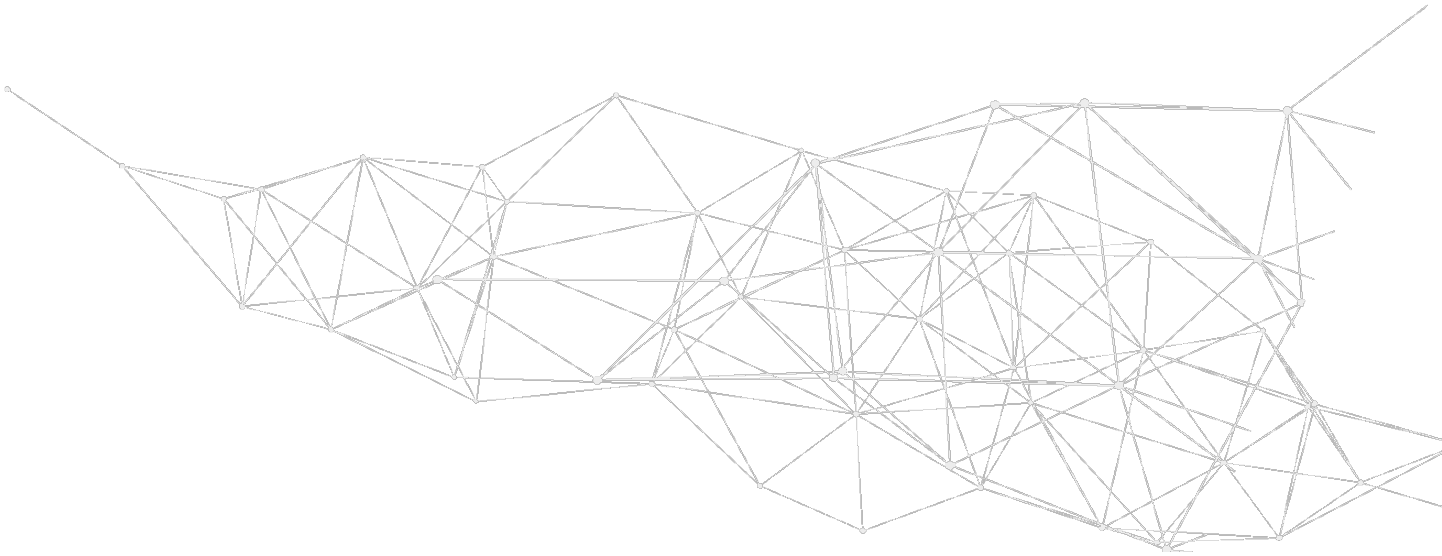
webbula
The Data Solutions Experts

Webbula Data Enhancement API Developer Guide

Version 3.1.4



5000 Stonewood Drive, Suite 310 Wexford, PA 15090 888.993.2285 Webbula.com

webbula
The Data Solutions Experts

Contents

1 | Webbula Data Enhancement Platform Overview

The Webbula Data Enhancement API is a RESTful service that provides a conduit for integrating Webbula's Data Enhancement platform with 3rd party webforms and applications. The platform has been designed to give developers the flexibility to create powerful self-serving internal tools or web based client-facing applications. The Data Enhancement API has been compatibility tested on PHP, Java, Python, C#, Go, and Perl platforms and produces responses in JSON.

1.1 | Authentication and Security

Once your Data Enhancement account has been created an Account Name and Token will be provided along with additional account details. It is vital to keep these credentials secure and hidden from the public domain. Webbula will not be held responsible for any damages due to compromised tokens.

The API requires the "Authorization: Basic" header (https://en.wikipedia.org/wiki/Basic_access_authentication) with a Base64-encoded login string. The login string is constructed by joining the Account Name and Token separated by a colon (:).

"Account Name:T0k3n"

That string needs to be Base64-encoded and added to the basic access authorization header. Continuing with the example credentials for this section, the authorization header would be:

Authorization: Basic QWNjb3VudE5hbWU6VDBrM24K

Many HTTP request libraries will handle necessary encoding and generation of the basic authorization header. For example, the cURL utility does this via the --user parameter.

For an extra layer of security the platform makes use of HTTPS. In addition, the static IP address(es) of the server(s) that will be interacting with the API must be provided to Webbula. Only the static IP addresses listed in Webbula's hardware and software firewalls will be able to successfully access the Data Enhancement API.

1.2 | Access Points

The URL endpoint is created by combining the base URL access point of the environment with the function to execute. The mask of the complete function call is:

<access_point><function>

All of the API functions can be called via HTTP POST request, including when submitting data via multipart/form-data.

The base URL access point is:

Access Point	Base URL
Production	https://api.dataenhancement.com/append/persona

To make a full request using the endpoint you will need to pass your login string into the "Authorization: Basic" header. Additionally you will need to insert your desired preset into the preset parameter in the POST data stream which will determine what will be appended. Lastly, the input values that are to be used to make the append need to be placed in the "where" object in the POST data stream.

