



ETUMOS

## FlowBoost User Guide

---

How to implement FlowBoost, best practices, and  
real-world examples to get started quickly

# Contents

<b>Overview</b>	<b>3</b>
Key Components	3
How it Works	3
<b>Getting Started with FlowBoost</b>	<b>4</b>
Creating a FlowBoost Webhook Template in Marketo	4
Configuring a FlowBoost "Laboratory" with Postman (RECOMMENDED)	5
<b>Creating a FlowBoost Webhook in Marketo</b>	<b>8</b>
Create a New Webhook	8
Create Response Mappings	8
<b>Testing FlowBoost Functions in Marketo</b>	<b>10</b>
Creating your Marketo Test Bed Template	10
Testing a FlowBoost Webhook	10
<b>Sample FlowBoost Functions</b>	<b>11</b>
"Hello World!"	11
Text Capitalization	12
Mathematical Calculations	12
Extract Domain from Email	13
Create a Counter	13
Bucket People into Evenly Distributed Groups	14
Bucket People into a Random Cohort	14
Removing Values from Multi-Value Field	15
Search and Replace (Simple)	15
Search and Replace - Advanced (Infer Industry from Company)	16
Look Up Values and Replace Using a List	17
Lookup Values and Replace Using a List in a CSV File	18
Limit the Number of Registrants for an Event (ADVANCED)	19
<b>Additional Ideas for Flowboost Solutions</b>	<b>20</b>
Clean up your data	20
Replace Country and State Values with 2-digit Codes	20
Calling multiple webhooks in a single FlowBoost Call	20
Create unique promo codes for promotions	20
Autofill Custom Form Fields	20
Using Flowboost Outside of Marketo	20
Have other ideas?	20
<b>Appendix</b>	<b>21</b>
Additional Resources	21
Full Country Conversion Code	21
Revision Log	26

# Overview

FlowBoost is an easy-to-use helper service that extends Marketo and other applications with all of the power, utility, and simplicity of JavaScript - one of the most universal programming languages in use today. Using easy-to-use webhooks, FlowBoost opens up an enormous range of new possibilities for performing the data transformation, cross-application communication, advanced calculation and sophisticated data processing needed by today's modern marketer. To learn more about Flowboost including common use cases, pricing and more visit our Product Overview page [here](#).

## Key Components

A completely configured FlowBoost execution consists of the following key components:

- **Webhook** - A single webhook call with custom HTTP headers ( Content-Type and FlowBoost's X-API-Key for authentication) and POST payload (JavaScript function that should be executed).
- **FlowBoost Service** - A properly licensed connection to FlowBoost is required. The service receives the webhook containing the JavaScript payload. The payload may include Marketo Lead, Company, Program or Campaign tokens , and Text `{{my.tokens}}`.
- **Data Mapping** - FlowBoost responds with a JSON object. In Marketo, this can be mapped back to Lead, Company, and Program Member fields. Outside of Marketo, the application sending the FlowBoost request must be able to process the JSON response from FlowBoost.
- **Trigger** - In Marketo, the FlowBoost Webhook is configured as a flow step in any requestable Smart Campaign. If using FlowBoost from an app other than Marketo, consult that product's webhook guidelines..

## How it Works

The following outlines how a FlowBoost request works **using Marketo**:

1. FlowBoost webhook is configured in the Admin panel of Marketo for each function that needs to be performed. These webhooks include the function as the payload, a custom header, and data mappings for each of the variables defined in the function that is sent in the payload.
2. The webhook is then triggered within a Marketo flow.
3. Whenever a person within Marketo triggers the flow containing the webhook, the webhook gathers any person or company data fields (tokens) needed and sends the request to FlowBoost.
4. FlowBoost receives the request, extracts the JavaScript and executes.
5. If function processing takes longer than 30 seconds, the request will time out and return an error.
6. If function processing completes successfully, FlowBoost returns a JSON response containing the current value of any variable declared in the original call. All variables have the same name as within your webhook code.
7. Marketo parses FlowBoosts JSON response, writing data into mapped fields as the initial campaign flow continues.

# Getting Started with FlowBoost

## What you'll need

When using FlowBoost within Marketo, before getting started ensure that you have the following:

- The FlowBoost private and public license key(s) delivered by Etumos upon successful signup. NOTE: Never use your private key to access FlowBoost from a public website. The public key should be used for this use case.
- The FlowBoost endpoint URL - <https://api.teknkl.com/flowboost/v19/run>
- Administrative Access to create webhooks

## Creating a FlowBoost Webhook Template in Marketo

As you begin to use FlowBoost, it is helpful to think of each webhook as an app or function. You will create webhooks for each task that you would like FlowBoost to perform. You will likely have a handful of FlowBoost Webhooks as part of your initial implementation. For this reason, it is helpful to configure a FlowBoost WebHook Template.

1. Login to your Marketo instance at <https://app.marketo.com>.
2. Select the Admin tab. From the Admin Center panel select "Webhooks" or from the left navigation menu scroll down to Integration >> Webhooks.
3. Create a "New Webhook" with the following parameters.
  - a. **Webhook Name:** FlowBoost - [Template]
  - b. **Description:** Template used to create new FlowBoost functions. FlowBoost accepts Javascript functions to perform complex calculations and data manipulation outside of Marketo. More information is available at <https://etumos.com>. NOTE: THIS FLOW NOT FOR USE IN PRODUCTION FLOWS AS IT PERFORMS NO FUNCTION.
  - c. **URL:** <https://api.teknkl.com/flowboost/v19/run?authoringEnv=standard>.

---

**IMPORTANT:** While the authoringEnv parameter is not required, explicitly setting it to a value of standard will ensure that all editions of FlowBoost, including Pro, will automatically return all global variables as values. If not set, when using Pro edition you will need to manually 'return' the values you wish to retrieve from FlowBoost.
  - d. **Request Type:** POST
  - e. **Template:** /\* Place the valid Javascript desired here. Marketo tokens may be used to assign values to variables. Remove this text in the final webhook. Need help, access FlowBoost documentation from <https://etumos.com> \*/
  - f. **Request Token Encoding:** JSON
  - g. **Response Type:** JSON

---

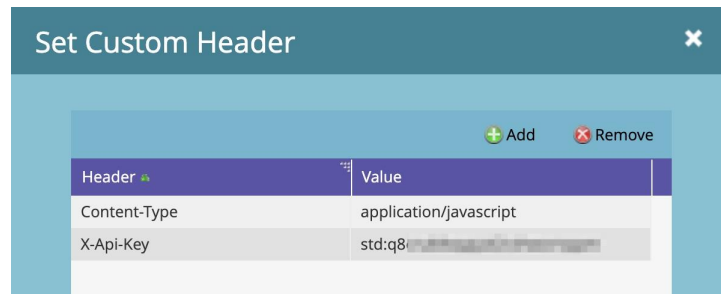
**NOTE:** We recommend using the exact template above with a verbose description and commented template text to ensure ease of use for anyone that would use FlowBoost in the future or who may wonder what the platform is.

