

WebSmart
RBDI



WebSmart Delivery Indicator

Reference Guide

Copyright

Companies, names, and data used in examples herein are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Melissa Data Corporation. This document and the software it describes are furnished under a license agreement, and may be used or copied only in accordance with the terms of the license agreement.

© 2011. Melissa Data Corporation. All rights reserved.

Information in this document is subject to change without notice. Melissa Data Corporation assumes no responsibility or liability for any errors, omissions, or inaccuracies that may appear in this document.

Trademarks

WebSmart Delivery Indicator is a trademark of Melissa Data Corporation. Windows is a registered trademark of Microsoft Corp.

The following are registrations and trademarks of the United States Postal Service: U.S. Postal Service, United States Post Office, United States Postal Service, USPS, ZIP, ZIP Code, and ZIP + 4.

MELISSA DATA CORPORATION
22382 Avenida Empresa
Rancho Santa Margarita, CA 92688-2112
Phone: 1-800-MELISSA (1-800-635-4772)
Fax: 949-589-5211
E-mail: info@MelissaData.com
Web site: www.MelissaData.com

For the latest version of this Reference Guide, visit
<http://www.MelissaData.com/tech/websmart.htm>.

Document Code: WSRRFG
Revision Number: 131021.042
Last Update: October 21, 2013

Dear Programmer,

I would like to take this opportunity to introduce you to Melissa Data Corp. Founded in 1985, Melissa Data provides data quality solutions, with emphasis on address and phone verification, postal encoding, and data enhancements.

We are a leading provider of cost-effective solutions for achieving the highest level of data quality for lifetime value. A powerful line of software, databases, components, and services afford our customers the flexibility to cleanse and update contact information using almost any language, platform, and media for point-of-entry or batch processing.

This online manual will guide you through the properties and methods of our easy-to-use programming tools. Your feedback is important to me, so please don't hesitate to email your comments or suggestions to ray@MelissaData.com.

I look forward to hearing from you.

Best Wishes,

A handwritten signature in black ink, appearing to read "Ray Melissa". The signature is fluid and cursive, with a long horizontal stroke at the end.

Raymond F. Melissa
President

Table of Contents

| | |
|---|----|
| Welcome to WebSmart Services..... | 1 |
| An Introduction to Delivery Indicator | 4 |
| Adding Delivery Indicator to a Project | 4 |
| Submitting an XML request | 5 |
| Building a REST Request | 5 |
| Delivery Indicator Request Object | 6 |
| Request Elements | 8 |
| Delivery Indicator Response..... | 14 |
| Response Object XML Format | 22 |

1

Welcome to WebSmart Services

The WebSmart Services are a collection of services that can be accessed by any application, allowing you to incorporate Melissa Data's technology into your programs without worrying about continually downloading and installing updates.

Melissa Data currently offers the following services:

- **Address Verifier** — Verify and standardize one or more mailing address. This service also appends ZIP + 4[®] and Carrier Route information.
- **Email Verifier** — Verify, correct and update, domain names from one or more email addresses.
- **GeoCoder** — Returns geographic, census, and demographic data for almost any location in the United States.
- **IP Locator** — Returns name and geographic information for the owner of a public IP address.
- **Delivery Indicator** — Indicates whether an address represents a business or residential address.
- **Name Parser** — Parses and genderizes personal names and also generates salutations for correspondence.
- **Street Search** — Searches a ZIP Code[™] from street address ranges matching a specific pattern and, optionally, a street number.

- **ZIP Search** — Matches city names with ZIP/Postal codes, ZIP/Postal codes with city names and searches for city names matching a pattern with a given state.
- **Phone Verifier** — Verifies and parses phone numbers, as well as identifying phone numbers as residential, business, VOIP or wireless.
- **Property** — Returns basic or detailed information about the size, ownership, and structures on a given parcel of land.

Both GeoCoder and Delivery Indicator work from an “address key” returned by the Address Verifier service, therefore, an address must first be submitted to the Address Verifier before you can use either of the other two services.