Example:

Execute express append via POST call:

```
curl \
  -X POST \
  -H "Authorization: Basic VEVTVDE6UFJFU0VUMQ==" \
  -d
'{"preset":"E","where":{"first_name":"Homer","last_name":"Simpson","address":"742
Evergreen Terrace","zip":"12345"}}' \
 https://api.dataenhancement.com/append/persona
```

1.3 | Append Presets

The Data Enhancement API will know what data to append based upon the Preset value specified in the HTTP request. Webbula has enabled one or more of these Presets based on your use case and needs. The table below has been provided as a high-level overview of some common Presets. You can contact your account representative or support@webbula.com if you would like to enable more Presets to your account or if you would like to have a custom Preset built.

Frequently Used Append Presets:

Name	Value	Description
Email Append	E	Appends an email address
Telephone Append	T	Appends a telephone number and telephone type
Postal Append	ACSZ	Appends postal (address, city, state, zipcode)
Automotive Year, Make & Model Append	AUTOYMM	Appends automotive year, make, and model

1.4 | Test Values

The Test Value table shows records that can be used to ensure proper understanding and functionality before sending production requests.

First Name	Last Name	Address	City	State	Zip	Email	Phone
AAMIR	REHMAN	125 DOUGLAS RD	STATEN ISLAND	NY	10304	URDUONE@URDUCARDS.COM	6462380236
AARON	ARMSTRONG	3629 HIGHWAY 107	CHUCKEY	TN	37641	TLTWIN@AOL.COM	4232572953
AARIKA	GONZALEZ	PO BOX 71324	CORPUS CHRISTI	TX	78467	GON466@AOL.COM	3617260514

AARON	AGUILLARD	615 S 5TH 12 ST	NEDERLAND	TX	77627	KAGUILLARD@ADDRESS.COM	4097226824
AARON	ARTUS	1827 HIGHWAY 94	REYNOLDS	IL	61279	DARTUS@WINCO.NET	3093728964

1.5 | Starter Kits

Starter kits are available for a wide variety of platforms to reduce the integration time required. The goal is that the starter kits will demonstrate the API functions and serve as a starting point for integration. They are platform specific working code examples and the code will need to be adjusted to better serve your file system architecture.

Name	Platform
Append_Single_Starter_Kit.php	PHP
Append_Single_Starter_Kit.java	Java
Append_Single_Starter_Kit.pl	Perl
Append_Single_Starter_Kit.py	Python
Append_Single_Starter_Kit.cs	C#

2 | Persona Function

This section will outline the "*persona*" function which appends consumer related information.

2.1 | Express: Single Record Processing in Real Time

Express Mode is the processing of a single record that needs to be completed in real time. The POST request needs to have *content type* set to application/json because the function expects the input data to be in JSON format.

The Preset selected will determine what additional data the Data Enhancement API needs in order to conduct the append. Please refer to Section 3 for more details on the commonly used Presets.

Request Headers:

Header	Notes
Authorization	Your login and token login string (see section 1.1 for details)
Content-Type	Should be application/json

Input JSON Payload:

Parameter	Type	Required	Notes
preset	String	Yes	The name of the preset to use for appending. Preset defines which fields and how are going to be appended
where	Object	Yes	The container with input values and matching criteria
match	String	No	The types of matches to include in your append results. Valid values are "I" for only individual level matches; "H" for household level matches only; or "B" for both individual and household level matches. Defaults to "B" if not provided.
first_name	String	No	The first name

last_name	String	No	The last name
address	String	No	The street address
zip	String	No	The 5 digit zipcode
email	String	No	The email address
email_encoding	String	No	The encoding of the email address. The value can be one of *RAW*, *LOWER_MD5*, *UPPER_MD5*, *LOWER_SHA1*, *UPPER_SHA1*
phone	String	No	The 10-digit phone number
ip	String	No	The IP address

Output Response:

Name	Type	Description
success	Boolean	Success flag. "true" if all passed well and "false" if request failed
code	String	The code designates the state of the request. Empty on successful append
error	String	The error happened during function execution. Empty if no errors
record	Object	The container with the data appended
match	String	The exact match of the found individual
individual	Object	The container with individual data appended
auto	Object	The container with auto data appended
demo	Object	The container with demo data appended
email	Object	The container with email data appended
phone	Object	The container with phone data appended

isp	Object	The container with ISP data appended

Example:

```
curl \
  -X POST \
  -H “Authorization: Basic VEVTVDE6UFJFU0VUMQ==” \
  -d “{"preset":"E","where":{"first_name":"john","last_name":"smith","address":"789
Main Street","zip":"07058"}}” \
  https://api.dataenhancement.com/append/persona
```