PowerText LinkIt API

Documentation

Author: David Wales
Date: 15/04/2014

Table of Contents

1. Overview

The Link-It API is used specifically to “add” or “update” a contact record. It does not allow contact records to be searched or any type of contact data to be returned. Please see the SOAP API for data access functionality.

1.1 Form Setup

Form Action: `http://www.powertext.co.uk/LinkIt/processor.php` **Form Method:** `POST`

Setup your HTML form and set your form action and method as per the above url and method. This will set your form up ready to POST it’s data to the LinkIt API.

1.2 Form Variables

Please note that ALL variable names are case sensitive and must be presented within your HTML form in the correct case, as per the documentation.

1.2.1 Mandatory Form Variables

Your form must contain ALL of the following variables.

Variable: `accountID`

This is your Powertext accountID. You can request this by emailing support@powertext.co.uk

Variable: `sourceID`

Each method your customer can interact with Powertext has its own unique ID. This is called a “Contact Interaction”. You can create multiple contact interactions by passing the sourceID as an array. This is useful if your data collection form is being used by more than one unit.

You can request your current sourceID’s or set up a new one by emailing support@powertext.co.uk.

Variable: `formURL`

This should be a full URL, including http://, of your form.

Variable: `groupIDs[n]`

This array of ID’s is used to post the data to one or more groups. To post a variable as an array the field name must use this syntax groupIDs[n]. Please see example section for correct usage.

Variable: `version`

The value for this must be “6” if you are creating a standard POST HTML form. If you wish to use Ajax or another type of remote submission then this field must be set to “6Raw”.

Variable: `optinFields`

The value for this must be a comma separated list of opt-in fields which are present on your form. Only opt-in form items within this list will be processed by the LinkIt API call. Any opt-ins that are not within this list will not be processed. For existing contacts this means that their opt-in status will remain the same for non declared opt-in fields, for new contacts they will default to opted-out.

Allowed optinFields are allowSMS,allowEmail,allowSnailMail and allowCall. An example of a legal value would be 'allowEmail,allowSnailMail'.

1.2.2 Optional Form Variables

You can post any combination of the following variables to the LinkIt API but you should set at least one of either mobileNumber, email or a postal address and postcode.

Variable: `over18`

The over18 field can be used to check that the DOB entered into the form is over 18 years of age. To allow this check to be performed by the API, set this field to Y otherwise set it to N

Variable: `mandatoryFields`

Here You can specify any "optional" form variables that you'd like to make mandatory, the API will check that data has been entered in all mandatoryFields and return a suitable error message if the user has not filled out the mandatory fields you specify here. Please see example section for correct usage.

Variable: `returnedURL (not actioned under version 6Raw)`

On a successful form submission this is the URL the customer will be redirected to. If this is not set then formURL will be used. This is only applicable to forms using version 6, 6Raw will not redirect.

Variable: `errorURL (not actioned under version 6Raw)`

On a form submission error this is the URL the customer will be redirected to. If this is not set then formURL will be used. This is only applicable to forms using version 6. 6Raw will not redirect.

Variable: `allowSMS`

The value for this must be Y or N. If set to Y the customer will receive SMS marketing. This value must be displayed to the customer, allowing them to opt-out of SMS marketing unless you default this value to N. `allowSMS` must be in variable `optinFields` to be processed.

Variable: `allowCall`

The value for this must be Y or N. If set to Y the customer will receive tele marketing. This value must be displayed to the customer, allowing them to opt-out of tele marketing unless you default this value to N. `allowCall` must be in variable `optinFields` to be processed.

Variable: `allowEmail`

The value for this must be Y or N. If set to Y the customer will receive email marketing. This value must be displayed to the customer, allowing them to opt-out of Email marketing unless you default this value to N. `allowEmail` must be in variable `optinFields` to be processed.

Variable: `allowSnailMail`

The value for this must be Y or N. If set to Y the customer will receive postal marketing. This value must be displayed to the customer, allowing them to opt-out of Mail marketing unless you default this value to N. `allowSnailMail` must be in variable `optinFields` to be processed.

Variable: `eflyerReplyID`

The value for this must be numeric and be an Eflyer ID assigned to your unit. If set the customer will receive an email once the form is successfully submitted.

Variable: `tflyerReplyID`

The value for this must be numeric and be a Tflyer ID assigned to your unit. If set the customer will receive an sms once the form is submitted successfully submitted.

Variable: `UDFIDs[n]`

This variable allows you to post udf "user defined field" values to the LinkIt API. Where [n] is the udfID and the value is set to a string. You can request your current udfID's by emailing support@powertext.co.uk Please see example section for correct usage.

