pitney bowes

Location Intelligence

Spectrum™ Technology Platform

Version 2019.1.0

Global Geocoding REST Web Services Guide

Table of Contents

1 - Using the Global Geocoding REST API

In this section

Introduction to the Global Geocoding APIs

The Global Geocoding REST API allows you to develop and deploy geocoding desktop, mobile or Web applications that are capable of delivering location information for over 250 countries and territories.

This guide contains information on using the Global Geocoding REST API which provides the following web services:

- **Geocode Service**: performs forward geocoding using input addresses and returning location data and other information.
- **Reverse Geocode Service**: performs reverse geocoding using input coordinates and returns address information that is the best match for that point.
- **Interactive Geocode Service**: suggests addresses and place names as you type.
- **Key Lookup Service**: returns geocoded candidates when given a unique key. It is a more efficient method than matching with an address, as the key is unique to that address. Global Geocoding Module supports the pbKey™ unique identifier for US data and the G-NAF key for AUS data.

Each service has options that allow you to control matching and geocoding criteria, dataset resource configuration and more.

Getting Started

Review one of these topics for next steps:

- **Getting Started with the REST API** on page 6
- **Getting Started with the Java API**

2 - REST Web Services

In this section

Introduction to Global Geocoding Services

The Global Geocoding REST API provides the following services:

- **`Geocode`**—Takes a single input address or multiple input addresses and returns standardized US or international address and geocoding information.
- **`Interactive`**—Takes a partial address and other address elements to restrict the search area and return match candidates. Interactive data is used to match against the input.
- **`KeyLookup`**—Takes a key and key type to geocode an address and return additional information. The key is a unique identifier to that address.
- **`ReverseGeocode`**—Takes a single input latitude and longitude coordinates or multiple input coordinates and returns address information for the location(s).
- **`Capabilities`**—Returns the capabilities of the geocode service, such as the supported operations, the available country geocoding engines and the country-specific custom fields.
- **`Dictionaries`**—Returns information about the installed address dictionaries.

Related Topics

Getting Started with the REST API on page 6

Getting Started with the REST API

Requirements

- Java 8
- GGS distribution (ggs-dist-{version}.zip)
  - Once unzipped, you'll need the GGS SDK CLI located in "{directory}/ggs-dist-{version}/cli"
- Dataset(s) in the Spectrum data format (*.spd)
- Server capable of deploying and running a .war file (i.e., Tomcat)

Placeholders

The following conventions are used as placeholders for some common items:

- {directory} Any directory where you would like to install the resources for the SDK.
- {version} The version of the GGS SDK such as **3.0.0.**
- {spd} Dataset in the Spectrum data format. For example, **KGD082019.spd**.
- {tomcat} - installed Tomcat location. For example, **apache-tomcat-8.5.30**.

Configure using the GGS CLI

1. Create a location where datasets will be installed.
   a. Windows:

```
mkdir {directory}/data
```

   b. Linux:

```
mkdir -p {directory}/data
```

2. Run the GGS CLI. The command below starts the CLI in interactive mode.

   Note: The GGS CLI can be run in interactive or batch mode. [About the **CLI Batch Process**.]

   a. Windows:

```
cd {directory}/ggs-dist-{version}/cli
cli.cmd
```

   b. Linux:

```
cd {directory}/ggs-dist-{version}/cli
cli.sh
```

3. Locate .spd file(s) and extract the dataset(s). [About the **Extract command**.]

```
extract --s "{spd}" --d "{directory}/data"
```

4. Configure data for the SDK, based on the dataset location, by running this command from within the interactive CLI. [About the **Configure command**.]

```
configure --s "{directory}/data" --d
"{directory}/ggs-dist-{version}/resources/config"
```

Deploy REST API to Tomcat

Note: The `{tomcat}/webapps/Geocode` directory must be empty before proceeding with this command. [About the **Deploy command**.]

```
deploy --c "/ggs-dist-{version}/resources" --m "WAR_EXTRACTED"
  --d "{tomcat}/webapps/Geocode" --l "{tomcat}/logs"
```

Start the Server

1. From the tomcat directory `cd {tomcat}/bin`

a. Windows:

```
startup.bat
```

b. Linux:

```
./startup.sh
```

Sample Application

As you explore the REST services, you may find it helpful to use the **Global Geocoding API Sample**. This is an interactive web application that demonstrates the geocoding API. The URL is relative to any context of the server, depending on where the war is deployed. URL: **http://{server}:{port}/{:context}/sample/index.html**

For example, to see typeahead functionality, click the **Interactive Geocode** tab in the sample application.

> **Note:** Prior to using the sample application, if you haven't completed the steps above, you must first install and configure your geocoding datasets using the GGS CLI.

Additional Information

- **REST Web Services**

Making Requests using HTTP

WADL URL

The `WADL` for the Global Geocoding REST API web services is:

```
http://<server>:<port>/rest/GlobalGeocode/?_wadl
```

Supported Payload Formats

The supported message payload formats for the requests and responses are `JSON` and `XML`. The message exchange format is negotiated between the client and the service via information specified in the `HTTP` headers.

HTTP Headers

To negotiate the content type being sent between the client and service, the request includes an `Accept` header to indicate the acceptable media type. Optionally, it can also indicate the `MIME Content-Type` being sent in the request.

The response from the server will return a status code and the `Content-Type` of the response.

The following are example `HTTP` content negotiation headers for `JSON` and `XML`:

JSON

```
Accept: application/json; charset=utf-8
Content-Type: application/json; charset=utf-8
```

XML

```
Accept: application/xml; charset=utf-8
Content-Type: application/xml; charset=utf-8
```

The following table defines the type of response to expect based on the header information specified in the request.

Request	Header Information	Response Content Type
service_name.json	No special header information.	json
service_name.json	Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8	xml
service_name.json	Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8	json
service_name	Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8	json
service_name	Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8	xml
service_name	No special header information.	json

Request	Header Information	Response Content Type
service_name.xml	Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8	json
service_name.xml	Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8	xml
service_name.xml	No special header information.	xml

Supported HTTP Methods

A complete REST request is formed by combining an HTTP method with the full URI to the service you are addressing.

To create a complete request, combine the operation with the appropriate **HTTP headers** and any required **payload**.

Each Global Geocoding service (**Geocode**, **Reverse Geocode**, **Interactive Geocode**, **Key Lookup**, **Capabilities**, **Dictionaries**) supports `GET` and `POST` requests. A `GET` request uses a subset of the preferences while a `POST` request can specify the complete set.

