

A Quick Guide to the JAVA SMS REST Client

Integrating the Text Marketer API with your code

Integration Guide

Version 1.4

July 2013

www.textmarketer.co.uk

Contents

Document Change Log.....	3
Introduction.....	4
Getting Started.....	4
A Basic Command.....	4
Testing with the Sandbox.....	5
Functions Available.....	5

Document Change Log

Date	Version	Detail
03 March 2012	1.0	First publication
03 April 2012	1.1	Minor bug fix
30 December 2012	1.2	Added sub-account creation function
22 June 2012	1.3	Added schedule parameter to SendSMS
15 July 2013	1.4	Added delete SMS function

Introduction

The REST API is a powerful tool for programmatically controlling your Text Marketer account. It is the most feature-rich of all our APIs and provides the most detailed information about the success/failure of your requests.

This document describes the Java REST client library, which makes it really easy to use the REST API in your Java code by just importing the provided jar library file and making the appropriate function calls.

For more advanced use of, and errors returned by, the REST API please see the document **The Complete Text Marketer RESTful SMS Services API** also available from www.textmarketer.co.uk/developers/.

Getting Started

You will need a copy of the Java library TMRestClient.jar, available from www.textmarketer.co.uk/developers/ (go to the *Available Libraries* section of the REST API page).

You must 'import' the jar library in your Java code:

```
import uk.co.textmarketer.RestAPI.*;
```

The Jar library file needs to be in your java CLASSPATH or added as a command-line option in your java virtual machine e.g.:

```
java -classpath /usr/java/library/TMRestClient.jar
```

A Basic Command

The following is a complete example of using the REST client library. The Java REST client may throw exceptions, which means that you need to place any function calls in a try-catch block, as follows:

```
import java.util.Hashtable;
import java.util.Map;

import uk.co.textmarketer.RestAPI.*;

public class Example {
    /**
     * @param args
     */
    public static void main(String[] args) {
        RestClient tmClient = new RestClient("myusername",
            "ypassword", RestClient.ENV_SANDBOX);
        try {
            tmClient.isLoginValid();
            System.out.println("Login is OK!");

            int credits = tmClient.getCredits();
            System.out.println("Account have " + credits + "
```

```
credits.");

        Hashtable<String, String> result =
tmClient.sendSMS("Hello SMS World!", "440000000123", "Hello World", 72,
"", "");

        System.out.println("Used " + result.get("credits_used")
+ " Credits, ID: " + result.get("message_id") + ", Scheduled ID: " +
result.get("scheduled_id") + ", Status: " + result.get("status"));

        System.out.println(tmClient.getXML());

    } catch (RestClientException e) {
        e.printStackTrace();

        Hashtable<String, String> errors =
tmClient.getLastErrors();
        for (Map.Entry<String, String> error: errors.entrySet())
            System.out.println("Error code " + error.getKey()
+ ": " + error.getValue());
    }
}
```

That's it! Obviously you would need to change the code to use your own API username/password. You can find your API username and password (which may be different to your web interface username/password) via the web interface:

<https://messagebox.textmarketer.co.uk> (Settings > API Settings)

If you don't have an account, you can set one up for free at www.textmarketer.co.uk.

Testing with the Sandbox

A sandbox system is available to you for testing. The sandbox will not modify your account in any way, but will respond normally to your requests. In other words if you use the sandbox to get the number of credits on your account, it will provide the correct number. However if you use it to transfer credits between accounts, it will simulate a successful transfer (assuming you have enough credits on the account, and provide valid target account details, etc.) but it will not actually modify the number of credits on either account.

Similarly, the sandbox system will not actually send SMS messages and will not deduct credits for simulated sends.

To use the sandbox simply specify the sandbox when you create the client object:

```
RestClient tmClient = new RestClient("myusername", "ypassword",
RestClient.ENV_SANDBOX);
```

Functions Available

When you download the Java library (see *Getting Started* above) you will find automatically generated documentation included. This is the best place to see what functions are available and

what arguments they take.

A simplified list of RestClient class methods is given below.

- addGroup – create a new 'send group'
- addNumbersToGroup – add new numbers to a 'send group'
- createSubAccount – create a new sub account
- getCredits – get the number of credits available in your account
- getDeliveryReport – get the contents of a delivery report
- getDeliveryReports – get a list of the available delivery reports
- getGroup – get the numbers in a group
- getGroups – get a list of the available 'send' and 'merge' groups
- getKeyword – get the availability of a keyword for use on our 88802 short code number
- getLastErrors – get the last errors returned from the last function call (to the API)
- getLastErrorCode – get the last error code raised from the last RestClient call
- getLastErrorMessage – get the last error message raised from the last RestClient call
- getXML – get the xml string returned from the last RestClient call
- isLoginValid – check the username/password credentials used are valid
- sendSMS – send an SMS to a given recipient
- deleteSMS – delete a previously scheduled SMS
- transferCreditsToAccount – transfer credits to a specified account number
- transferCreditsToUser – transfer credits to a specified account username