1.2.3 Variable Specification

Field name	Max length	Type	Mandatory
title	-	Mr,Mrs,Ms,Dr,Miss,Prof	N
firstName	30	Alphanumeric	N
lastName	20	Alphanumeric	N
gender	1	M,F	N
dobDay	2	01 – 31	N
dobMonth	2	01 – 12	N
dobYear	4	Numeric	N
email	50	Alphanumeric	N
mobileNumber	15	Numeric	N
mandatoryFields	-	Field1,field2	N
sourceID	-	Array	Y
formURL	-	Alphanumeric	Y
returnURL	-	Alphanumeric	N
errorURL	-	Alphanumeric	N
groupIDs[]	-	Array	Y
buildingName	30	Alphanumeric	N
buildingNumStreet	30	Alphanumeric	N
locality	30	Alphanumeric	N
city	20	Alphanumeric	N
county	30	Alphanumeric	N
postCode	10	Alphanumeric	N

membershipType	255	Alphanumeric	N
membershipNumber	255	Alphanumeric	N
over18	1	Y,N	N
version	6	6, 6Raw	Y
UDFIDs[]	-	Array	N
optinFields	255	Alphanumeric	Y
allowSMS	1	Y,N	N*
allowCall	1	Y,N	N*
allowEmail	1	Y,N	N*
allowSnailMail	1	Y,N	N*
tflyerReplyID	-	Numeric	N
eflyerReplyID	-	Numeric	N

* Allow fields are mandatory where they are defined within optinFields

1. 3 Response: Return Fields

Each request to the LinkIt API will respond with a “status” and “message” variable. Depending on the version used (6,6Raw) will depend on how data is returned. The standard version “6” will respond via redirecting to the returnedURL, errorURL or formURL. The variables will be appended as GET variables. Version “6Raw” will respond with an serialized json object as a string.

1.3.1 Response: Variable Specification

Field name	Values
status	OK,ERROR
message	Message describing status / error

1.3.2 Response: Version 6 / GET

When using the standard version “6” your form will submit directly to the API URL before being redirected back to one of the following: returnedURL, errorURL or formURL variables. In addition to status and message variables, all data submitted to the API in the request will also be returned in the redirect url as GET variables.

/index.html?status=ERROR&message=Form+has+not+been+filled+in&accountID=3&sourceID=116&groupIDs[1]=275&formURL=http%3A%2F%2Fpt5.davidwales.co.uk%2FDebugScripts%2FLinkITDebug.html&mandatoryFields=mobileNumber%2Cemail%2CdobDay%2CdobMonth%2CdobYear%2CUDFIDs[480]&allowSMS=Y&allowCall=Y&allowEmail=Y&over18=N&version=6&gender=&UDFIDs[480]=&firstName=&lastName=&email=&mobileNumber=&dobDay=&dobMonth=&dobYear=&submit=submit

Or

/index.html?status=OK&message=Thank+you

1.3.3 Response: Version 6Raw

When using the ajax version “6Raw” your form will submit to your own website via javascript / ajax and your site must relay the data to the API using the script from section 2.2.1 or similar. The API will respond with raw output which can be processed by your own javascript / ajax.

```
status=ERROR&message=Form+has+not+been+filled+in&sourceID=0000&accountID=0000&mandatoryFields=firstName%2ClastName%2CmobileNumber%2Cemail%2CdobDay%2CdobMonth%2CdobYear%2Cgender&over18=Y&groupIDs[1]=0000&formURL=http%3A%2F%2Fwww.facebook.com%2F&version=6Raw&firstName=&lastName=&email=&mobileNumber=&dobDay=&dobMonth=&dobYear=&gender=&allowSMS=Y&allowEmail=Y&allowCall=Y

Or

status=OK&message=Thank+you
```

1. 4 Examples: Standard Forms

This is an example of a simple version 6 form layout, collecting email, mobileNumber, optin options and a UDF.

```
<html><head><title>Form</title></head>
<body>
  <!-- Output message to screen -->
  <p><?php if(isset($_GET['message'];)) { echo $_GET['message']; } ?></p>

  <!-- Start Form -->
  <form action='http://www.powertext.co.uk/LinkIt/processor.php' method="POST">

  <!-- hidden required variables -->
  <input type="hidden" value="1" id="accountID" name="accountID" />
  <input type="hidden" value="123" id="sourceID" name="sourceID" />
  <input type="hidden" value="1234" id="groupIDs" name="groupIDs[1]" />
  <input type="hidden" value="http://my.url.com/form.html " name="formURL" />
  <input type="hidden" value="email,UDFIDs[12345] " name="mandatoryFields" />
  <input type="hidden" value="N" id="over18" name="over18" />
  <input type="hidden" value="6" name="version" />
  <input type="hidden" value="allowSMS,allowEmail" name="optinFields" />

  <!-- visible user input variables -->
  Email: <input type="text" name="email" value="" id="email"  />
  Mobile: <input type="text" name="mobileNumber" value="" id="mobileNumber"  />
  UDF: <input type="text" name="UDFIDs[12345]" value="" id="udf12345"  /><br />

  <!-- visible opt-in / out checkboxes -->
  Receive SMS <input type="checkbox" value="Y" name="allowSMS" id="allowSMS"
class="checkboxes" checked />
  Receive Email <input type="checkbox" value="Y" name="allowEmail"
id="allowEmail" class="checkboxes" checked />

  <!-- submit button -->
  <input type='submit' name='submit' value='submit' />
  <!-- End Form -->
  </form>
</body>
</html>
```