---

4. Click "Save". **IMPORTANT:** Whenever you save a Webhook, Marketo refreshes the interface and automatically selects the first webhook in your list. From the right panel menu, find and select

your new webhook before continuing.

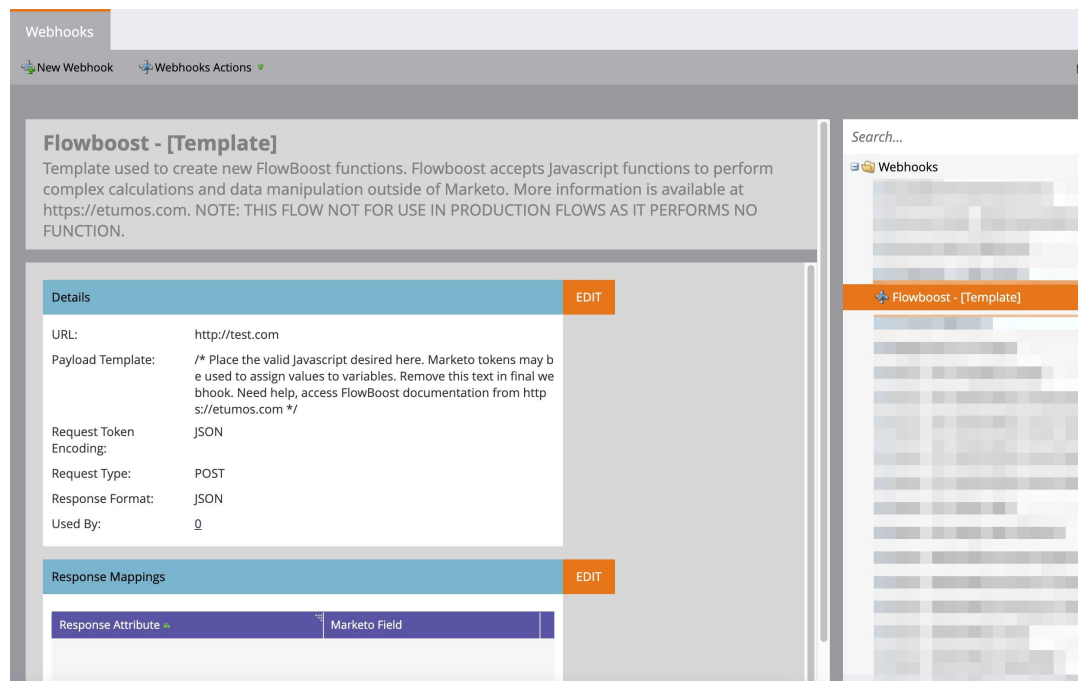
5. Click "Webhook Actions" in the contextual menu along the top edge of the Webhooks panel and select "Set Custom Header".
6. Click "+Add". Enter "Content-Type" for Header text and "application/javascript" as the Value.
7. Click "+Add". Enter "X-API-Key" for Header text and your FlowBoost access key as the Value.
8. Your custom headers should be configured as below:



The "Set Custom Header" dialog box shows a table with two columns: "Header" and "Value". There are two rows of headers added. The first row has "Content-Type" as the header and "application/javascript" as the value. The second row has "X-API-Key" as the header and a partially visible value starting with "std:q8". Above the table are buttons for "+ Add" and "Remove".

Header	Value
Content-Type	application/javascript
X-API-Key	std:q8...

9. Click "Save". Your completed FlowBoost Webhook template should look like the below.



The "Flowboost - [Template]" configuration screen is shown. It includes a "Details" section with fields for URL, Payload Template, Request Token, Encoding, Request Type, Response Format, and Used By. There is also a "Response Mappings" section with a table for mapping response attributes to Marketo fields. The "Details" section has an "EDIT" button, and the "Response Mappings" section also has an "EDIT" button.

**Details** EDIT

URL:	http://test.com
Payload Template:	/* Place the valid javascript desired here. Marketo tokens may be used to assign values to variables. Remove this text in final webhook. Need help, access FlowBoost documentation from <a href="https://etumos.com">https://etumos.com</a> */
Request Token	JSON
Encoding:	JSON
Request Type:	POST
Response Format:	JSON
Used By:	0

**Response Mappings** EDIT

Response Attribute	Marketo Field
--------------------	---------------

## Configuring a FlowBoost "Laboratory" with Postman (RECOMMENDED)

When creating a new FlowBoost function it is recommended to have a testbed or "Laboratory" outside of Marketo or the system you are working with. This will avoid adverse effects associated with simulation data. In addition, it will speed the creation of new webhooks as you will be able to easily send requests, modify the webhook body, and see the response from FlowBoost. The recommended tool for creating such a "Laboratory" is Postman.

### Download Postman and Create an Environment

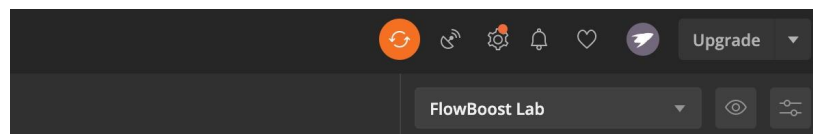
Postman is free to download. A key feature that makes Postman so indispensable to the FlowBoost user are "Environments" which enable the use of tokens similar to those in Marketo. Environments, when configured properly, make JavaScript created in Postman easier to copy, paste and use in Marketo.

1. Download and install the Postman app for Windows or Mac from <https://www.postman.com/downloads>. NOTE: You may need to create a free account. NO PURCHASE IS NECESSARY as of the authoring date of this document.
2. Click "+New" and create a new "Environment" with the name of "FlowBoost Lab".
3. To mimic Marketo lead tokens, create and set the following variables as below.

Variable	Initial Value	Current Value
privateKey	Your FlowBoost Private Key	Your FlowBoost Private Key
lead.First Name	"Hello"	"Hello"
lead.Last Name	"World!"	"World!"
lead .Email Address	"myEmail@myDomain.com"	"myEmail@myDomain.com"
lead.Company Name	"My Company Name"	"My Company Name"
lead.Country	"France"	"France"
lead.Industry	"Technology"	"Technology"
lead.Job Title	"Manager of myDepartment"	"Manager of myDepartment"
lead.Postal Code	"33309"	"33309"

**NOTE: With the exception of the privateKey variable, all variables are named exactly like Marketo token names. Keep this in mind when creating your own.**

4. Click "Add" and "X" to save your environment.
5. In the environment selector in the upper-right corner of the main Postman window, select the "FlowBoost Lab" environment.