HTTP Status Codes

Each response to a request contains an HTTP status code. The HTTP status code reports on the outcome of the HTTP request to a service. The following table provides the most common status codes that are returned by the services.

Status Code	Short Description	Description
200	OK	The request is successful. Typically returned by a `GET` or a `POST` returning information.
400	Bad Request	The request contained an error. This status is returned by various methods when the data provided by the client - either as part of the URI, query parameters or the body - does not meet the server requirements.
404	Not Found	The requested resource was not found.
405	Method Not Allowed	The method requested is not allowed for the resource in the URI.
406	Not Acceptable	The requested media type specified in the Accept header is not supported. The supported media types include `application/JSON` and `application/xml`.
500	Internal Server Error	An internal error was encountered that prevents the server from processing the request and providing a valid response.

Geocoding Requests

The POST request enables you to submit a single input address or a list of addresses for batch processing. Matching and/or geocoding preferences can optionally be specified to the Geocode service and receive the associated latitude/longitude coordinates and location information. The preference options for a POST request are the complete set of available options.

The GET request enables you to submit an input address and matching and/or geocoding preferences to the Geocode service and receive a response that provides the candidates object which contains the associated latitude/longitude coordinates and other matching and location information about each candidate. The preferences for a GET request are a subset of the total available with the POST request. Each key/value pair is separated by an ampersand (&).

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/geocode[.content type]
```

For supported parameters for the Geocode Service see **Request Fields**, **Preferences**, and **Output Fields**.

Geocode Service Request

Geocode GET Request

The GET request enables you to submit an input address and matching and/or geocoding preferences to the Geocode service and receive a response that provides the candidates object which contains the associated latitude/longitude coordinates and other matching and location information about each candidate. The preferences for a GET request are a subset of the total available with the POST request. Each key/value pair is separated by an ampersand (&).

Base URI

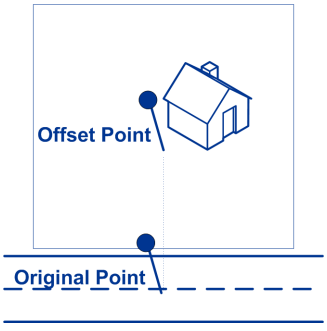
```
http://<server>:<port>/rest/GlobalGeocode/geocode[.content type]
```

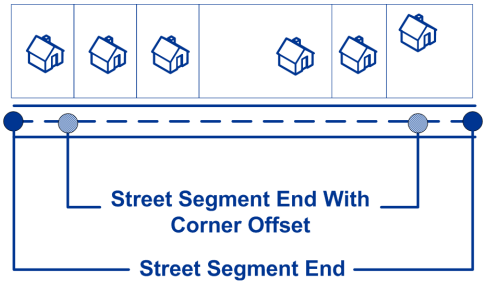
Query Parameters

The following table defines the GET query parameters for the Geocode service. For information on the response, see **GeocodeServiceResponse Object** on page 23.

Parameter	Type	Description
POST: mainAddressLine **GET**: mainAddress	String	**Single Line input**—If no other field is populated, then the `mainAddress` entry will be treated as a single line input and can be a collection of address field elements. The input order of the address fields should reflect the normal address formatting for your country. Optional. For example: **4750 Walnut St., Boulder CO, 80301** **Multiline Input** If the address fields (placeName, lastLine, postalCode, etc.) are provided separately, then the content of this field will be treated as the street address part and can include company name, house number, building names and street names. Optional. **Street Intersection Input**—To enter an intersection, specify the two street names separated by a double ampersand (&&).
country	String	ISO 3166-1 alpha-3 country code. Required. For country codes, see **ISO 3166-1 Country Codes** on page 209.
areaName1	String	Specifies the largest geographic area, typically a state or province. Optional.
areaName2	String	Specifies the secondary geographic area, typically a county or district. Optional.
areaName3	String	Specifies a city or town name. Optional.
areaName4	String	Specifies a city subdivision or locality. Optional.
POST: postCode1 **GET**: postalCode	String	The postal code in the appropriate format for the country. Optional.
POST: postCode2	String	The postal code in the appropriate format for the country. Optional.
placeName	String	Building name, place name, Point of Interest (POI), company or firm name associated with the input address. Optional. For example: **Pitney Bowes** 4750 Walnut St. Boulder, CO 80301

Parameter	Type	Description
POST: addressLastLine **GET**: lastLine	String	The last line of the address. Optional.
matchMode	String	Match modes determine the leniency used to make a match between the input address and the reference data. Select a match mode based on the quality of your input and your desired output. The following match modes are available: **EXACT** Requires a very tight match. This restrictive mode generates the fewest match candidates, which decreases the processing time. **STANDARD** Requires a close match and generates a moderate number of match candidates. Default. **RELAXED** Allows a loose match and generates the most match candidates, which increases the processing time and results in more multiple matches. **CUSTOM** Provides the capability for you to define the matching criteria by setting `MustMatch` fields; however, you can only set the `MustMatch` fields using a `POST` request. For a `GET` request, the `MustMatch` default values are used.
fallbackGeo	Boolean	Specifies whether to attempt to determine a geographic region centroid when an address-level geocode cannot be determined. Optional. **true** Return a geographic centroid when an address-level centroid cannot be determined. Default. **false** Do not return a geographic centroid when an address-level centroid cannot be determined.
fallbackPostal	Boolean	Specifies whether to attempt to determine a post code centroid when an address-level geocode cannot be determined. Optional. **true** Return a post code centroid when an address-level centroid cannot be determined. Default. **false** Do not return a post code centroid when an address-level centroid cannot be determined.
maxCands	Integer	The maximum number of candidates to return. Optional. Must be an integer value. Default = 1.

Parameter	Type	Description
maxRanges	Integer	A range is a series of addresses along a street segment. For example, 5400-5499 Main St. irepresents an address range in the 5400 block of Main St. A range may represent just odd or even addresses within a segment, or both. A range may also represent a single building with multiple units, such as an apartment building. This option specifies the maximum number of ranges to return for each candidate. Since the geocoder returns one candidate per segment, and since a segment may contain multiple ranges, this option allows you to see the other ranges in a candidate's segment. Must be an integer value. Default = 0.
maxRangeUnits	Integer	This option specifies the maximum number of units (for example, apartments or suites) to return for each range. For example, if you were to geocode an office building at 65 Main St. containing four suites, there would be a maximum of four units returned for the building's range: 65 Suite 1, 65 Suite 2, 65 Suite 3, and 65 Suite 4. If you were to specify a maximum number of units as 2, then only two units would be returned instead of all four. Must be an integer value. Default = 0.
streetOffset	Double	The offset distance from the street segments. The distance is in the units you specify in the `streetOffsetUnits` preference. Default value = 7 meters. The offset distance is used in street-level geocoding to prevent the geocode from being in the middle of a street. It compensates for the fact that street-level geocoding returns a latitude and longitude point in the center of the street where the address is located. For example, an offset of 50 feet means that the geocode will represent a point 50 feet back from the center of the street. The distance is calculated perpendicular to the portion of the street segment for the address. Offset is also used to prevent addresses across the street from each other from being given the same point. 

Parameter	Type	Description
streetOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).
cornerOffset	Double	Distance to offset the street end points in street-level matching. The distance is in the units you specify in the `cornerOffsetUnits` preference. This value is used to prevent addresses at street corners from being given the same geocode as the intersection. Defines the offset position of the geocoded point with respect to the corner. Default value = 7 meters. The following diagram compares the end points of a street to offset end points. 
cornerOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).

Geocode POST Request

The `POST` request enables you to submit a single input address or a list of addresses for batch processing. Matching and/or geocoding preferences can optionally be specified to the `Geocode` service and receive the associated latitude/longitude coordinates and location information. The preference options for a `POST` request are the complete set of available options.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/geocode[.content type]
```