There are three ways to access the WebSmart Services:

- **SOAP** — The SOAP interface allows you to add the Web Service to an application as if it were a component object or DLL. You can then access the Web Service elements and execute commands as if they were properties and methods.
- **XML** — The Web Service can also submit a request as an XML document. It will then return the processed records as another XML document that can be parsed using whatever XML tools you utilize in your development environment.
- **REST** — This interface allows you to submit a single address record as part of a URL string and returns the processed record as an XML document identical to the one returned by the XML interface.

Using the REST service may require that you encode certain characters using the proper URL entities before adding them to a URL. Characters like spaces, slashes, ampersands and others must be replaced by special codes, which usually consist of a percent sign followed by a two-digit hexadecimal number.

The following table shows the replacements for the most common characters.

| Character | URL Encoded |
|-----------|-------------|
| Space | %20 or + |
| * | %2A |
| # | %23 |
| & | %26 |
| % | %25 |
| \$ | %28 |
| + | %2B |

| Character | URL Encoded |
|-----------|-------------|
| , | %2C |
| / | %2F |
| : | %3A |
| ; | %3B |
| < | %3C |
| = | %3D |
| > | %3E |
| ? | %3F |
| @ | %40 |
| [| %5B |
|] | %5D |
| ~ | %7E |

Many modern programming languages have a URL encode and URL decoding function that automates these character replacements.

Special Characters

Because the WebSmart Services are XML-based, certain characters cannot be passed as data. They would be interpreted as part of the XML structure and would cause errors. The following codes must be substituted for these characters.

| Character | URL Encoded |
|-----------|---|
| & | & (ampersand) |
| " | " (left/right quotes should be replaced with straight quotes) |
| ' | ' (apostrophe) |
| < | < (less-than) |
| > | > (greater-than) |

2

An Introduction to Delivery Indicator

The WebSmart Delivery Indicator returns geographic, demographic and census data for a given location, defined by the address key returned by the Address Verifier service.

The Delivery Indicator Service returns a result code indicating whether the submitted address is a business or a residential address.

Adding Delivery Indicator to a Project

If you are using the SOAP service with Visual Studio .NET, you need to add a web reference to the service to your project. Click on the Project menu and select Add Web Reference... Enter the following URL on the Add Web Reference dialog box:

```
https://rbdi.melissadata.net/v2/SOAP/Service.svc
```

If you are not using Visual Studio .NET, see the documentation for your SOAP interface for the procedure for adding the service to your project.

Submitting an XML request

After building your XML string from your data, an XML request to the web service is submitted using an HTTP POST operation to the following URL:

```
https://rbdi.melissadata.net/v2/XML/Service.svc/doRBDI
```

Building a REST Request

Query strings are sent to the web service as part of the URL using an HTTP Get operation appended to following URL:

```
https://rbdi.melissadata.net/v2/REST/Service.svc/doRBDI
```

3

Delivery Indicator Request Object

At the very minimum, a request to the WebSmart Delivery Indicator consists of the user's Customer ID and at least one AddressKey.

SOAP Request

The following Visual Basic Code shows a simple order of operations for building and submitting a RequestArray object, submitting it to the web service and retrieving a response object.

Step 1 – Create the Request and Response Objects

```
Dim ReqRBDI As New dqwsRBDI.RequestArray  
Dim ResRBDI As New dqwsRBDI.ResponseArray
```

Step 2 – Assign the General Request Values

There are two properties of the Request Array object that apply to the request as a whole. CustomerID is required.

```
ReqRBDI.CustomerID = strCustID  
ReqRBDI.TransmissionReference = strTranRef
```

The Transmission Reference is a unique string value that identifies this request array.

Step 3 – Dimension the Record Array

The maximum number of records per request is 100, therefore the largest dimension will be 99.

```
ReDim ReqRBDI.Record(99)
```

For maximum efficiency, you should dimension the array using the exact number of records being submitted minus one.

Step 4 – Build the Record Array

The exact method for building the array will depend on the exact database software in use, but you will need to loop through every record to be submitted and assign the required values to the corresponding elements for each record in the RequestArray.

```
ReqRBDI.Record(intRecord) = New  
    dqwsRBDI.RequestArrayRecord  
ReqRBDI.Record(intRecord).AddressKey = "92688211282"
```

The lines above show only the elements that are absolutely required to submit a record to the web service. See the rest of this chapter for a description of all the elements available to include with a request record.