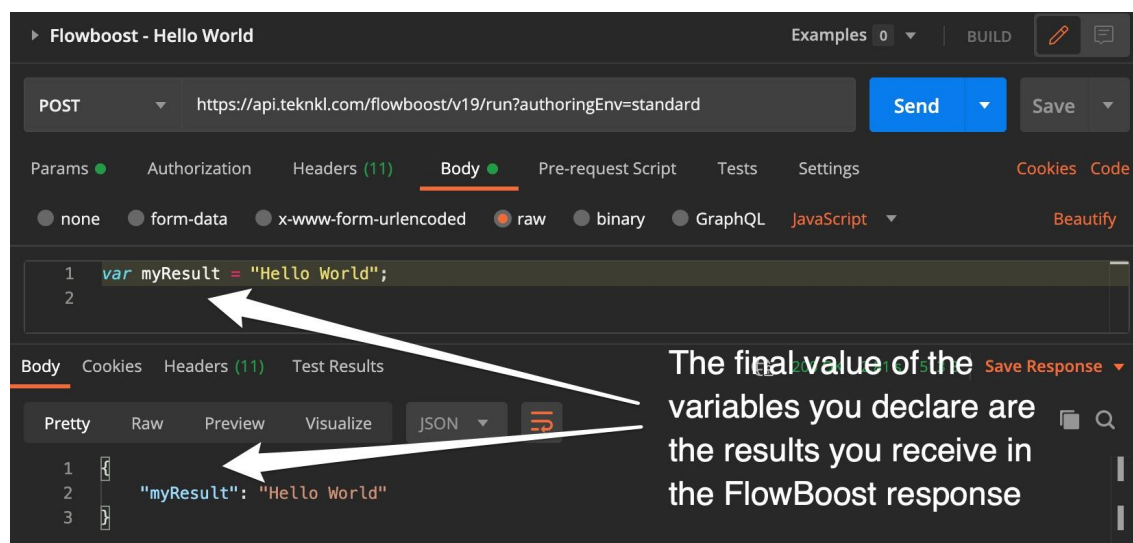


## Configure Your First Collection and Template Request

1. Click "+ New" and create a new collection with the name of "FlowBoost Functions".
2. Click "+ New" and create a new request. Name the request "FlowBoost - Hello World". At the bottom of the "Save Request" window, find and select the "FlowBoost Functions" collection.
3. Click "Save". Your new FlowBoost request appears in the main panel to the right.
4. Change the method from GET to POST.
5. Enter the request URL as <https://api.teknkl.com/flowboost/v19/run>
6. Click "Params" and enter a new key of "authoringEnv" and a value of "standard".
7. Click "Headers" and add the following new headers to the end of the list:
  - a. KEY: "Content-Type" and VALUE: "application/JavaScript"
  - b. KEY: "x-api-key" and VALUE: {{privateKey}}
8. Click "Body" and select the "raw" radio button.
9. In the code area below, type the following:

```
var myResult = "Hello World!";
```

10. Click "Save" and click "Send". You should see a similar setup to the below.



11. Now, let's use the environment variables we created by replacing the code in line 1 with:

```
var myResult = {{lead.First Name}} + " " + {{lead.Last Name}}
```

12. Click "Save" and "Send". Notice how Postman simulates how a Marketo Webhook works by replacing the values for each environment variable before sending the request to FlowBoost. Since the Postman notation and variables are the same as Marketo, this code can be copied and pasted into a Marketo Webhook with no modification.
13. To create new FlowBoost test requests in Postman, right-click "FlowBoost - Hello World" and click duplicate. Create any Marketo tokens needed as environment variables in Postman. Write your JavaScript including the appropriate Postman variables. Test and finalize in Postman and simply copy and paste for code into Marketo for final testing before using in production.

# Creating a FlowBoost Webhook in Marketo

## What you'll need

- Marketo Webhook FlowBoost Template
- Your Javascript code (custom code created by you in your [Postman FlowBoost Laboratory](#), one of the sample functions in this doc, or any other custom Javascript function)
- Fields that you intend to use in your FlowBoost function should be available in Marketo
- Fields that you intend to map responses to should be available in Marketo

## Create a New Webhook

It is highly recommended that you create and test your Javascript code in a [Postman FlowBoost Laboratory](#) before creating a new webhook.

1. Login to Marketo and open the Admin panel.
2. Open Integration and Webhooks.
3. Select the [FlowBoost - \[Template\] webhook](#) you previously created.
4. From the "Webhooks Actions" menu, select "Clone Webhook".
5. Give your new Webhook and name such as "FlowBoost - [What it does]". If desired, enter a description that explains what this FlowBoost Webhook does. Click "Clone"
6. Copy and paste your Javascript code directly into the "Payload Template" field.
7. Replace values as needed with Marketo tokens by following the steps below for each:
  - a. Select the code you wish to replace with a token.
  - b. Click "Insert Token".
  - c. Search for and select an available Token from the list.
  - d. As a best practice, set a default value that will cause your code to fail gracefully if no value exists in the field when the webhook runs for any person.
  - e. Click "Insert".
  - f. Repeat for all variable values that require the use of Marketo values.
  - g. Click "Save" to save your new Marketo Webhook.
8. Ensure that your new Marketo Webhook is selected in the right-hand navigation.
9. Proceed to "Creating Response Mappings".

## Create Response Mappings

Your response from FlowBoost will likely contain data that you want to store in your Marketo database. The response sent back from FlowBoost will be a JSON object containing all of the global variables that you defined within the Javascript you sent in the payload of the Marketo Webhook. Some or all of these values can be mapped to Marketo fields for storage.

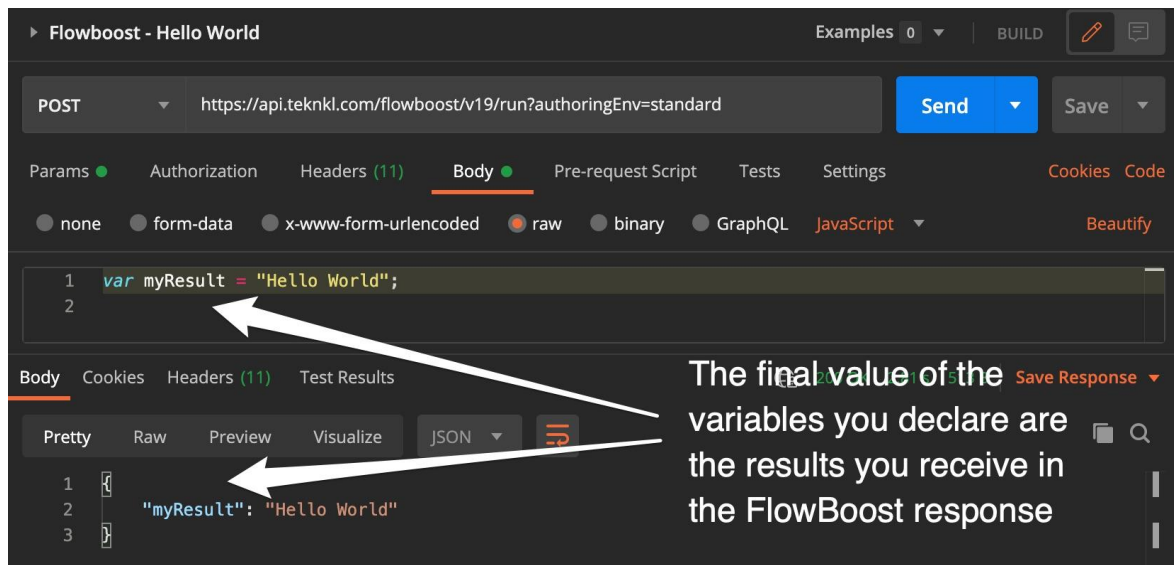
---

**NOTE:** The webhook template created with the steps above uses the `?authoringEnv=standard` parameter in the FlowBoost URL. The 'standard' authoring environment automatically returns all declared **global variables** created with 'var'. Variables created with 'let' or 'const' will not be automatically returned and must be returned using the 'return' statement. The 'return' statement must also be used to include ANY global variable value in the FlowBoost response when using the 'pro' authoring environment. Using 'let' and 'const' are considered good practices in modern code and provide the benefit of keeping the FlowBoost response JSON clean.