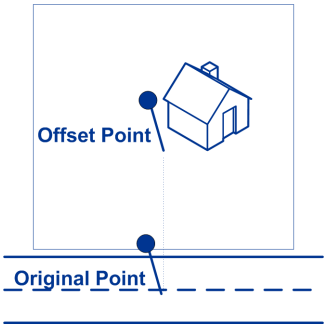
Request Parameters

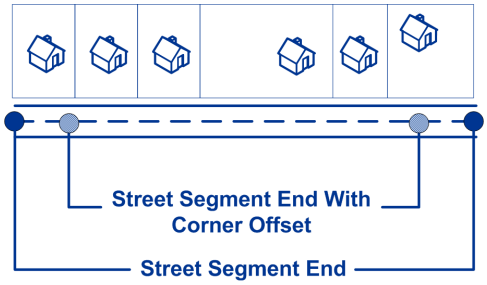
The `POST` request comprises the following input parameters:

- `addresses` - The address or addresses to be geocoded. The addresses array of Address objects. The addresses array may contain one or more input addresses. Required.
- `type` - The type of geocode. Optional. The type parameter is optional.
- `preferences` - The matching and geocoding options. Optional.
- `mustMatchMode` - The match criteria for determining match candidates Optional.
- `returnFieldsDescriptor` - Controls the return of additional data on a candidate. Optional.

These objects and their elements are defined in the following table.

Parameter	Type	Description
POST: type **GET**: geocodeType	String	Indicates the geocode type to be performed. Optional. **ADDRESS** Geocode to a street address. Default. **GEOGRAPHIC** Geocode to the geographic centroid of a city or state. **POSTAL** Geocode to a postal code.
POST: returnAllCandidateInfo	Boolean	Specifies whether to return all available information for each candidate. **true** Return all available information for each candidate. **false** Do not return all available information for each candidate. Default.

Parameter	Type	Description
POST: fallbackToGeographic **GET**: fallbackGeo	Boolean	Specifies whether to attempt to determine a geographic region centroid when an address-level geocode cannot be determined. Optional. **true** Return a geographic centroid when an address-level centroid cannot be determined. Default. **false** Do not return a geographic centroid when an address-level centroid cannot be determined.
POST: fallbackToPostal **GET**: fallbackPostal	Boolean	Specifies whether to attempt to determine a post code centroid when an address-level geocode cannot be determined. Optional. **true** Return a post code centroid when an address-level centroid cannot be determined. Default. **false** Do not return a post code centroid when an address-level centroid cannot be determined.
POST: maxReturnedCandidates **GET**: maxCands	Integer	The maximum number of candidates to return. Optional. Must be an integer value. Default = 1.
streetOffset	Double	The offset distance from the street segments. The distance is in the units you specify in the `streetOffsetUnits` preference. Default value = 7 meters. The offset distance is used in street-level geocoding to prevent the geocode from being in the middle of a street. It compensates for the fact that street-level geocoding returns a latitude and longitude point in the center of the street where the address is located. For example, an offset of 50 feet means that the geocode will represent a point 50 feet back from the center of the street. The distance is calculated perpendicular to the portion of the street segment for the address. Offset is also used to prevent addresses across the street from each other from being given the same point. 

Parameter	Type	Description
streetOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).
cornerOffset	Double	Distance to offset the street end points in street-level matching. The distance is in the units you specify in the `cornerOffsetUnits` preference. This value is used to prevent addresses at street corners from being given the same geocode as the intersection. Defines the offset position of the geocoded point with respect to the corner. Default value = 7 meters. The following diagram compares the end points of a street to offset end points. 
cornerOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).
matchMode	String	Match modes determine the leniency used to make a match between the input address and the reference data. Select a match mode based on the quality of your input and your desired output. The following match modes are available: **EXACT** Requires a very tight match. This restrictive mode generates the fewest match candidates, which decreases the processing time. **STANDARD** Requires a close match and generates a moderate number of match candidates. Default. **RELAXED** Allows a loose match and generates the most match candidates, which increases the processing time and results in more multiple matches. **CUSTOM** Provides the capability for you to define the matching criteria by setting `MustMatch` fields; however, you can only set the `MustMatch` fields using a `POST` request. For a `GET` request, the `MustMatch` default values are used.

Parameter	Type	Description
maxRanges	Integer	A range is a series of addresses along a street segment. For example, 5400-5499 Main St. is an address range representing addresses in the 5400 block of Main St. A range may represent just odd or even addresses within a segment, or both odd and even addresses. A range may also represent a single building with multiple units, such as an apartment building. This option specifies the maximum number of ranges to return for each candidate. Since the geocoder returns one candidate per segment, and since a segment may contain multiple ranges, this option allows you to see the other ranges in a candidate's segment. Must be an integer value. Default = 0.
maxRangeUnits	Integer	This option specifies the maximum number of units (for example, apartments or suites) to return for each range. For example, if you were to geocode an office building at 65 Main St. containing four suites, there would be a maximum of four units returned for the building's range: 65 Suite 1, 65 Suite 2, 65 Suite 3, and 65 Suite 4. If you were to specify a maximum number of units as 2, then only two units would be returned instead of all four. Must be an integer value. Default = 0.
POST: clientCoordSysName	String	Specifies the coordinate system that you want to convert the geometry to. The format must be the European Petroleum Survey Group (EPSG) code or the SRID code. Default = `EPSG:4326`. Specify the coordinate reference system in the format `codespace:code`.
POST: matchOnAddressNumber	Boolean	**true** A match must be made to the input address number. **false** A match does not need to be made to the input address number. Default.
POST: matchOnPostCode1	Boolean	**true** A match must be made to the input `PostCode1` field. **false** A match does not need to be made to the input `PostCode1` field. Default.

