. For example, if a request were made for all the latest unread messages, and 60 unread messages were present in the inbox, the txttools system would return the first 50 messages, then return a <MessagesLeftInSet> element with the value 10.

This, again, is designed to prevent messages in the inbox from being missed due to the returned data set being too large.

MessagesLeftInSet Reponse Structure:

Element	Description	Always Included
MessagesLeftInSet	Single-element structure, holds the number of messages still to be returned from the XML connector.	NO

Example Response with MessagesLeftInSet element:

```
<InboundMessage>
  <Ticket>1278</Ticket>
  <MessageText><![CDATA[Example]]></MessageText>
  <Phone><![CDATA[+447777777776]]></Phone>
  <Date>123456123</Date>
  <Destination><![CDATA[88020]]></Destination>
</InboundMessage>
<MessagesLeftInSet>1</MessagesLeftInSet>
```

Getting Account Details

When using txttools accounts via the XML connector, you may wish to know how many message credits you have remaining on a given account, or how many messages have been sent through it. These basic account statistics are available for querying via the <AccountDetails> element.

Account Details Request Structure:

Element	Description	Required
AccountDetails	Top level element for this structure	YES
GetMessagesRemaining	Gets the number of message credits remaining on this account. Boolean field - valid values are TRUE/FALSE.	NO
GetMessagesUsed	Gets the number of message credits used on this account. Boolean field - valid values are TRUE/FALSE.	NO

Example Detail Request:

```
<AccountDetails>
  <GetMessagesRemaining>
    <![CDATA[TRUE]]>
  </GetMessagesRemaining>
  <GetMessagesUsed>
    <![CDATA[TRUE]]>
  </GetMessagesUsed>
</AccountDetails>
```

Account Details Response Structure:

Element	Description	Always Included
AccountDetail	Top level element for this structure	YES
MessagesRemaining	Holds the number of message credits remaining on this account.	NO
MessagesUsed	Holds the number of message credits used on this account.	NO

Example Detail Response:

```
<AccountDetail>
  <MessagesRemaining>50</MessagesRemaining>
</AccountDetail>
<AccountDetail>
  <MessagesUsed>15</MessagesUsed>
</AccountDetail>
```

Example Request and Response

Below is a full example request to the connector where two messages are sent, three inbound are retrieved, a status update is requested for one message that is outstanding, and account details are retrieved.

PLEASE NOTE: When copy-pasting the text below, you may need to replace the double quotes as matching quotes may not be understood by the XML.

Example Request:

```
<?xml version="1.0" ?>
<Request>
  <Authentication>
    <Username><![CDATA[MyUsername]]></Username>
    <Password><![CDATA[MyPassword]]></Password>
  </Authentication>

  <Message>
    <MessageText>
      <![CDATA[This is a test message.]]>
    </MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Type>3</Type>
    <MessageDate>1234567890</MessageDate>
    <UniqueID><![CDATA[Techies]]></UniqueID>
  </Message>
  <Message>
    <MessageText>
      <![CDATA[This is a another test message.]]>
    </MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Phone><![CDATA[+44777777778]]></Phone>
    <Type>3</Type>
  </Message>

  <RetrieveInbound>
    <RetrieveSince>1234567890</RetrieveSince>
    <RetrieveType><![CDATA[UNREAD]]></RetrieveType>
    <RetrieveNumber>3</RetrieveNumber>
    <Destination><![CDATA[88020]]></Destination>
  </RetrieveInbound>

  <RequestStatus>
    <Ticket>1234</Ticket>
  </RequestStatus>

  <AccountDetails>
    <GetMessagesRemaining>
      <![CDATA[TRUE]]>
    </GetMessagesRemaining>
    <GetMessagesUsed>
      <![CDATA[TRUE]]>
    </GetMessagesUsed>
  </AccountDetails>
</Request>
```