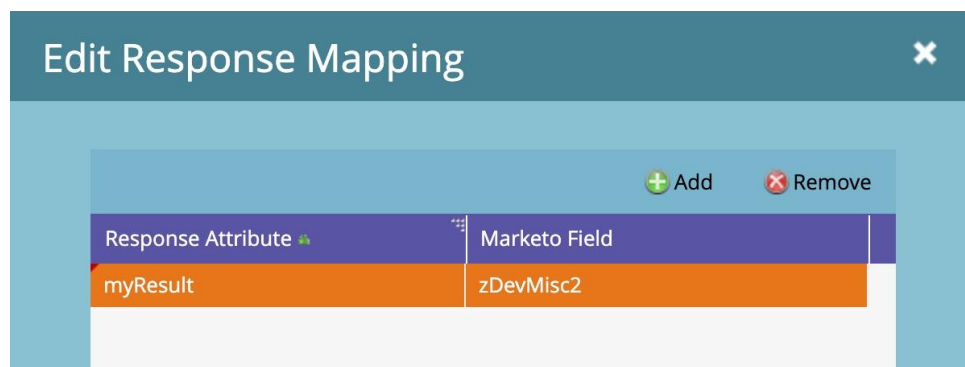
---



In the example below, we see our Postman FlowBoost Laboratory test for "Hello World". Notice that FlowBoost returns the variable back to us in the JSON response. If 'let' or 'const' were used in place of 'var', then 'return myResult;' would need to be added in order to see myResult in the JSON response.



Though you cannot see this response in Marketo, when FlowBoost responds to a request, Marketo will receive a similar response and the response mapping will tell Marketo what to do with the data. To map the FlowBoost response for the variable "myResult" to a lead/person field in Marketo called "zDevMisc2," we would select the Marketo Webhook for the appropriate FlowBoost function and click "Edit" in the Response Mappings section. Click "+ Add". In the Response Attribute field, type the case sensitive name of the variable you wish to store (e.g. "myResult"). In the Marketo Field, search for the field you wish to store the value into (e.g. "zDevMisc2"). When creating webhooks that return multiple values, create multiple response mappings and click "Save".



# Testing FlowBoost Functions in Marketo

Testing should be performed prior to production use of FlowBoost functions. While there are many ways to test Webhooks within Marketo, the suggestion below is but one of those methods.

## What you'll need

- Test Program containing a test Form, Landing Page and SmartCampaign
- [FlowBost Webhook Template](#) and optionally, a configured FlowBoost Webhook.

## Creating your Marketo Test Bed Template

1. In Marketing Activities, create a folder called "FlowBoost". Within this folder, create another called "Test Lab".
2. Create a new program called "Test Bed Template". Use channel and audience settings that provide the most flexibility and broadest scope within your Marketo instance.
3. Right click on your "Test Bed Template" program and create a new local asset as a form called FBForm. Follow the wizard to create a simple form that includes the email address field. Save and approve the form.
4. Right click on your "Test Bed Template" program and create a new local asset as a landing page called FBLP. Create a simple landing page that includes FBForm. Save and approve the landing page.
5. Right click on your "Test Bed Template" program and create a Smart Campaign called FBFlow.
  - a. Click the "Smart List" tab and add "Fills Out Form" as a trigger with "Test Bed Template.FBForm" selected as the form name.
  - b. Click the "Flow" tab and add "Call Webhook" as a flow step. Select "FlowBoost - [Template]" as the webhook to call.
6. Your new FlowBoost Test Lab and Test Bed Template should appear as below.



## Testing a FlowBoost Webhook

1. Clone your "Test Bed Template" and name it the same as the webhook you are testing.
2. Ensure that the FBForm in your new Test Bed has any fields relevant to your webhook.
3. Ensure that the FBFlow Campaign Flow step Calls the correct FlowBoost Webhook.
4. Click the "Schedule" tab and "Activate" your campaign.
5. Visit your Test Bed's FBLP landing page and submit the form.
6. In the "Results" tab, click on the person you created and verify that the webhook performed as expected by observing person fields and inspecting the person's activity log.

## Sample FlowBoost Functions

The sample code below assumes that you have created a FlowBoost Webhook Template in Marketo and/or a Postman Laboratory for testing. Each example has the following format:

<b>What it does</b>	An explanation of what the code below does.
<b>Use case</b>	An example of how you might use this code.
<b>Additional Setup Info</b>	If additional configuration outside of the Javascript is required it will be detailed in this field.
<b>Payload Template</b>	Code sample for use in your Postman Request Body or Marketo Webhook Payload Template field. Make sure that the Marketo Tokens, denoted as <b>{{marketoObject.fieldName}}</b> , match those that are available in your instance and, if using a Postman Laboratory, are created as Postman Environment variables.
<b>Example Response</b>	An example of the JSON response you should receive if the code is working correctly.
<b>Response Mappings</b>	Guidance on which values to map back to Marketo fields

### "Hello World!"

<b>What it does</b>	Basic code that makes FlowBoost respond back with Hello World!
<b>Use case</b>	Useful as a first example to learn how to send Javascript and to visualize how FlowBoost will respond in JSON.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>var myResult = "Hello World!";</pre>
<b>Example Response</b>	<pre>{   "myResult": "Hello World" }</pre>
<b>Response Mappings</b>	None

## Text Capitalization

<b>What it does</b>	Capitalizes all text in a value
<b>Use case</b>	Using standard uppercase for names and identifying text is useful for direct mail and also for analysis and reporting where case-sensitive grouping is used.
<b>Payload Body/Template</b>	<pre>var cappedText = {{lead.Last Name}}.toUpperCase();</pre>
<b>Example Response</b>	<pre>{   "cappedText": "WORLD!" }</pre>
<b>Response Mappings</b>	cappedText = Your last name or some other text field in Marketo