Parameter	Type	Description	
POST: matchOnPostCode2	Boolean	**true**	A match must be made to the input `PostCode2` field.
		false	A match does not need to be made to the input `PostCode2` field. Default.
POST: matchOnAreaName1	Boolean	**true**	A match must be made to the input `AreaName1` field.
		false	A match does not need to be made to the input `AreaName1` field. Default.
POST: matchOnAreaName2	Boolean	**true**	A match must be made to the input `AreaName2` field.
		false	A match does not need to be made to the input `AreaName2` field. Default.
		Note: This option is not supported by USA.	
POST: matchOnAreaName3	Boolean	**true**	A match must be made to the input `AreaName3` field.
		false	A match does not need to be made to the input `AreaName3` field. Default.
POST: matchOnAreaName4	Boolean	**true**	A match must be made to the input `AreaName4` field.
		false	A match does not need to be made to the input `AreaName4` field. Default.
POST: matchOnAllStreetFields	Boolean	**true**	A match must be made to the input street name, type and directional fields.
		false	A match does not need to be made to the input street name, type and directional fields. Default.

Parameter	Type	Description
POST: returnAllCustomFields	Boolean	**true** Return all of the custom fields for the candidate. **false** Return only the standard set of fields for the candidate. Default.
POST: returnedCustomFieldKeys	List<String>	Specifies a list of keys that represent the custom fields to be returned in the candidate's `customFields` output. To specify multiple key/value pairs for a country, use spaces to separate the names of the custom fields to be returned. Custom fields vary by country. For example: "`CTYST_KEY`" or "`DATATYPE`". Default: empty.
POST: returnMatchDescriptor	Boolean	**true** Return the match descriptor object, which indicates the parts of the candidate that matched the input address. **false** Do not return the match descriptor object. Default.
POST: returnStreetAddressFields	Boolean	**true** Return all of the individual street fields that make up the `formattedStreetAddress` field separately, as follows: • `MAIN_ADDRESS` • `THOROUGHFARE_TYPE` • `ADDRESS_ID` • `PRE_ADDRESS` • `POST_ADDRESS` • `PRE_DIRECTIONAL` • `POST_DIRECTIONAL` **false** Do not return the individual street fields separately; return these values in the `formattedStreetAddress` field. Default.
POST: returnUnitInformation	Boolean	**true** Where available, return unit type and unit value information separately in the `unitType` and `unitValue` fields, as well as in the `formattedStreetAddress` field. **false** Where available, return unit type and unit value information only in the `formattedStreetAddress` field. Default.

Geocode Service Response

GeocodeServiceResponse Object

A request to the `Geocode` service returns a `GeocodeServiceResponse` object that contains:

- `totalPossibleCandidates`— the total number of possible candidates.
- `totalMatches`— the total number of matches.
- `candidates` — lists one or more candidates that matched to your input address/addresses. Matching and location information is returned for each match candidate.

Name	Type	Description
totalPossibleCandidates	Integer	Indicates the total number of possible candidates.
totalMatches	Integer	Indicates the total number of matches.
`candidates` object of type `Candidate`, consisting of an array with one or more match candidates and associated address, matching and location information. Contains the following elements:		

Name	Type	Description
precisionLevel	Integer	A code describing the precision of the geocode. One of the following: **0** No coordinate information is available for this candidate address. **1** Interpolated street address. **2** Street segment midpoint. **3** Postal code 1 centroid. **4** Partial postal code 2 centroid. **5** Postal code 2 centroid. **6** Intersection. **7** Point of interest. (If database contains POI data.) **8** State/province centroid. **9** County centroid. **10** City centroid. **11** Locality centroid. **12-15** Reserved for unspecified custom items. **16** The result is an address point. **17** The result was generated by using address point data to modify the candidate's segment data. **18** The result is an address point that was projected using the centerline offset feature. You must have both a point and a street range database to use the centerline offset feature. **Note:** This field is not returned for USA. For geocode precision information for USA, see **Location Codes** on page 182.
formattedStreetAddress	String	The formatted main address line.
formattedLocationAddress	String	The formatted last address line.
identifier	String	For street- or point-level candidates, this is usually the segment ID.

Name	Type	Description
precisionCode	String	

Name	Type	Description
		A code describing the precision of the geocode. The format of the geocode result string is `match_category[additional_match_information]`. The possible match categories are as follows: **Z1** Postal match with post code 1 centroid. **Z2** Postal match with partial post code 2 centroid. **Z3** Postal match with post code 2 centroid. **G1** Geographic match with area name 1 centroid. **G2** Geographic match with area name 2 centroid. **G3** Geographic match with area name 3 centroid. **G4** Geographic match with area name 4 centroid. The matches in the 'S' category indicate that the record was matched to a single address candidate. **SX** Point located at a street intersection. **SC** Match point located at the house level that has been projected from the nearest segment. **S0** No coordinates are available, but parts of the address may have matched the source data. **S4** The geocode is located at a street centroid **S5** The geocode is located at a street address. **S7** The geocode is located at a street address that has been interpolated between point house locations. **S8** Match point located at the house location. Additional match information is of the format `HPNTSCSZA`. If a match result was not made for the specified component, a dash (-) will appear in place of a letter **H** House number. **P** Street prefix direction. **N** Street name. **T** Street type. **S** Street suffix direction. **C** City name. **Z** Post code. **A** Geocoding dataset.

Name	Type	Description
		U Custom user dataset. **Note:** For more detailed information including country-specific meanings and values, see **Global Result Codes** on page 198.
sourceDictionary	String	Identifies the dictionary that is the source for the candidate information and data. The source dictionary is a 0-based integer value that indicates which configured dictionary the candidate came from. If you only have a single dictionary this will always be "0".
`matching` object. Indicates what parts of the input matched; consisting of the following elements:		
matchOnAddressNumber	Boolean	Indicates if the input address number matched the candidate's address number. **True** The input address number matched the candidate's address number. **False** The input address number did not match the candidate's address number.
matchOnPostCode1	Boolean	Indicates if the input `postCode1` field matched the candidate's `postCode1` field. **True** The input `postCode1` matched the candidate's `postCode1`. **False** The input `postCode1` did not match the candidate's `postCode1`.
matchOnPostCode2	Boolean	Indicates if the input `postCode2` field (post code extension) matched the candidate's `postCode2` field. **True** The input postCode2 matched the candidate's postCode2. **False** The input `postCode2` did not match candidate's postCode2
matchOnAreaName1	Boolean	Indicates if the input `areaName1` field matched the candidate's `areaName1` field. **True** The input `areaName1` matched the candidate's `areaName1`. **False** The input `areaName1` did not match the candidate's `areaName1`.

