Reference Guide

WebSmart
Zip Search

MELISSA DATA®

WebSmart ZIP Search

Reference Guide

Copyright

Trademarks

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 number: WSZRFG
Revision Number: 131018.089
Last Update: October 18, 2013

Dear Developer,

I would like to take this opportunity to thank you for your interest in Melissa Data products and introduce you to the company.

Melissa Data has been a leading provider of data quality and address management solutions since 1985. Our data quality software, Cloud services, and data integration components verify, standardize, consolidate, enhance and update U.S., Canadian, and global contact data, including addresses, phone numbers, and email addresses, for improved communications and ROI. More than 5,000 companies rely on Melissa Data to gain and maintain a single, accurate and trusted view of critical information assets.

This manual will guide you through the functions 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 me at: Ray@MelissaData.com.

I look forward to hearing from you.

Best Wishes,

Raymond F. Melissa

President/CEO

Table of Contents

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. Uses multisource data to return latitude and longitude down to rooftop accuracy of over 95% of all physical addresses 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

Character	URL Encoded
$	%28
+	%2B
,	%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 ZIP Search

For both the U.S. and Canada, Web Smart ZIP Search Service will:

- Finds cities tied to a given ZIP or Postal code.
- Retrieves ZIP or Postal Codes associated with a given city and state/province.
- Find city names that match a specific pattern within a given state or province.

The ZIP Search service has three different types of searches.

Find Cities in ZIP Code

This search returns all of the matching city name records associated with a submitted ZIP/Postal Code. This may include unincorporated communities adjoining a city and alternative names and spellings for a given city.

If the OfficialCityNamesOnly option is enabled, this search will only return the record that matches the official USPS city name associated with the ZIP Code.

Find ZIP In Cities Search

This search returns all of the ZIP/Postal code records associated with a submitted combination of city name and state.

Find Cities in State Search

This search returns all of city names that match a submitted city name pattern and state/province. The city name can employ a wildcard character. For example, the search strings “Rancho S*” and “CA” will return records for all cities in California that begin with the letters “Rancho S.”

Adding ZIP Search 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://zipsearch.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://zipsearch.melissadata.net/v2/XML/Service.svc/
   doZipSearch
```

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://zipsearch.melissadata.net/v2/REST/Service.svc/
   doZipSearch
```

3 ZIP Search Request

At the minimum, a request to the WebSmart ZIP Search Service will consist of either a ZIP/Postal Code or a City and State combination, depending on the type of search selected. Only one record is allowed per request.

SOAP Request

The following Visual Basic Code shows a simple order of operations for building and submitting a Request object, submitting it to the Web Service and retrieving a response object.

Step 1 – Create the Request and Response Objects

```
Dim ReqZipSearch As New dqwsZipSearch.Request
Dim ResZipSearch As New dqwsZipSearch.Response
```

Step 2 – Assign the General Request Values

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

```
ReqZipSearch.CustomerID = strCustID
ReqZipSearch.TransmissionReference = strTranRef
ReqZipSearch.SearchType = "1"
ReqZipSearch.OptOfficialCityNameOnly = True
```

The Transmission Reference is a unique string value that identifies this request. The OptOfficialCityNameOnly option applies only to Search Type 1 (Cities in ZIP Code).

Step 3 – Build the Record

The exact method for building the array will depend on the exact database software in use, but you will need to assign the required values to the corresponding elements in the Request.

Find Cities In ZIP Code

```
ReqZipSearch.Zip = "92688"
```

Find ZIP Codes in City

```
ReqZipSearch.City = "Rancho Santa Margarita"
ReqZipSearch.State = "CA"
```

Find Cities in State

```
ReqZipSearch.City = "Rancho S*"
ReqZipSearch.State = "CA"
```

Step 4 – Submit the Request

The final step is to create the Service Client Object and then submit the Request object doZipSearch method. This sends the data to the web service and retrieves the Response object.

```
ZipSearchClient = New dqwsZipSearch.Service
ResZipSearch = ZipSearchClient.doZipSearch(ReqZipSearch)
```

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://zipsearch.melissadata.net/v2/XML/Service.svc/
   doZipSearch
```

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

Find Cities in ZIP Code

```
<Request>
   <TransmissionReference>Web Service Test 2008/12/31
   </TransmissionReference>
   <CustomerID>123456789</CustomerID>
   <SearchType>1</SearchType>
```

```
   <OptOfficialCityNameOnly>True
   </OptOfficialCityNameOnly>
   <Zip>92688</Zip>
   <Country />
</Request>
```

Find ZIP Codes in City