## Mathematical Calculations

<b>What it does</b>	Adds together multiple numerical values for score
<b>Use case</b>	A simple example of performing basic calculations using FlowBoost. In this case, it provides the ability to combine multiple lead scores (e.g. behavior, demographic, etc.) into a single value. You can even multiply, divide, subtract or perform any number of complex calculations with Javascript. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>var finalScore = {{lead.leadScore_1}} +   {{lead.leadScore_2}} +   {{lead.leadScore_3}}</pre>
<b>Example Response</b>	<pre>{   "finalScore": "235" }</pre>
<b>Response Mappings</b>	finalScore = Your custom final score field in Marketo

## Extract Domain from Email

<b>What it does</b>	Extracts the domain portion of a given email address
<b>Use case</b>	Domain is usually used to request account-level enrichment from 3rd party services. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>var domainMatch = {{lead.Email Address}}.match(/@(.+)/).pop();</pre>
<b>Example Response</b>	<pre>{   "domainMatch": "myDomain.com" }</pre>
<b>Response Mappings</b>	domainMatch = Your custom domain field in Marketo

## Create a Counter

<b>What it does</b>	Increments a running count by 1 each time you call it
<b>Use case</b>	Great for counting the number of times an activity took place, such as the number of times an activity has occurred. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>let counterName = <b>"/countOf/pgm"</b> + {{program.id}} FBCounter.autoAdd(counterName)     .then( newEntry =&gt; success(newEntry.entryIndex)).catch( failure );</pre>
<b>Example Response</b>	<pre>{   "response": 6 }</pre>
<b>Response Mappings</b>	response = {{lead.myField}} or any field where you wish to store this data. This value is stored in FlowBoost and can be retrieved from other webhooks. To retrieve the current value of a counter, create a webhook with the URL of ?authoringEnv=pro and use the following in your JS:  <pre>FBCounter.count("/demo-counters/simple").then(success)</pre>

## Bucket People into Evenly Distributed Groups

<b>What it does</b>	Returns a number corresponding to a group.
<b>Use case</b>	Great for splitting a group of leads into evenly distributed groups. The example below will return 0, 1, 2, or 3 since there are 4 buckets. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>let counterName = <b>"/bucketMaker/MarchPromoTest/pgm"</b> + <b>{{program.id}}</b> let numBuckets = <b>4</b>; FBCounter.autoAdd(counterName)     .then( newEntry =&gt; success(newEntry.entryIndex % numBuckets)     ).catch( failure );</pre>
<b>Example Response</b>	<pre>{   "response": 1 /* Example above will return 0, 1, 2 or 3 */ }</pre>
<b>Response Mappings</b>	Response = <b>{{lead.testingBucket}}</b> or some other custom field to hold the distribution category that the person is assigned to.

## Bucket People into a Random Cohort

<b>What it does</b>	Returns a random number corresponding to a group.
<b>Use case</b>	Sometimes the Marketo "Random Sample" flow step choice isn't random enough. FlowBoost provides an alternative.
<b>Marketo Webhook Payload/Postman Body</b>	<pre>function randomInt(min,max) {   return Math.floor(Math.random()*(max+min+1)+min); } var randomNum1 = randomInt(<b>1,10</b>);</pre>
<b>Example Response</b>	<pre>{   "randomNum1": 9 }</pre>
<b>Response Mappings</b>	randomNum1 = <b>{{lead.cohortRandom}}</b> or some other custom field to hold the distribution category that the person is assigned to.

## Removing Values from Multi-Value Field

What it does	Removes an item from a list of items separated by ";"
Use case	The example below removes a product from a list of products in the Product Interest field. A similar method can be employed for any multi-value field. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
Marketo Webhook Payload/Postman Body	<pre>let engagedProducts = {{lead.Product Interest}},productToBeRemoved = <b>"myProductName"</b>; var trimmed = engagedProducts     .split( ";" )     .filter( product =&gt; product !== productToBeRemoved)     .join( ";" )</pre>
Example Response	<pre>{   "trimmed": "product1;product2;product3 " }</pre>
Response Mappings	<code>trimmed = {{lead.Product Interest}}</code>

## Search and Replace (Simple)

What it does	Searches for Text inside of a string and replaces with specified text
Use case	The example below searches for the word "The " in the string "The Department of Transportation" and replaces it with nothing. It uses a regular expression to find a match but any method supported by Javascript is possible. Regular expressions are very powerful ways of matching data. <a href="#">Learn more about Regular Expressions</a> and how to create expressions yourself. Once you know how to create them, here is a <a href="#">handy cheat sheet</a> for the syntax. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
Marketo Webhook Payload/Postman Body	<pre>var newDepartment = {{lead.Department}}     .replace(/(^<b>The</b> ) ( <b>Department</b>\$)/ig,"");</pre>
Example Response	<pre>{   "newDepartment": "Department of Transportation" }</pre>
Response Mappings	<code>newDepartment = {{lead.Department}}</code>

## Search and Replace - Advanced (Infer Industry from Company)

<b>What it does</b>	Searches for any of a list of values in a string and replaces with specified text
<b>Use case</b>	The example below searches for words in a company name that might appear to fit in an industry category and returns the matching category. It uses a case statement and regular expressions to find a match. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
<b>Marketo Webhook Payload/Postman Body</b>	<pre> let myString = {{lead.Company Name}}; var myCategory = ""; switch (true) {   /*Look for items separated by ' ' and return myCategory*/   case /.*(<b>finance financial insurance bank</b>)/i.test(myString):     myCategory = "<b>Financial Services</b>";     break;   case /.*(<b>healthcare hospital medical pharma</b>)/i.test(myString):     myCategory = "<b>Healthcare and Medical Services</b>";     break;   default:     myCategory = "" /*If nothing is found return blank*/ } </pre>
<b>Example Response</b>	<pre> {   "myCategory": "Financial Services" } </pre>
<b>Response Mappings</b>	myCategory = {{lead.Industry}} or a custom {{lead.Inferred Industry}} field.



## Look Up Values and Replace Using a List

<b>What it does</b>	Replaces text from a list of values
<b>Use case</b>	The example below matches long country names to ISO-country names ( <a href="#">See full code</a> ) but it can be modified for many purposes including state and province abbreviations, industry, department and title standardization and more. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.
<b>Marketo Webhook Payload/Postman Body</b>	<pre> var leadCountry = <b>{{lead.Country:default=}}</b>; var dataMap = [   {Original:"Afghanistan",New:"AF"},   {Original:"Åland Islands",New:"AX"},  /* Add more countries in the same format here */    {Original:"Zambia",New:"ZM"},   {Original:"Zimbabwe",New:"ZW"} ];  var dataMatch = dataMap.reverse().find(   (mapEntry) =&gt; new RegExp('^' + mapEntry.Original +     '\\s \$', 'i').test(leadCountry)); if (dataMatch.New) {var revisedCountry = dataMatch.New} else {var revisedCountry = leadCountry}; dataMap = undefined; <b>/*IMPORTANT*/</b> </pre>
<b>Example Response</b>	<pre> {   "leadCountry": "Zambia",   "dataMatch": {     "Original": "Zambia",     "New": "ZM"   },   "revisedCountry": "ZM" } </pre>
<b>Response Mappings</b>	<p>The example above returns the original value of leadCountry in the revisedCountry field if a match is not found. If you are overwriting the Country field with the returned value then create a response mapping where revisedCountry = <b>{{lead.Country}}</b></p> <p>If you are writing into a custom field and NOT overwriting the Country field then create a response mapping for dataMatch.New = the field where you wish to store the 2 country code.</p>