Name	Type	Description
matchOnAreaName2	Boolean	Indicates if the input `areaName2` field matched the candidate's `areaName2` field. **True** The input `areaName2` matched the candidate's `areaName2`. **False** The input `areaName2` did not match the candidate's `areaName2`.
matchOnAreaName3	Boolean	Indicates if the input `areaName3` field matched the candidate's `areaName3` field. **True** The input `areaName3` matched the candidate's `areaName3`. **False** The input `areaName3` did not match the candidate's `areaName3`.
matchOnAreaName4	Boolean	Indicates if the input `areaName4` field matched the candidate's `areaName4` field. **True** The input `areaName4` matched the candidate's `areaName4`. **False** The input `areaName4` did not match the candidate's `areaName4`.
matchOnCountry	Boolean	Indicates if the candidate country matches the input country. **True** The candidate country matches the input country. **False** The candidate country does not match the input country.
matchOnAllStreetFields	Boolean	Indicates if the all of the input street fields matched all of the candidate's street fields. **True** All of the input street fields matched all of the candidate's street fields. **False** One or more of the input street fields do not match the candidate's street fields.
matchOnStreetName	Boolean	Indicates if the input street name matched the candidate's street name. **True** The input street name matched the candidate's street name. **False** The input street name did not match the candidate's street name.

Name	Type	Description
matchOnStreetType	Boolean	Indicates if the input street type matched the candidate's street type. **True** The input street type matched the candidate's street type. **False** The input street type did not match the candidate's street type.
matchOnStreetDirectional	Boolean	Indicates if the input street directional matched the candidate's street directional. **True** The input street directional matched the candidate's street directional. **False** The input street directional did not match the candidate's street directional.
matchOnPlaceName	Boolean	Indicates if the input place name matched the candidate's place name. **True** The input place name matched the candidate's place name. **False** The input place name did not match the candidate's place name.
`geometry` object. Returned geocode consisting of the following elements:		
coordinates	Double	The candidate's geocode, specified as x (longitude) and y (latitiude) coordinates separated by a comma.
crs	String	The coordinate reference system used for the candidate's geocode.
type	String	Geometry type. The return value is always `Point`.
`address` object. Returned candidate address which may contain some of the following elements:		
mainAddressLine	String	Candidate address line.
addressLastLine	String	Candidate last address line.
placeName	String	Firm, company, organization, business or building name.
areaName1	String	State, province or region.
areaName2	String	County or district.

Name	Type	Description
areaName3	String	City, town or suburb.
areaName4	String	Locality
postCode1	String	Main postal code.
postCode2	String	Secondary postal code, where one exists.
country	String	Country
addressNumber	String	House or building number.
streetName	String	Street name.
unitType	String	The type of unit, such as Apt., Ste. and Bldg.
unitValue	String	The unit value/number, such as "3B".
customFields	Object	The fields and corresponding values returned are country-specific.
`ranges`: `CandidateRange` object. Contains information about a candidate's ranges, consisting of the following elements:		
placeName	String	If applicable, indicates the name of the candidate's place or building.
lowHouse	String	Indicates the low house number in the candidate's street range.
highHouse	String	Indicates the high house number in the candidate's street range.
side	String	Provides information on the side of street that the candidate's range is located. **LEFT** The range is on the left side of the street. **RIGHT** The range is on the right side of the street. **BOTH** The range is on both the left and right side of the street. **UNKNOWN** No information is available on the side of the street this range is located.

Name	Type	Description
oddEvenIndicator	String	Provides information on the house numbering of the candidate's range. **ODD** The range contains odd house numbers. **EVEN** The range contains even house numbers. **BOTH** The range contains both odd and even house numbers. **IRREGULAR** The range contains both even and odd numbers in an irregular order. **UNKNOWN** No information is available on the odd/even house numbering on this range.
customValues	Map	A map of local values associated with the candidate's range.
`units`: `CandidateRangeUnit` object. Contains information about a candidate range's units, consisting of the following elements:		
placeName	String	If applicable, indicates the name of the candidate's place or building.
unitType	String	Indicates the unit type (APT, STE, etc.).
highUnitValue	String	Indicates the high unit number for this range unit.
lowUnitValue	String	Indicates the low unit number for this range unit.
customValues	Map	A map of local values associated with the unit.
totalPossibleCandidates	Integer	Indicates how many match candidates were found.

Examples

Example: JSON GET Request & Response

The following is an example of a `JSON GET` request for the Geocode service. Note that the query parameters are separated by an ampersand.

```
GET http://myserver:8080/rest/GlobalGeocode/geocode.json?
mainAddress=SANTA ANA&country=Mex&areaName1=DISTRITO FEDERAL
&postalCode=44910 HTTP/1.1
```

The following shows the `JSON` response returned by the previous request.


```
{
   "totalPossibleCandidates": 3,
   "totalMatches": 3,
   "candidates": [
      {
         "precisionLevel": 3,
         "formattedStreetAddress": "",
         "formattedLocationAddress": "44910 GUADALAJARA, JALISCO",
         "identifier": null,
         "precisionCode": "Z1",
         "sourceDictionary": "0",
         "matching": null,
         "geometry": {
            "type": "Point",
            "coordinates": [
               -103.356,
               20.64732
            ],
            "crs": {
               "type": "name",
               "properties": {
                  "name": "epsg:4326"
               }
            }
         },
         "address": {
            "mainAddressLine": "",
            "addressLastLine": "44910 GUADALAJARA, JALISCO",
            "placeName": "",
            "areaName1": "JALISCO",
            "areaName2": "GUADALAJARA",
            "areaName3": "GUADALAJARA",
            "areaName4": "8 DE JULIO 1RA SECC",
            "postCode1": "44910",
            "postCode2": "",
            "country": "MEX",
```

```
            "addressNumber": "",
            "streetName": "",
            "unitType": null,
            "unitValue": null,
            "customFields": {}
         },
         "ranges": []
      }
   ]
}
```