This is an example of a simple version 6 form layout which allows the user to select which additional group they are written to. All submissions would be written to groupID 1234 and then also to which ever group is selected.

```
<html><head><title>Form</title></head>
<body>
  <!-- Output message to screen -->
  <p><?php if(isset($_GET['message'];)) { echo $_GET['message']; } ?></p>

  <!-- Start Form -->
  <form action='http://www.powertext.co.uk/LinkIt/processor.php' method="POST">

  <!-- hidden required variables -->
  <input type="hidden" value="1" id="accountID" name="accountID" />
  <input type="hidden" value="123" id="sourceID" name="sourceID" />
  <input type="hidden" value="1234" id="groupIDs" name="groupIDs[1]" />
  <input type="hidden" value="http://my.url.com/form.html " name="formURL" />
  <input type="hidden" value="email,groupIDs[2]" name="mandatoryFields" />
  <input type="hidden" value="N" id="over18" name="over18" />
  <input type="hidden" value="6" name="version" />
  <input type="hidden" value="allowSMS,allowEmail" name="optinFields" />

  <!-- visible user input variables -->
  Email: <input type="text" name="email" value="" id="email"  />
  Mobile: <input type="text" name="mobileNumber" value="" id="mobileNumber"  />
  Group:  <select name="groupIDs[2]">
            <option value="0">-- please select --</option>
            <option value="12345">groupA</option>
            <option value="12346">groupB</option>
          </select><br />

  <!-- visible opt-in / out checkboxes -->
  Receive SMS: <input type="checkbox" value="Y" name="allowSMS" id="allowSMS"
class="checkboxes" checked />
  Receive Email: <input type="checkbox" value="Y" name="allowEmail"
id="allowEmail" class="checkboxes" checked />

  <!-- submit button -->
  <input type='submit' name='submit' value='submit' />
  <!-- End Form -->
  </form>
</body>
</html>
```

2. Booking Data Forms

In addition to the standard contact data elements, defined in the previous sections, the LinkIT API can be used to store Party Booking / Enquiry data.

2.1 Booking Form Variables

Please note that ALL variable names are case sensitive and must be presented in your form in the correct case, as per the documentation.

2.1.1 Mandatory Booking Form Variables

Your form must contain ALL of the variables outlined in section 1.2.1 in addition to ALL of the variables listed below.

Variable: `bookingenquiry`

This variable should be a hidden variable and must be set to a value of "1". This is used to tell the API to expect booking data in the request. Without this field and value the API will not save booking data elements.

Variable: `booking_enquiry_source_id`

This variable should be set to one of the pre-defined values which best suits the location of the form. To add a new source id to the pre-defined list, please contact support@powertext.co.uk and your request will be reviewed.

```
mysql> select * from booking_enquiry_source;
+---------------------------+-----------------------------+
| booking_enquiry_source_id | booking_enquiry_source_name |
+---------------------------+-----------------------------+
|                         1 | Website                     |
|                         2 | Facebook                    |
+---------------------------+-----------------------------+
```

Variable: `booking_type_id`

This variable should be set to one of the pre-defined values. To add a new booking_type_id to the pre-defined list, please contact support@powertext.co.uk and your request will be reviewed.

```
mysql> select * from booking_type;
+-----------------+-------------------+
| booking_type_id | booking_type_name |
+-----------------+-------------------+
|               1 | Birthday          |
+-----------------+-------------------+
```

Variable: `booking_party_size`

The number of people attending the party / booking. If you are not collecting this data from the customer you must provide a default value, possibly within a hidden field.

Variables: `booking_party_date_Day,booking_party_date_Month,booking_party_date_Year`

The date the customer would like to book for their party / function. If you are not collecting this data from the customer you must provide a default value to the API.

Variables: `booking_party_date_Hour,booking_party_date_Minute`

The time the customer would like to arrive for their party / function. If you are not collecting this data from the customer you must provide a default value to the API.

2.1.2 Optional Booking Form Variables

Variable(s): `booking_data_element_01 through booking_data_element_20`

In addition to the standard variables above, the booking system allows the storage of an additional 20 fields. PowerText can assign these fields as and when required but once assigned they must continue to store the same data set.