## Lookup Values and Replace Using a List in a CSV File

<b>What it does</b>	Looks up values in a CSV file to correct invalid data
<b>Use case</b>	The example below is used to correct email address domain typos from a list that you would create on your own. HOWEVER, this example can be extended in any number of ways. For example, you could create a file of common invalid industries or departments or titles with mappings on valid values to replace with.
<b>Additional Setup Info</b>	<p>Change your Webhook URL to <a href="https://api.teknkl.com/flowboost/v19/run?authoringEnv=pro">https://api.teknkl.com/flowboost/v19/run?authoringEnv=pro</a>. Create a CSV with contents as below and host it in a publicly accessible path. If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs.</p> <pre>typoDomain,correctedDomain gmail.co,gmail.com example.tesdt,example.test</pre>
<b>Marketo Webhook Payload/Postman Body</b>	<pre>var emailParts = FBUtil.string.partsFromEmail({{lead.Email Address}}); FBHttp   .fetch( "https://www.domain.com/files/domainTypos.csv" )   .then( FBText.CSV.autoParse )   .then( rows =&gt;     rows.find( row =&gt; row.typoDomain === emailParts.domain )   )   .then( foundDomain =&gt;     foundDomain       ? emailParts.mailbox + "@" + foundDomain.correctedDomain       : undefined     )   .then( success )</pre>
<b>Example Response</b>	<pre>{   "response": "myemail@gmail.com" }</pre>
<b>Response Mappings</b>	response = a custom Marketo field called emailAddressCorrected or whichever field you'd like to store the corrected data that will be returned.

## Limit the Number of Registrants for an Event (ADVANCED)

<b>What it does</b>	Creates a counter for the number of event registrants which can be used to prevent additional registrations on a website.
<b>Use case</b>	The example below creates a program-specific counter using FBCounter and would be triggered each time a person submits a registration form or is added to the program. The latest value of the counter can then be accessed from your website using your FlowBoost public key. If the counter reaches a specific value, you can run a function that prevents your code from loading.
<b>Additional Setup Info</b>	<p>If using the below in your Postman Laboratory remember to create the environment variables noted in bold. Also ensure that they exist in your Marketo instance as appropriate. Also modify the strings in bold-italic to match your needs. On the page that contains your Marketo event registration form, you will need to call a script whose source is the URL of the FlowBoost counter you create and increment within the Marketo Webhook each time a new user submits your registration form. The script for the example below would be:</p> <pre>&lt;script src="https://api.teknkl.com/fbcounter_v1/counter/count/[counterName]?public-api-key=[publicApiKey]&amp;cb=[callbackFunction]"&gt;</pre> <p>FlowBoost will respond as follows:</p> <pre>{   "count" : 8,   "origin" : "",   "callback" : "" }</pre> <p>The callback JS function specified in the source URL parameters will be called by FlowBoost. This custom function should take the "object.count" value in the response and compare the number to the max registrants desired. Logic can be added to show a message to potential registrants (e.g. "This event is full" or "Only 8 spots left" or "Event is full") and even hide the form or prevent it from being submitted. See <a href="#">Sample LP Code</a> and <a href="#">Marketo Forms API documentation</a> for more information.</p>
<b>Marketo Webhook Payload/Postman Body</b>	<pre>let counterName = "/regLimitCounters/forProgram" + {{program.id}} FBCounter.autoAdd(counterName)   .then( newEntry =&gt; success(newEntry.entryIndex)).catch( failure );</pre>
<b>Example Response</b>	<pre>{   "response": 6 }</pre>
<b>Response Mappings</b>	None. This value is stored in FlowBoost and is retrieved from the website each time the registration landing page is loaded. See "Additional Setup" above.

## Additional Ideas for Flowboost Solutions

### Clean up your data

Using the [CSV Lookup example](#) in this document as guidance, you can create a csv with country strings and matching 2-digit country codes and then call FlowBoost to check for matches and return the correct country codes. Doing it this way rather than in a flow step makes it easier to add incorrect or alternate country spellings to your list rather than in the criteria for the Marketo flow. This same concept may be applied to State or Province abbreviations. Here are [additional ideas](#) for addressing this need.

### Replace Country and State Values with 2-digit Codes

Using the [CSV Lookup example](#) in this document as guidance, you can create a csv with country strings and matching 2-digit country codes and then call FlowBoost to check for matches and return the correct country codes. Doing it this way rather than in a flow step makes it easier to add incorrect or alternate country spellings to your list rather than in the criteria for the Marketo flow. This same concept may be applied to State or Province abbreviations. Here are [additional ideas](#) for addressing this need.

### Calling multiple webhooks in a single FlowBoost Call

Rather than calling multiple webhooks in a flow, use a single FlowBoost call to aggregate all of them into a single step. Check out this blog post from Sanford Whiteman on the topic.

<https://blog.teknkl.com/and-yes-flowboost-can-be-that-webhook-aggregator/>

### Create unique promo codes for promotions

To ensure that you create unique promotional codes for registrants to a program like a giveaway of some sort, you can use FlowBoost and FBCounter to ensure uniqueness and to limit the number of codes available. You can even prevent the registration form from loading on your landing page whenever the number of codes is exhausted. Check out this blog post from Sanford Whiteman on the topic. <https://blog.teknkl.com/creating-and-distributing-promo-codes-in-js-flowboost/>

### Autofill Custom Form Fields

Marketo does not allow form fill on embedded forms used on a 3rd party site. FlowBoost can be used to facilitate this. Using the **FBCounter** utility, FlowBoost stores a JSON object for each user email address. This JSON object can contain known data that you would like to auto-populate after a user provides an email address. Once the person-specific FBCounter object is created, you can access it from your web site using your Public API key for Flowboost.

### Using Flowboost Outside of Marketo

If you have configured your [Laboratory in Postman](#) you've probably guessed by now that FlowBoost isn't just for Marketo. In fact, you can use FlowBoost with any application that has the ability to send Webhooks and process JSON responses. Experiment with your other applications and use cases.

[Contact us](#) if you have questions or need help.

### Have other ideas?

The possibilities with FlowBoost are endless really. If you have an idea for using FlowBoost and need help with it, [contact us](#). We're happy to talk through it with you or provide an estimate for our services.