</Request>

Example Response:

```
<?xml version="1.0"?>
<Response>
  <MessageStatus>
    <MessageText>
      <![CDATA[This is a test message.]]>
    </MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Type>1</Type>
    <Ticket>1235</Ticket>
    <Status>0</Status>
    <StatusMessage>
      <![CDATA[Queued For Delivery]]>
    </StatusMessage>
  </MessageStatus>

  <MessageStatus>
    <MessageText>
      <![CDATA[This is a another test message.]]>
    </MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Type>1</Type>
    <Ticket>3236</Ticket>
    <Status>0</Status>
    <StatusMessage>
      <![CDATA[Queued For Delivery]]>
    </StatusMessage>
  </MessageStatus>

  <MessageStatus>
    <MessageText>
      <![CDATA[This is a another test message.]]>
    </MessageText>
    <Phone><![CDATA[+44777777778]]></Phone>
    <Type>1</Type>
    <Ticket>3237</Ticket>
    <Status>0</Status>
    <StatusMessage>
      <![CDATA[Queued For Delivery]]>
    </StatusMessage>
  </MessageStatus>

  <InboundMessage>
    <Ticket>1278</Ticket>
    <MessageText>
      <![CDATA[But can I send to Myself?]]>
    </MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Date>123456123</Date>
    <Destination><![CDATA[88020]]></Destination>
  </InboundMessage>

  <InboundMessage>
    <Ticket>3122</Ticket>
    <MessageText><![CDATA[Message 2]]></MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
```

```
        <Date>123456123</Date>
        <Destination><![CDATA[88020]]></Destination>
</InboundMessage>

<InboundMessage>
    <Ticket>3343</Ticket>
    <MessageText><![CDATA[Message 3]]></MessageText>
    <Phone><![CDATA[+44777777777]]></Phone>
    <Date>123456123</Date>
    <Destination><![CDATA[88020]]></Destination>
</InboundMessage>

<MessagesLeftInSet>5</MessagesLeftInSet>

<MessageStatus>
    <Ticket>1234</Ticket>
    <Status>5</Status>
    <StatusMessage>
        <![CDATA[Delivered To Handset]]>
    </StatusMessage>
</MessageStatus>

<AccountDetail>
    <MessagesRemaining>50</MessagesRemaining>
</AccountDetail>
<AccountDetail>
    <MessagesUsed>15</MessagesUsed>
</AccountDetail>
</Response>
```

XML Push API for Messaging

What is the XML Push API?

txttools™ offer customers the ability to provide a URI which can accept delivery of message status updates and inbound SMS messages. Any messages or status updates received by txttools are automatically “pushed” to this URI without any intervention required from the customer's system. This allows developers to build applications without the added complexity inherent in needing to continuously poll the txttools™ server for status information and messages.

How to use the API

Customers who wish to implement this interface must provide txttools with a URI for the txttools system to connect to. This URI should accept three parameters (via POST, not GET):

Parameter	Description
u	A username and password (specified by the customer), used to authenticate the connection.
p	
x	The XML being sent by the txttools system.

A few notes about the XML push API:

- 1) The XML payload structure follows the same schema as shown for the messaging API earlier in this document (but contains only inbound messages and status updates).
- 2) SSL encryption is supported on the push API, and is a recommended transmission method, to ensure secure use of the system.
- 3) Please ensure that your firewall is correctly configured to allow incoming TCP connections to this URI (typically on port 80 for HTTP or port 443 for SSL).
- 4) The structure for pushed inbound messages differs slightly from that provided when requesting messages, to help indicate which account/number the message was sent to. Please see below:

InboundMessage Reponse Structure in XML Push:

Element	Description	Required
InboundMessage	Top level element for this structure	YES
Ticket	The internal ticket for this message	YES
MessageText	The text of the message.	YES
Phone	The phone number of the sender	YES
Date	A UTC timestamp showing when the message was received at txttools.	YES
Destination	The phone number that the message was sent to.	YES
DestinationAccount	The account username that the message was pushed from	YES

Example Response:

```
<InboundMessage>
  <Ticket>1278</Ticket>
  <MessageText><![CDATA[Example]]></MessageText>
  <Phone><![CDATA[+4471234567890]]></Phone>
  <Date>1234567890</Date>
  <Destination>
    <![CDATA[+449876543210]]>
  </Destination>
  <DestinationAccount>
    <![CDATA[GregJPreece]]>
  </DestinationAccount>
</InboundMessage>
```