For example, you may wish to collect an extra booking specific data element from your customers such as "party name". To do this you must contact support@powertext.co.uk and request an "extended booking data map" on your PowerText account. PowerText will then provide you with one of your available 20 element slots and map the required backend parameters so that future search functionality and reporting is enabled.

Variables: `special_requirements`

Add a note to the booking request to record any special requirements the user may ask for, this is plain text and will add a note to the booking thread on creation. This data is not available for use within broadcast search criteria.

2. 2 Examples: Booking Forms

All booking forms must be submitted as version 6Raw. This means you must use javascript to submit your form to a script on the same web address as your form is served from. The script must relay the full POST request to the LinkIT API and output the response so that your javascript can read the response.

2.2.1 Example: PHP AJAX relay script

The example PHP code below will relay all ajax POST data from your form to the LinkIT API and output the API response for your javascript form function to process.

```
<?php

// PHP LinkIT API Proxy V3 2012.
// Responds to HTTP POST requests
// Requires: Curl

// define LinkIT API URL
define ('API_URL', 'http://www.powertext.co.uk/LinkIt/processor.php');

// Open the Curl session
$session = curl_init(API_URL);
$postvars = http_build_query($_POST);
curl_setopt ($session, CURLOPT_POST, true);
curl_setopt ($session, CURLOPT_POSTFIELDS, $postvars);

// Don't return HTTP headers. Do return the contents of the call
curl_setopt($session, CURLOPT_HEADER, false);
curl_setopt($session, CURLOPT_RETURNTRANSFER, true);

$xml = curl_exec($session);  // Make the call and proxy data
curl_close($session);   // close the session

// stop google etc accessing this file by accident, if the default error
is returned and accountID is not posted
if(trim($xml) == 'Serialized array to CentralPoint was emptySerialized
array to CentralPoint was empty' && !isset($_POST['accountID'])) {
    header("HTTP/1.0 404 Not Found");
    header('Content-Type: text/html',true);
    echo ' <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
        <html><head> <title>404 Not Found</title> </head>
        <body> <h1>Not Found</h1> <p>The requested URL was not found on
this server.</p> <hr> </body></html> ';
} else {
    // Relay the response to ajax call
    header('Content-type: text/plain');
    echo trim($xml);
}

?>
```

2.1.2 Example: Javascript

The example javascript function below will POST data to the PHP Ajax Relay Script located on your server and process the response.

```
function send_data(myFormID,successMSG,successURL,errorMSG) {

    var myurl = '/ajax/linkitajax.php'; // location of php Ajax Script

    $.ajaxSetup({async:false});     // setup ajax to be synchronous

    /* start jQuery post of the data */
    $.post(myurl,

           $('#'+myFormID).serialize(true),

           function(data) {

               /* create an array from return string */
               var dataAr = [];
               dataAr = data.split('&');

               /* extract the status from return data array */
               var tmpAr = [];
               tmpAr = dataAr[0].split('=');
               var status = tmpAr[1];

               /* extract the message from return data array */
               tmpAr = [];
               tmpAr = dataAr[1].split('=');
               var tmpmsg = tmpAr[1].split('+');
               var message = unescape(tmpmsg.join(' '));

               /* try to extract the failed ObjectID element */
               var ObjectID = false;

               if(typeof dataAr[2] != 'undefined') {
                   tmpAr = [];
                   tmpAr = dataAr[2].split('=');
                   if(tmpAr[0] == 'id') {
                       ObjectID = unescape(tmpAr[1]);
                   }
               }

               /* check status and set things depending on its value*/
               switch(status)
               {
                   case "OK":
                       if(!successURL) {
                           if(successMSG != false) {
                               message = successMSG; // override
```

```
                        }

                    } else {
                        window.location.replace(successURL); // divert
                    }
                break;

                case "ERROR":
                    // if we attempt to highlight, allow own message
                    if(document.getElementByID(ObjectID)) {
                        if(errorMSG != false) {
                            message = errorMSG;
                        } else {
                            // attempt to format message.
                            message = message.toLowerCase();
                            if(message.indexOf('must be filled in') >=
0) {
                                field = message.replace(' must be
filled in','');
                                message = ucwords(field)+' must be
filled in';
                            } else {
                                message = ucfirst(message);
                            }
                        }
                    } else {
                        if(errorMSG != false) {
                            message = errorMSG;
                        }
                    }
                break;

                default:
                    /* if status not recognised, default message */
                    message = 'An unknown error ocurred, please try
again later.';
                break;

            }

            alert(message); // display message.

            if(status == "OK") {
                return true;
            } else {
                return false;
            }
        }
    );
}
```