# Appendix

## Additional Resources

- Authoring code for FlowBoost Standard vs. Pro  
<https://blog.teknkl.com/authoring-code-for-flowboost-standard-vs-pro/>
- Calling other API's using FlowBoost  
<https://blog.teknkl.com/and-yes-flowboost-can-be-that-webhook-aggregator/>
- Creating and distributing unique promo codes  
<https://blog.teknkl.com/creating-and-distributing-promo-codes-in-js-flowboost/>
- Code Anatomy: Get a real array from an array-like string (a.k.a. "list")  
<https://blog.teknkl.com/code-anatomy-real-array-from-array-like-string/>
- FlowBoost loves your line breaks  
<https://blog.teknkl.com/flowboost-loves-your-line-breaks/>
- Code Anatomy: Turning Google Sheets API results into (better) objects, Part I  
<https://blog.teknkl.com/code-anatomy-rows-to-objects-1/>
- Country-to-ISO-Code translation in FlowBoost (+ notes on text compression)  
<https://blog.teknkl.com/quick-country-name-to-iso-code-translation-in-flowboost/>
- Generating Munchkin Associator Tokens in SFDC/Apex or FlowBoost/JS  
<https://blog.teknkl.com/generating-munchkin-associator-tokens-in-sfdc-apex-or-flowboost-js/>

## Full Country Conversion Code

```
var leadCountry = {{lead.Country:default=}};
var dataMap = [
  {Original:"Afghanistan",New:"AF"},
  {Original:"Åland Islands",New:"AX"},
  {Original:"Albania",New:"AL"},
  {Original:"Algeria",New:"DZ"},
  {Original:"American Samoa",New:"AS"},
  {Original:"Andorra",New:"AD"},
  {Original:"Angola",New:"AO"},
  {Original:"Anguilla",New:"AI"},
  {Original:"Antarctica",New:"AQ"},
  {Original:"Antigua and Barbuda",New:"AG"},
  {Original:"Argentina",New:"AR"},
  {Original:"Armenia",New:"AM"},
  {Original:"Aruba",New:"AW"},
  {Original:"Australia",New:"AU"},
  {Original:"Austria",New:"AT"},
  {Original:"Azerbaijan",New:"AZ"},
  {Original:"Bahamas",New:"BS"},
```

```

{Original:"Bahrain",New:"BH"},
{Original:"Bangladesh",New:"BD"},
{Original:"Barbados",New:"BB"},
{Original:"Belarus",New:"BY"},
{Original:"Belgium",New:"BE"},
{Original:"Belize",New:"BZ"},
{Original:"Benin",New:"BJ"},
{Original:"Bermuda",New:"BM"},
{Original:"Bhutan",New:"BT"},
{Original:"Bolivia",New:"BO"},
{Original:"Bonaire",New:"BQ"},
{Original:"Bosnia and Herzegovina",New:"BA"},
{Original:"Botswana",New:"BW"},
{Original:"Bouvet Island",New:"BV"},
{Original:"Brazil",New:"BR"},
{Original:"British Indian Ocean Territory",New:"IO"},
{Original:"Brunei Darussalam",New:"BN"},
{Original:"Bulgaria",New:"BG"},
{Original:"Burkina Faso",New:"BF"},
{Original:"Burundi",New:"BI"},
{Original:"Cape Verde",New:"CV"},
{Original:"Cambodia",New:"KH"},
{Original:"Cameroon",New:"CM"},
{Original:"Canada",New:"CA"},
{Original:"Cayman Islands",New:"KY"},
{Original:"Central African Republic",New:"CF"},
{Original:"Chad",New:"TD"},
{Original:"Chile",New:"CL"},
{Original:"China",New:"CN"},
{Original:"Christmas Island",New:"CX"},
{Original:"Cocos (Keeling) Islands",New:"CC"},
{Original:"Colombia",New:"CO"},
{Original:"Comoros",New:"KM"},
{Original:"Congo",New:"CG"},
{Original:"Democratic Republic of the Congo",New:"CD"},
{Original:"Cook Islands",New:"CK"},
{Original:"Costa Rica",New:"CR"},
{Original:"Côte d'Ivoire",New:"CI"},
{Original:"Croatia",New:"HR"},
{Original:"Cuba",New:"CU"},
{Original:"Curaçao",New:"CW"},
{Original:"Cyprus",New:"CY"},
{Original:"Czech Republic",New:"CZ"},
{Original:"Denmark",New:"DK"},
{Original:"Djibouti",New:"DJ"},
{Original:"Dominica",New:"DM"},
{Original:"Dominican Republic",New:"DO"},
{Original:"Ecuador",New:"EC"},
{Original:"Egypt",New:"EG"},
{Original:"El Salvador",New:"SV"},

```

```

{Original:"Equatorial Guinea",New:"GQ"},
{Original:"Eritrea",New:"ER"},
{Original:"Estonia",New:"EE"},
{Original:"Ethiopia",New:"ET"},
{Original:"Falkland Islands",New:"FK"},
{Original:"Faroe Islands",New:"FO"},
{Original:"Fiji",New:"FJ"},
{Original:"Finland",New:"FI"},
{Original:"France",New:"FR"},
{Original:"French Guiana",New:"GF"},
{Original:"French Polynesia",New:"PF"},
{Original:"French Southern Territories",New:"TF"},
{Original:"Gabon",New:"GA"},
{Original:"Gambia",New:"GM"},
{Original:"Georgia",New:"GE"},
{Original:"Germany",New:"DE"},
{Original:"Ghana",New:"GH"},
{Original:"Gibraltar",New:"GI"},
{Original:"Greece",New:"GR"},
{Original:"Greenland",New:"GL"},
{Original:"Grenada",New:"GD"},
{Original:"Guadeloupe",New:"GP"},
{Original:"Guam",New:"GU"},
{Original:"Guatemala",New:"GT"},
{Original:"Guernsey",New:"GG"},
{Original:"Guinea",New:"GN"},
{Original:"Guinea-Bissau",New:"GW"},
{Original:"Guyana",New:"GY"},
{Original:"Haiti",New:"HT"},
{Original:"Heard Island and McDonald Islands",New:"HM"},
{Original:"Holy See (Vatican City State)",New:"VA"},
{Original:"Honduras",New:"HN"},
{Original:"Hong Kong",New:"HK"},
{Original:"Hungary",New:"HU"},
{Original:"Iceland",New:"IS"},
{Original:"India",New:"IN"},
{Original:"Indonesia",New:"ID"},
{Original:"Iran",New:"IR"},
{Original:"Iraq",New:"IQ"},
{Original:"Ireland",New:"IE"},
{Original:"Isle of Man",New:"IM"},
{Original:"Israel",New:"IL"},
{Original:"Italy",New:"IT"},
{Original:"Jamaica",New:"JM"},
{Original:"Japan",New:"JP"},
{Original:"Jersey",New:"JE"},
{Original:"Jordan",New:"JO"},
{Original:"Kazakhstan",New:"KZ"},
{Original:"Kenya",New:"KE"},
{Original:"Kiribati",New:"KI"},