Example: XML GET Request & Response

The following is an example of an XML request for the Geocode service.

```
GET http://myserver:8080/rest/GlobalGeocode/geocode.xml?
mainAddress=18 Merivales St&country=AUS&areaName1=QLD&postalCode=4101
HTTP/1.1
```

The following shows the XML response returned by the previous request.

```
<?xml version="1.0" encoding="UTF-8"?>
<GeocodeServiceResponse>
   <totalPossibleCandidates>1</totalPossibleCandidates>
   <totalMatches>1</totalMatches>
   <candidates>
      <precisionLevel>1</precisionLevel>
      <formattedStreetAddress>
             18 MERIVALE STREET</formattedStreetAddress>
      <formattedLocationAddress>
             SOUTH BRISBANE QLD 4101</formattedLocationAddress>
      <identifier>300211549</identifier>
      <precisionCode>S5HP-TSCZA</precisionCode>
      <sourceDictionary>0</sourceDictionary>
      <geometry>
         <type>Point</type>
         <coordinates>153.01511420131578</coordinates>
         <coordinates>-27.47292827752508</coordinates>
         <crs>
            <type>name</type>
            <properties>
               <name>epsg:4326</name>
            </properties>
         </crs>
      </geometry>
      <address>
         <mainAddressLine>18 MERIVALE STREET</mainAddressLine>
         <addressLastLine>SOUTH BRISBANE QLD 4101</addressLastLine>
         <placeName />
         <areaName1>QLD</areaName1>
         <areaName2>BRISBANE CITY</areaName2>
```

```
         <areaName3>SOUTH BRISBANE</areaName3>
         <areaName4 />
         <postCode1>4101</postCode1>
         <postCode2 />
         <country>AUS</country>
         <addressNumber>18</addressNumber>
         <streetName>MERIVALE</streetName>
         <customFields />
      </address>
      <ranges>
         <lowHouse>6</lowHouse>
         <highHouse>18</highHouse>
         <side>RIGHT</side>
         <oddEvenIndicator>BOTH</oddEvenIndicator>
         <customValues />
      </ranges>
   </candidates>
</GeocodeServiceResponse>
```

Example: JSON POST Request & Response

The following is an example of a `JSON POST` request for the Geocode service. In this example the address point interpolation feature is enabled in `customPreferences`.

```
POST http://myserver:8080/rest/GlobalGeocode/geocode.json HTTP/1.1
{
   "type": "ADDRESS",
   "preferences": {
      "returnAllCandidateInfo": null,
      "fallbackToGeographic": null,
      "fallbackToPostal": null,
      "maxReturnedCandidates": null,
      "distance": null,
      "streetOffset": null,
      "cornerOffset": null,
      "matchMode": null,
      "clientLocale": null,
      "clientCoordSysName": null,
      "distanceUnits": null,
      "streetOffsetUnits": null,
      "cornerOffsetUnits": null,
      "mustMatchFields": {
         "matchOnAddressNumber": false,
         "matchOnPostCode1": false,
         "matchOnPostCode2": false,
         "matchOnAreaName1": false,
         "matchOnAreaName2": false,
         "matchOnAreaName3": false,
         "matchOnAreaName4": false,
         "matchOnAllStreetFields": false,
```

```
            "matchOnStreetName": false,
            "matchOnStreetType": false,
            "matchOnStreetDirectional": false,
            "matchOnPlaceName": false,
            "matchOnInputFields": false
         },
         "returnFieldsDescriptor": null,
         "customPreferences": {
            "USE_ADDRESS_POINT_INTERPOLATION": "true"
         },
         "preferredDictionaryOrders": null
      },
      "addresses": [
         {
            "mainAddressLine": "21 Byng Ave, toronto ON M9W 2M5",
            "addressLastLine": null,
            "placeName": null,
            "areaName1": null,
            "areaName2": null,
            "areaName3": null,
            "areaName4": null,
            "postCode1": null,
            "postCode2": null,
            "country": "CAN",
            "addressNumber": null,
            "streetName": null,
            "unitType": null,
            "unitValue": null,
            "customFields": null
         }
      ]
}
```

The following shows the `JSON` response returned by the previous request.

```
{
   "responses": [
      {
         "totalPossibleCandidates": 1,
         "totalMatches": 1,
         "candidates": [
            {
               "precisionLevel": 16,
               "formattedStreetAddress": "21 BYNG AVE",
               "formattedLocationAddress": "TORONTO ON M9W 2M5",
               "identifier": "29566199",
               "precisionCode": "S8HPNTSCZA",
               "sourceDictionary": "1",
               "matching": null,
               "geometry": {
                  "type": "Point",
                  "coordinates": [
```

```
                        -79.54916,
                        43.72659
                    ],
                    "crs": {
                        "type": "name",
                        "properties": {
                            "name": "epsg:4326"
                        }
                    }
                },
                "address": {
                    "mainAddressLine": "21 BYNG AVE",
                    "addressLastLine": "TORONTO ON M9W 2M5",
                    "placeName": "",
                    "areaName1": "ON",
                    "areaName2": "TORONTO",
                    "areaName3": "TORONTO",
                    "areaName4": "",
                    "postCode1": "M9W",
                    "postCode2": "2M5",
                    "country": "CAN",
                    "addressNumber": "21",
                    "streetName": "BYNG",
                    "unitType": null,
                    "unitValue": null,
                    "customFields": {}
                },
                "ranges": [
                    {
                        "placeName": null,
                        "lowHouse": "21",
                        "highHouse": "21",
                        "side": "LEFT",
                        "oddEvenIndicator": "ODD",
                        "units": [],
                        "customValues": {
                            "AREA_NAME_1": "ON",
                            "POST_CODE_1": "M9W",
                            "POST_CODE_2": "2M5",
                            "AREA_NAME_3": "ETOBICOKE"
                        }
                    }
                ]
            }
        ]
}
```