Repeat for each record being submitted with the current RequestArray.

Step 5 – Submit the Request Array

The final step is to create the Service Client Object and then submit the RequestArray object doRBDI method. This sends the data to the web service and retrieves the ResponseArray object.

```
RBDIClient = New dqwsRBDI.Service  
ResRBDI = RBDIClient.doRBDI(ReqRBDI)  
RBDIClient.Dispose()
```

XML Request

The raw XML request is built using whatever XML tools are available via your development tools and submitted to the following URL using an HTTP POST request.

```
https://RBDI.melissadata.net/v2/XML/Service.svc/doRBDI
```

Rather than an array of Record object, an XML request contains a <Record> element for each address record, up to 100.

The following XML Code contains the same request as the SOAP example about.

```
<RequestArray>
  <TransmissionReference>Web Service Test 2008/12/31
</TransmissionReference>
  <CustomerID>123456789</CustomerID>
  <Record>
    <RecordID>1</RecordID>>
    <AddressKey>92688211282</AddressKey>
  </Record>
  <Record>
    . . .
  </Record>
</RequestArray>
```

REST Request

A REST request can submit a single address record via an HTTP GET. The following example uses the same address as the SOAP and XML samples.

```
https://RBDI.melissadata.net/v2/REST/Service.svc/
doRBDI?id=12345678&t=RestTest&key=92688211282
```

The record ID element does not exist for the REST interface, since you can only submit a single record per request.

Request Elements

The following section lists the elements that set the basic options for each and identify the user to the web service.

Customer ID

This is a required string value containing the identifier number issued to the customer when signing up for Melissa Data Web Services.

Remarks

You need a customer ID to access any Melissa Data web service. If this element is not populated, the web service will return an error. To receive a customer ID, call your Melissa Data sale representative at 1-800-MELISSA.

Syntax

SOAP

```
Request.CustomerID = string
```

XML

```
<RequestArray>  
  <CustomerID>String</CustomerID>  
</RequestArray>
```

REST

```
id={CustomerID}
```

Transmission Reference

This is an optional string value that may be passed with each Request Array to serve as a unique identifier for this set of records.

Remarks

This value is returned as sent by the Response Array, allowing you to match the Response to the Request.

Syntax

SOAP

```
Request.TransmissionReference = string
```

XML

```
<RequestArray>  
  <TransmissionReference>String</TransmissionReference>  
</RequestArray>
```

REST

```
t={transMissionReference}
```

Record Elements

For the SOAP and XML web services, the Request Array will contain an element or property called Record. In SOAP this property is an array of object variables of the type Record. XML will have as many Record elements as there are addresses being submitted to the web service.

The REST interface only allows a single record per request.

Record ID

This element is a string value containing a unique identifier for the current record.

Remarks

Use this element to match the record with the record returned with the Response Array.

When using the SOAP interface, if this element is not populated, the web service will automatically insert a sequential number for each record.

There is no equivalent for Record ID for the REST interface.

Syntax

SOAP

```
Request.Record().RecordID = string
```

XML

```
<RequestArray>  
  <Record>  
    <RecordID>String</RecordID>  
  </Record>  
</RequestArray>
```

AddressKey

This element is a required eleven-character string value containing a unique number representing a specific location.

Remarks

This value is returned by submitting an address record to the Address Verifier service.

Syntax

SOAP

```
Request.Record().AddressKey = string
```

XML

```
<RequestArray>  
  <Record>  
    <AddressKey>String</AddressKey>  
  </Record>  
</RequestArray>
```

REST

```
key={AddressKey}
```

4

Delivery Indicator Response

The SOAP interface for the Delivery Indicator service returns a ResponseArray Object. The primary component of this object is an array of Record objects, one for each record submitted with the RequestArray, containing the verified and standardized address data.

The XML interface returns an XML document containing a number of <Record> elements, one for each record submitted with the Request, containing the verified and standardized address data.

The REST interface returns an XML document with a single <Record> element.

TransmissionReference

Returns a string value containing the contents of the TransmissionReference element from the original Request.

Remarks

If you passed any value to the TransmissionReference element when building your request, it is returned here. You can use this property to match the response to the request.