```
<Request>
   <TransmissionReference>Web Service Test 2010/01/31
   </TransmissionReference>
   <CustomerID>123456789</CustomerID>
   <SearchType>2</SearchType>
   <City>Rancho Santa Margarita</City>
   <State>CA</State>
</Request>
```

Find Cities in State

```
<Request>
   <TransmissionReference>Web Service Test 2010/01/31
   </TransmissionReference>
   <CustomerID>123456789</CustomerID>
   <SearchType>3</SearchType>
   <City>Rancho S*</City>
   <State>CA</State>
</Request>
```

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.

Find Cities in ZIP Code

```
https://zipsearch.melissadata.net/v2/REST/Service.svc/
   doZipSearch?id=12345678&opt=true&stype=1&zip=92688
```

Find ZIP Codes in City

```
https://zipsearch.melissadata.net/v2/REST/Service.svc/
   doZipSearch?id=12345678&stype=2&city=Rancho%20Santa%20
   Margarita&state=CA
```

Find Cities in State

```
https://zipsearch.melissadata.net/v2/REST/Service.svc/
   doZipSearch?id=12345678&stype=3&city=Rancho%20S%2A&sta
   te=CA
```

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

```
<Request>
   <CustomerID>String</CustomerID>
</Request>
```

REST

```
id={CustomerID}
```

Transmission Reference

This is an optional string value that may be passed with each Request 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

```
<Request>
    <TransmissionReference>String</TransmissionReference>
</Request>
```

REST

```
t={transmissionReference}
```

SearchType

This element selects which search method the ZIP Search web service will use with the current request.

Remarks

The valid values are:

“1” – Find Cities in ZIP Code (Default)

“2” – Find ZIP Codes in City

“3” – Find City Names in State

Any values not matching one of these values will be ignored and the Default value of “1” will be used.

Syntax

SOAP

```
Request.SearchType = string
```

XML

```
<Request>
   <SearchType>String</SearchType>
</Request>
```

REST

```
stype={SearchType}
```

OptOfficialCityNameOnly

This Boolean value instructs the web service to return all cities associated with a ZIP/Postal Code during a Find Cities in ZIP Code Search or to return just the official city.

Remarks

This value will be ignored during a Find ZIP Codes in City Search and Find City Names in State Search.

True – Will only return one search record, the official city.

False – Will return all cities found in the input ZIP/Postal code (Default)

If a invalid value is entered or this element is omitted altogether, the Default value of False will be used.

Syntax

SOAP

```
Request.OptOfficialCityNameOnly = string
```

XML

```
<Request>
   <OptOfficialCityNameOnly>String</OptOfficialCityNameOnly>
</Request>
```

REST

```
opt={OptOfficialCityNameOnly}
```

City

This element passes the city name to the ZIP Search service for the ZIP/ Postal Codes in City and Cities in State Search Types.

Remarks

The City Element is required for both the ZIP/Postal Codes in City and Cities in State Search Types.

For the cities in State search, the City element can contain a trailing wildcard character — “*” — for multiple matches. Note that the asterisk character must be encoded (“%2A”) when using with the REST interface.

Syntax

SOAP

```
Request.City = string
```

XML

```
<Request>
    <City>String</City>
</Request>
```

REST

```
city={City}
```

State

This element passes the state abbreviation to the ZIP Search service for the ZIP/Postal Codes in City and Cities in State.

Remarks

The State Element is required for both the ZIP/Postal Codes in City and Cities in State Search Types.

Syntax

SOAP

```
Request.State = string
```

XML

```
<Request>
    <State>String</State>
</Request>
```

REST

```
state={State}
```

Zip

This element passes a five-digit ZIP Code or a six-digit Postal Code to the ZIP Search service for the Cities in ZIP Code Search Type.

Remarks

The ZIP element is required for the Cities in ZIP Code Search Type.

Syntax

SOAP

```
Request.Zip = string
```

XML

```
<Request>
   <Zip>String</Zip>
</Request>
```

REST

```
zip={Zip}
```

Country

This element passes a two-character country code to the ZIP Search service.

Remarks

Currently, ZIP Search only works on addresses within the United States and Canada, so this element is optional and has no effect. It is included for future compatibility.

Syntax

SOAP

```
Request.Country = string
```

XML

```
<Request>
    <Country>String</Country>
</Request>
```

REST

```
ctry={Country}
```

4 Zip Search Response

The SOAP interface for the Zip Search service returns a Response Object. The primary component of this object is a record array of ZIP/Postal Code and City information, one for each ZIP record returned.

The XML and REST interfaces return XML documents containing a number of <ZipRecord> elements, one for each zip record returned.

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 and REST

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

Total Records

Returns a string value containing the number of ZIP/Postal Code or City records returned with the current response.

Remarks

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

Syntax

SOAP