```

```

{Original:"North Korea",New:"KP"},
{Original:"South Korea",New:"KR"},
{Original:"Kuwait",New:"KW"},
{Original:"Kyrgyzstan",New:"KG"},
{Original:"Laos",New:"LA"},
{Original:"Latvia",New:"LV"},
{Original:"Lebanon",New:"LB"},
{Original:"Lesotho",New:"LS"},
{Original:"Liberia",New:"LR"},
{Original:"Libya",New:"LY"},
{Original:"Liechtenstein",New:"LI"},
{Original:"Lithuania",New:"LT"},
{Original:"Luxembourg",New:"LU"},
{Original:"Macao",New:"MO"},
{Original:"Macedonia",New:"MK"},
{Original:"Madagascar",New:"MG"},
{Original:"Malawi",New:"MW"},
{Original:"Malaysia",New:"MY"},
{Original:"Maldives",New:"MV"},
{Original:"Mali",New:"ML"},
{Original:"Malta",New:"MT"},
{Original:"Marshall Islands",New:"MH"},
{Original:"Martinique",New:"MQ"},
{Original:"Mauritania",New:"MR"},
{Original:"Mauritius",New:"MU"},
{Original:"Mayotte",New:"YT"},
{Original:"Mexico",New:"MX"},
{Original:"Micronesia",New:"FM"},
{Original:"Moldova",New:"MD"},
{Original:"Monaco",New:"MC"},
{Original:"Mongolia",New:"MN"},
{Original:"Montenegro",New:"ME"},
{Original:"Montserrat",New:"MS"},
{Original:"Morocco",New:"MA"},
{Original:"Mozambique",New:"MZ"},
{Original:"Myanmar",New:"MM"},
{Original:"Namibia",New:"NA"},
{Original:"Nauru",New:"NR"},
{Original:"Nepal",New:"NP"},
{Original:"Netherlands",New:"NL"},
{Original:"New Caledonia",New:"NC"},
{Original:"New Zealand",New:"NZ"},
{Original:"Nicaragua",New:"NI"},
{Original:"Niger",New:"NE"},
{Original:"Nigeria",New:"NG"},
{Original:"Niue",New:"NU"},
{Original:"Norfolk Island",New:"NF"},
{Original:"Northern Mariana Islands",New:"MP"},
{Original:"Norway",New:"NO"},
{Original:"Oman",New:"OM"},

```



```

{Original:"Pakistan",New:"PK"},
{Original:"Palau",New:"PW"},
{Original:"Palestine",New:"PS"},
{Original:"Panama",New:"PA"},
{Original:"Papua New Guinea",New:"PG"},
{Original:"Paraguay",New:"PY"},
{Original:"Peru",New:"PE"},
{Original:"Philippines",New:"PH"},
{Original:"Pitcairn",New:"PN"},
{Original:"Poland",New:"PL"},
{Original:"Portugal",New:"PT"},
{Original:"Puerto Rico",New:"PR"},
{Original:"Qatar",New:"QA"},
{Original:"Réunion",New:"RE"},
{Original:"Romania",New:"RO"},
{Original:"Russian Federation",New:"RU"},
{Original:"Rwanda",New:"RW"},
{Original:"Saint Barthélemy",New:"BL"},
{Original:"Saint Helena, Ascension and Tristan da Cunha",New:"SH"},
{Original:"Saint Kitts and Nevis",New:"KN"},
{Original:"Saint Lucia",New:"LC"},
{Original:"Saint Martin (French part)",New:"MF"},
{Original:"Saint Pierre and Miquelon",New:"PM"},
{Original:"Saint Vincent and the Grenadines",New:"VC"},
{Original:"Samoa",New:"WS"},
{Original:"San Marino",New:"SM"},
{Original:"Sao Tome and Principe",New:"ST"},
{Original:"Saudi Arabia",New:"SA"},
{Original:"Senegal",New:"SN"},
{Original:"Serbia",New:"RS"},
{Original:"Seychelles",New:"SC"},
{Original:"Sierra Leone",New:"SL"},
{Original:"Singapore",New:"SG"},
{Original:"Sint Maarten",New:"SX"},
{Original:"Slovakia",New:"SK"},
{Original:"Slovenia",New:"SI"},
{Original:"Solomon Islands",New:"SB"},
{Original:"Somalia",New:"SO"},
{Original:"South Africa",New:"ZA"},
{Original:"South Georgia and the South Sandwich Islands",New:"GS"},
{Original:"South Sudan",New:"SS"},
{Original:"Spain",New:"ES"},
{Original:"Sri Lanka",New:"LK"},
{Original:"Sudan",New:"SD"},
{Original:"Suriname",New:"SR"},
{Original:"Svalbard and Jan Mayen",New:"SJ"},
{Original:"Swaziland",New:"SZ"},
{Original:"Sweden",New:"SE"},
{Original:"Switzerland",New:"CH"},
{Original:"Syrian Arab Republic",New:"SY"},

```

```

{Original:"Taiwan",New:"TW"},
{Original:"Tajikistan",New:"TJ"},
{Original:"Tanzania",New:"TZ"},
{Original:"Thailand",New:"TH"},
{Original:"Timor-Leste",New:"TL"},
{Original:"Togo",New:"TG"},
{Original:"Tokelau",New:"TK"},
{Original:"Tonga",New:"TO"},
{Original:"Trinidad and Tobago",New:"TT"},
{Original:"Tunisia",New:"TN"},
{Original:"Turkey",New:"TR"},
{Original:"Turkmenistan",New:"TM"},
{Original:"Turks and Caicos Islands",New:"TC"},
{Original:"Tuvalu",New:"TV"},
{Original:"Uganda",New:"UG"},
{Original:"Ukraine",New:"UA"},
{Original:"United Arab Emirates",New:"AE"},
{Original:"United Kingdom",New:"GB"},
{Original:"United States",New:"US"},
{Original:"United States Minor Outlying Islands",New:"UM"},
{Original:"Uruguay",New:"UY"},
{Original:"Uzbekistan",New:"UZ"},
{Original:"Vanuatu",New:"VU"},
{Original:"Venezuela",New:"VE"},
{Original:"Viet Nam",New:"VN"},
{Original:"Virgin Islands, British",New:"VG"},
{Original:"Virgin Islands, U.S.",New:"VI"},
{Original:"Wallis and Futuna",New:"WF"},
{Original:"Western Sahara",New:"EH"},
{Original:"Yemen",New:"YE"},
{Original:"Zambia",New:"ZM"},
{Original:"Zimbabwe",New:"ZW"}
];

```

```

var dataMatch = dataMap.reverse().find(
  (mapEntry) => new RegExp('^' + mapEntry.Original +
    '(\s|$)', 'i').test(leadCountry));
if (dataMatch.New) {var revisedCountry = dataMatch.New} else {var revisedCountry =
leadCountry};
dataMap = undefined /*IMPORTANT TO REDUCE RESPONSE PAYLOAD SIZE*/;

```

## Revision Log

07-JUN-2021	Branding update and minor formatting (VS)
06-APR-2021	Updated "Look Up Values and Replace Using a List" section. Added "Full Country Conversion Code" and "Revision Log" to appendix.
01-FEB-2021	Published