Syntax

SOAP

```
string = Response.TransmissionReference
```

XML

```
<ResponseArray>  
  <TransmissionReference>  
    String  
  </TransmissionReference>  
</ResponseArray>
```

Total Records

Returns a string value containing the number records returned with the current response.

Remarks

This property returns the number of records processed and returned by the response as a string value.

Syntax

SOAP

```
string = Response.TotalRecords
```

XML

```
<ResponseArray>  
  <TotalRecords>String</TotalRecords>  
</ResponseArray>
```

Results

Returns a string value containing the general and system error messages from the most recent request sent to the service.

Remarks

Do not confuse this element with the Results element returned with each record, described on page 21. This element returns error messages caused by the most recent request as a whole.

The possible values are:

| Code | Short Description | Long Description |
|------|--------------------------------|---|
| SE01 | Web Service Internal Error | The web service experienced an internal error. |
| GE01 | Empty Request Structure | The SOAP, JSON, or XML request structure is empty. |
| GE02 | Empty Request Record Structure | The SOAP, JSON, or XML request record structure is empty. |
| GE03 | Records Per Request Exceeded | The counted records sent more than the number of records allowed per request. |
| GE04 | Empty CustomerID | The CustomerID is empty. |
| GE05 | Invalid CustomerID | The CustomerID is invalid. |
| GE06 | Disabled CustomerID | The CustomerID is disabled. |
| GE07 | Invalid Request | The SOAP, JSON, or XML request is invalid. |

Syntax

SOAP

```
string = Response.Results
```

XML

```
<ResponseArray>
  <Results>String</Results>
</ResponseArray>
```

Version

Returns a string value containing the current version number of the Delivery Indicator web service.

Syntax

SOAP

```
string = Response.Version
```

XML

```
<ResponseArray>  
  <Version>String</Version>  
</ResponseArray>
```

Record Elements

The SOAP version of the Response Array returns a property called Record which is an array of Record objects, one for each record submitted with the original Request Array.

The XML service returns one <Record> element for every record submitted with the original request.

The REST response is identical to the XML response, but will only contain a single <Record> element.

The following section describes the elements returned by each record in the Response Array.

Record ID

For each record in the Response Array, this element returns a string value containing the unique identifier for the current record if one was passed to the Request Array.

Remarks

Use this element to match the record in the Response Array with the record originally passed with the request.

Syntax

SOAP

```
string = Response.Record().RecordID
```

XML

```
<ResponseArray>  
  <Record>  
    <RecordID>String</RecordID>  
  </Record>  
</ResponseArray>
```

Results

For each record in the Response Array, this element returns a string value containing either the RBDI status of the location represented by the submitted address key or an error code if the service was unable to process the submitted data.

Remarks

This element returns the status and error messages for each record in the Response Array. For the general status and error messages generated by the most recent RBDI request, see the general Result element on page 17.

The Result element may return one or more four-character strings, separated by commas, depending on the result generated by the current record.

If the address in the current record was verified, this element will contain the value “GS01” at the very minimum and may include more of the “GS” codes. If the address could not be verified, the codes beginning with “GE” will indicate the reason or reasons why verification failed.

The possible values are:

| Code | Short Description | Long Description |
|------|----------------------|---|
| DS01 | Residential Address | The processed address is a residential address. |
| DS02 | Business Address | The processed address is a business address. |
| DS03 | Unknown Address Type | The address type could not be determined for the current address. |
| DE | General Error | The AddressKey field was empty, address length was less or more than 11 characters. |

Syntax

SOAP

```
string = Response.Record().Results
```

XML

```
<ResponseArray>
  <Record>
    <Address>
      <Results>String</Results>
    </Address>
  </Record>
</ResponseArray>
```

Response Object XML Format

The following shows the structure of the XML document returned by the WebSmart Delivery Indicator service.

```
<?xml version="1.0" encoding="UTF-8"?>
<ResponseArray xmlns="urn:mdWebServiceRBDI"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  >
  <version>String</version>
  <TransmissionReference>String
</TransmissionReference>
  <Results>String</Results>
  <TotalRecords>String</TotalRecords>

  <Record>
    <RecordID>String</RecordID>
    <Address>
      <Results>String</Results>
    </Address>
  </Record>
</ResponseArray>
```