Example: XML POST Request & Response

The following is an example of an `XML POST` request to the Geocode service. This example illustrates enabling the centerline offset feature in `customPreferences`, as well as setting the

matchOnAddressNumber and matchOnStreetName fields in the mustMatchFields object. To enable the mustMatchFields settings, the matchMode field is set to CUSTOM.

```
POST http://myserver:8080/rest/GlobalGeocode/geocode.xml HTTP/1.1
<?xml version="1.0" encoding="UTF-8"?>
<geocodeRequest>
   <type>ADDRESS</type>
   <preferences>
      <returnAllCandidateInfo
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <fallbackToGeographic
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <fallbackToPostal
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <maxReturnedCandidates
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <distance
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <streetOffset
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <cornerOffset
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:nil="true" />
      <matchMode>CUSTOM</matchMode>
      <mustMatchFields>
         <matchOnAddressNumber>true</matchOnAddressNumber>
         <matchOnPostCode1>false</matchOnPostCode1>
         <matchOnPostCode2>false</matchOnPostCode2>
         <matchOnAreaName1>false</matchOnAreaName1>
         <matchOnAreaName2>false</matchOnAreaName2>
         <matchOnAreaName3>false</matchOnAreaName3>
         <matchOnAreaName4>false</matchOnAreaName4>
         <matchOnAllStreetFields>false</matchOnAllStreetFields>
         <matchOnStreetName>true</matchOnStreetName>
         <matchOnStreetType>false</matchOnStreetType>
         <matchOnStreetDirectional>false</matchOnStreetDirectional>
         <matchOnPlaceName>false</matchOnPlaceName>
         <matchOnInputFields>false</matchOnInputFields>
      </mustMatchFields>
      <customPreferences>
         <entry>
            <key
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">CENTERLINE_OFFSET_UNIT</key>
            <value
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

```
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">FEET</value>
         </entry>
         <entry>
            <key
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">CENTERLINE_OFFSET</key>
            <value xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">30.0</value>
         </entry>
         <entry>
            <key
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">USE_CENTERLINE_OFFSET</key>
            <value
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="xs:string">true</value>
         </entry>
      </customPreferences>
   </preferences>
   <addresses>
      <mainAddressLine>
            36 Rue de la Haute Moline Champagne-Ardenne 10800
      </mainAddressLine>
      <country>FRA</country>
   </addresses>
</geocodeRequest>
```

The following shows the `XML` response returned by the previous request.

```
<?xml version="1.0" encoding="UTF-8"?>
<GeocodeServiceResponseList>
   <responses>
      <totalPossibleCandidates>1</totalPossibleCandidates>
      <totalMatches>1</totalMatches>
      <candidates>
         <precisionLevel>1</precisionLevel>
         <formattedStreetAddress>
               36 rue de la Haute Moline
         </formattedStreetAddress>
         <formattedLocationAddress>
               10800 Saint-Julien-les-Villas
         </formattedLocationAddress>
         <identifier>65277882</identifier>
         <precisionCode>S5HPNTS-ZA</precisionCode>
         <sourceDictionary>0</sourceDictionary>
         <geometry>
            <type>Point</type>
```

```
            <coordinates>4.10284503209829</coordinates>
            <coordinates>48.28588205764661</coordinates>
            <crs>
               <type>name</type>
               <properties>
                  <name>epsg:4326</name>
               </properties>
            </crs>
         </geometry>
         <address>
           <mainAddressLine>36 rue de la Haute Moline</mainAddressLine>

            <addressLastLine>
                   10800 Saint-Julien-les-Villas
            </addressLastLine>
            <placeName />
            <areaName1>Champagne-Ardenne</areaName1>
            <areaName2>Aube</areaName2>
            <areaName3>Saint-Julien-les-Villas</areaName3>
            <areaName4 />
            <postCode1>10800</postCode1>
            <postCode2 />
            <country>FRA</country>
            <addressNumber>36</addressNumber>
            <streetName>de la Haute Moline</streetName>
            <customFields />
         </address>
         <ranges>
            <lowHouse>34</lowHouse>
            <highHouse>38</highHouse>
            <side>RIGHT</side>
            <oddEvenIndicator>EVEN</oddEvenIndicator>
            <customValues />
         </ranges>
      </candidates>
   </responses>
</GeocodeServiceResponseList>
```

Reverse Geocode Requests

For information on GET and POST requests and responses, see the Geocode Service **Geocoding Requests** on page 12.

Reverse Geocode Service Request

GET POST

Reverse Geocode GET Request

The `GET` request enables you to submit an input coordinate and a coordinate reference system, and optionally specify a search distance and country code to use for matching. The associated address data is returned. The preference options for a `GET` request are a subset of the total available with the `POST` request.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/reverseGeocode[,content
type]?[query parameters]
```

where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

[query parameters] are described in the following section.

Query Parameters

The following table defines the `GET` query parameters for the Reverse Geocode service. For information on the response, see **ReverseGeocodeServiceResponse Object** on page 46.

Name	Type	Description
x	Double	Longitude in degrees. Required. For example: -79.391165

Name	Type	Description
y	Double	Latitude in degrees. Required. For example: 43.643469
country	String	Three-letter ISO country code, for example: CAN. Optional. For a list of ISO codes, see **ISO 3166-1 Country Codes** on page 209.
coordSysName	String (URL-encoded)	Specifies the coordinate system that you want to convert the geometry to. The format must be the European Petroleum Survey Group (EPSG) code or the SRID code. Default = `EPSG:4326`. Specify the coordinate reference system in the format `codespace:code`.
distance	Double	Sets the radius in which the Reverse Geocode service searches for a match to the input coordinates. The unit of measurement is specified using `distanceUnits`. Default = 150 meters. Maximum value = 5280 feet (1 mile) or 1609 meters.
distanceUnits	String	Specifies the unit of measurement for the search distance. One of the following: • Feet • Meters - Default