```
string = Response.TotalRecords
```

XML and REST

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

Results

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

Remarks

The following codes can be returned:

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.
ZS01	Successful Cities in Zip Code Search	The search for Cities in Zip Code was successful.
ZS02	Successful Zip Code in City Search	The search for Zip Code in City was successful.
ZS03	Successful Cities in State Search	The search for Cities in State was successful.
DE01	Missing or Invalid Input	The required input for the service was not met or was in an invalid format.
DE02	No Records Found	There was a search error: No records were found.

Syntax

SOAP

```
string = Response.Results
```

XML and REST

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

Version

Returns a string value containing the current version number of the Zip Search service.

Syntax

SOAP

```
string = Response.Version
```

XML and REST

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

ZipRecord Elements

The SOAP version of the Response object returns a property called ZipRecord which is an array of ZipRecord objects matching the input submitted with the Request with the original.

The XML and REST services may return one or more <ZipRecord> elements that match the original request.

The maximum number of records returned in both cases in 1,000.

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

Record ID

For each record in the Response, this element returns a string value containing the sequential identifier for the current ZipRecord.

Remarks

The RecordID is sequential number to aid in organizing the ZipRecords returned by the Zip Search service.

Syntax

SOAP

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

XML and REST

```
<Response>
   <ZipRecord>
      <RecordID>String</RecordID>
   </ZipRecord>
</Response>
```

Zip

This element returns a string value with the zip code for the current ZIP record.

Remarks

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().Zip
```

XML and REST

```
<Response>
   <ZipRecord>
      <Zip>String</Zip>
   </ZipRecord>
</Response>
```

ZipType

This element returns a one-character string value with the code for the ZIP Code type for the current ZIP record.

Remarks

The possible values are:

Code	Explanation
P	A ZIP Code used only for PO Boxes.
U	Unique: A ZIP Code assigned to an organization or government institution such as the IRS.
M	Military: A ZIP Code assigned to an APO/FPO.
Empty	A standard ZIP Code or a Canadian Postal Code.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().ZipType
```

XML and REST

```
<Response>
   <ZipRecord>
      <ZipType>String</ZipType>
   </ZipRecord>
</Response>
```

City

These elements return string values with the city name and city abbreviation for the current ZIP record.

Remarks

If the return value of the GetCity function is longer than 13 letters, the GetCityAbbreviation function will return the official abbreviation the post office has associated with that city or municipality name. For example, for “Rancho Santa Margarita,” the CityAbbreviation function will return the abbreviation “Rcho Sta Marg.”

If the return value of the GetCity function is 13 letters long or shorter, the GetCityAbbreviation function will return the full city or municipality name.

This element is returned by all three search types.

Syntax

SOAP

```
string = Response.ZipRecord().City.Name
string = Response.ZipRecord().City.Abbreviation
```

XML and REST

```
<Response>
   <ZipRecord>
      <City>
         <Name>String</Name>
         <Abbreviation>String</Abbreviation>
      </City>
   </ZipRecord>
</Response>
```

County

These elements return string values with the county name and five-digit county FIPS code of the current ZIP record

Remarks

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

The FIPS Code is only returned for U.S. records.

Syntax

SOAP

```
string = Response.ZipRecord().County.Name
string = Response.ZipRecord().County.FIPS
```

XML and REST

```
<Response>
   <ZipRecord>
      <County>
         <Name>String</Name>
         <FIPS>String</FIPS>
      </County>
   </ZipRecord>
</Response>
```

State

This element returns a string value with the two-letter state abbreviation of the current ZIP record

Remarks

This element is returned by all three search types.

Syntax

SOAP

```
string = Response.ZipRecord().State
```

XML and REST

```
<Response>
   <ZipRecord>
      <State>String</State>
   </ZipRecord>
</Response>
```

Area Code

This element returns a three-character string value with the area code of the current ZIP record.

Remarks

If a ZIP/Postal Code has more than one area code assigned to it, the dominant area code will be returned.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().AreaCode
```

XML and REST

```
<Response>
   <ZipRecord>
      <AreaCode>String</AreaCode>
   </ZipRecord>
</Response>
```

Time Zone

These elements return string values containing the time zone name and time zone code of the current zip record.

Remarks

Possible values are:

Code	Zone	Code	Zone
0	Military (APO or FPO)	9	Alaska Time
4	Atlantic Time	10	Hawaii Time
5	Eastern Time	11	Samoa Time
6	Central Time	12	Marshall Island Time
7	Mountain Time	14	Guam Time
8	Pacific Time	15	Palau Time

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().TimeZone.Code
string = Response.ZipRecord().TimeZone.Name
```

XML and REST

```
<Response>
   <ZipRecord>
      <Latitude>
         <Code>String</Code>
         <Name>String</Name>
      </Latitude>
   </ZipRecord>
</Response>
```

Latitude

This element returns a string value with the latitude of the current ZIP record.

Remarks

The latitude is a 9 character number accurate to 6 decimal places.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().Latitude
```

XML and REST

```
<Response>
   <ZipRecord>
      <Latitude>String</Latitude>
   </ZipRecord>
</Response>
```

Longitude

This element returns a string value with the longitude of the current ZIP record.

Remarks

The longitude is a 9 character number accurate to 6 decimal places.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().Longitude
```

XML and REST

```
<Response>
   <ZipRecord>
      <Longitude>String</Longitude>
   </ZipRecord>
</Response>
```

LastLineIndicator

This element returns a one character string value with the LastLineIndicator of the current ZIP record.

Remarks

An “L” in this element indicates that the city name is the official U. S. Postal Service name for the ZIP Code (Only one record per ZIP Code is coded with an “L”).

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

This property does not return a value for Canadian records.

Syntax

SOAP

```
string = Response.ZipRecord().LastLineIndicator
```

XML and REST

```
<Response>
   <ZipRecord>
      <LastLineIndicator>String</LastLineIndicator>
   </ZipRecord>
</Response>
```

LastLineNumber

This element returns a string value with the LastLineNumber of the current ZIP record.

Remarks

The LastLineNumber is a 6 digit number used to break ties on certain records using city names.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

This property does not return a value for Canadian records.

Syntax

SOAP

```
string = Response.ZipRecord().LastLineNumber
```

XML and REST

```
<Response>
   <ZipRecord>
      <LastLineNumber>String</LastLineNumber>
   </ZipRecord>
</Response>
```

PreferredLastLineNumber

This element returns a string value with the Preferred LastLine Number of the current zip record.

Remarks

The PreferredLastLineNumber property is a 6-character string value containing the preferred last line number for the city name.

This is similar to the “LastLineNumber” element mentioned earlier. If the preferred last line number is the same as the last line number, this city name is the one the Post Office will recognize as the official name for that ZIP Code.

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

Syntax

SOAP

```
string = Response.ZipRecord().PreferredLastLineNumber
```

XML and REST

```
<Response>
   <ZipRecord>
      <PreferredLastLineNumber>String
         </PreferredLastLineNumber>
   </ZipRecord>
</Response>
```

FacilityCode

This element returns a string value with the facility code of the current ZIP record.

Remarks

This element returns values for the Find Cities in ZIP Code and Find ZIP Codes in City search types. It returns a blank for Find City Names in State.

This property does not return a value for Canadian records.

Code	Type	Code	Type
A	Airport Mail Facility	K	Bulk Mail Facility
B	Branch	M	Money Order Unit
C	Community Post Office	N	Community or Place name
D	Area Distribution Center	P	Post Office
E	Sectional Center Facility	S	Station
F	Delivery Distribution Center	U	Urbanization (Used in Puerto Rico)
G	General Mail Facility	X	Vanity name (Should not be used)

Syntax

SOAP

```
string = Response.ZipRecord().FacilityCode
```

XML and REST

```
<Response>
   <ZipRecord>
      <FacilityCode>String</FacilityCode>
   </ZipRecord>
</Response>
```

Response Object XML Format

The following shows the structure of the XML document returned by the WebSmart ZipSearch Service.

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
   <Version>String</Version>
   <TransmissionReference>String</TransmissionReference>
   <Results>String</Results>
   <TotalRecords>String</TotalRecords>
   <ZipRecord>
      <RecordID>String</RecordID>
      <Zip>String</Zip>
      <ZipType>String</ZipType>
      <City>
         <Name>String</Name>
         <Abbreviation>String</Abbreviation>
      </City>
      <County>
         <Name>String</Name>
         <FIPS>String</FIPS>
      </County>
      <State>String</State>
      <AreaCode>String</AreaCode>
      <TimeZone>
         <Name>String</Name>
         <Code>String</Code>
      </TimeZone>
      <Latitude>String</Latitude>
      <Longitude>String</Longitude>
      <LastLineIndicator>String</LastLineIndicator>
      <LastLineNumber>String</LastLineNumber>
      <PreferredLastLineNumber>String</
   PreferredLastLineNumber>
      <FacilityCode>String</FacilityCode>
   </ZipRecord>
</Response>
```