Reverse Geocode POST Request

The `POST` request enables you to submit a single input coordinate or a list of coordinates for batch processing. A country code, coordinate reference system and matching preferences can optionally be specified. A response containing a list of candidates with associated address data and matching information is returned. The preference options for a `POST` request are the complete set of available options.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/reverseGeocode[.content type]
```

where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

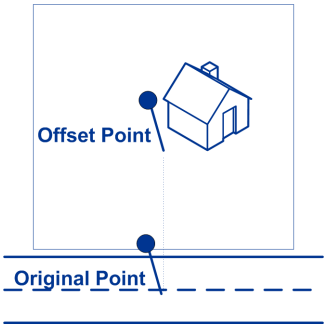
Request Parameters

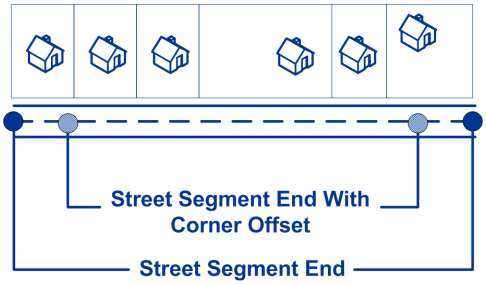
The `POST` request comprises the following input parameters:

- `points` — The input coordinates or multiple input coordinates to be reverse geocoded. Required.
- `preferences` — The matching options. Optional.

These objects and their elements are defined in the following table.

Name	Type	Description
`points` an array object containing both a geometry object and a country code string:		
country	String	Indicates the country to search for the reverse geocode result, specified using a 3-letter ISO country code. Optional. For country codes, see **ISO 3166-1 Country Codes** on page 209.
`geometry` object, consisting of the following elements:		
coordinates	Double	Specifies the x, y input coordinates, where x=longitude and y=latitude. For example: [-105.25175, 40.024494]
type	String	Indicates the type of geographic entity the input coordinates represent. **point** The input coordinates represent a point location.
crs	String	Indicates the coordinate reference system used for the input coordinates. The format must be the European Petroleum Survey Group (EPSG) code or the SRID code. Default = `EPSG:4326`. Specify the coordinate reference system in the format codespace:code.
`preferences` object, consisting of the following elements. **Note:** Only the following elements in the `preferences` object are applicable to the Reverse Geocode service. **Note:** To override the default value of a `preferences` element for a specific country, specify the key/value pair in the `customPreferences` object, with the key constant preceded by the ISO-3166 3-character country code plus period. For example: `DEU.streetOffset`.		
distance	Double	Sets the radius in which the Reverse Geocode service searches for a match to the input coordinates. The unit of measurement is specified using `distanceUnits`. Default = 150 meters. Maximum value = 5280 feet (1 mile) or 1609 meters.

Name	Type	Description
distanceUnits	String	Specifies the unit of measurement for the search distance. One of the following: • Feet • Meters - Default
clientLocale	String	This field is used for a country that has multiple languages to determine the preferred order of language candidates. The locale must be specified in the format "cc_CC", where "cc" is the language and "CC" is the ISO 3166-1 Alpha-2 code, such as: en-US, fr_CA or fr_FR. For example, Egypt supports both english and arabic. The clientLocale field could be set to either english-first (en-EN) or arabic-first (ar-EG). **Note:** For a listing of ISO Alpha-2 country codes, see **ISO 3166-1 Country Codes** on page 209.
POST: clientCoordSysName	String	Specifies the coordinate system that you want to convert the geometry to. The format must be the European Petroleum Survey Group (EPSG) code or the SRID code. Default = `EPSG:4326`. Specify the coordinate reference system in the format `codespace:code`.
streetOffset	Double	The offset distance from the street segments. The distance is in the units you specify in the `streetOffsetUnits` preference. Default value = 7 meters. The offset distance is used in street-level geocoding to prevent the geocode from being in the middle of a street. It compensates for the fact that street-level geocoding returns a latitude and longitude point in the center of the street where the address is located. For example, an offset of 50 feet means that the geocode will represent a point 50 feet back from the center of the street. The distance is calculated perpendicular to the portion of the street segment for the address. Offset is also used to prevent addresses across the street from each other from being given the same point. 

Name	Type	Description
streetOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).
cornerOffset	Double	Distance to offset the street end points in street-level matching. The distance is in the units you specify in the `cornerOffsetUnits` preference. This value is used to prevent addresses at street corners from being given the same geocode as the intersection. Defines the offset position of the geocoded point with respect to the corner. Default value = 7 meters. The following diagram compares the end points of a street to offset end points. 
cornerOffsetUnits	String	Unit of measurement for the street offset. One of the following: **Feet**, **Meters** (default).

Name	Type	Description

customPreferences	Map<String key, String value>	Specifies the country-specific input preferences. This object can be used to specify:
		• A country override to a default value of one or more elements in the `preferences` or `returnFieldsDescriptor` objects.
		• A custom country input option.
		
		To override the default value for a specific country, precede the key constant with the ISO-3 country code plus period, and then specify the value. For example, in an XML request, an entry for a country override would look as follows:
		
		`<customPreferences>`
		`  <entry>`
		`  <key>CAN.distance</key>`
		`  <value>300</value>`
		`  </entry>`
		`</customPreferences>`
		
		Custom country input options are available for the following countries:
		• **Australia (AUS)** on page 93
		• **Canada (CAN)** on page 97
		• **France (FRA)** on page 99
		• **Germany (DEU)** on page 100
		• **Great Britain (GBR)** on page 101
		• **New Zealand (NZL)** on page 102
		• **Portugal (PRT)** on page 104
		• **Singapore (SGP)** on page 105
		• **Sweden (SWE)** on page 106
		• **United States (USA)**
		
		In addition, for countries that support both custom user dictionaries and standard geocoding datasets, you can set a custom preference with the key `KEY_CUSTOM_DICTIONARY_USAGE` that will define the searching and matching preferences when both custom and standard dictionaries are available in the geocoding engine. This option is only available with forward geocoding. For more information, see **Setting Searching and Matching Preferences When Using Standard and Custom Dictionaries**. To locate information on whether your country supports custom user dictionaries, refer to the "Supported Geocoding Datasets" section in the country's write-up.

Reverse Geocode Service Response

ReverseGeocodeServiceResponse Object

A request to the Reverse Geocode service returns a `GeocodeServiceResponse` object that contains:

- `totalPossibleCandidates`— the total number of possible candidates.
- `totalMatches`— the total number of matches.
- `candidates` object — lists one or more candidates that matched to your input coordinate(s). Matching and address information is returned for each candidate.

Table 1: GeocodeServiceResponse Elements Definitions

Name	Type	Description
totalPossibleCandidates	Integer	Indicates the total number of possible candidates.
totalMatches	Integer	Indicates the total number of matches.
`candidates` object of type `Candidate`, consisting of an array with one or more match candidates and associated address, matching and location information. Contains the following elements:		

Name	Type	Description
precisionLevel	Integer	A code describing the precision of the geocode. One of the following: **0** No coordinate information is available for this candidate address. **1** Interpolated street address. **2** Street segment midpoint. **3** Postal code 1 centroid. **4** Partial postal code 2 centroid. **5** Postal code 2 centroid. **6** Intersection. **7** Point of interest. (If database contains POI data.) **8** State/province centroid. **9** County centroid. **10** City centroid. **11** Locality centroid. **12-15** Reserved for unspecified custom items. **16** The result is an address point. **17** The result was generated by using address point data to modify the candidate's segment data. **18** The result is an address point that was projected using the centerline offset feature. You must have both a point and a street range database to use the centerline offset feature. **Note:** This field is not returned for USA. For geocode precision information for USA, see **Location Codes** on page 182.
formattedStreetAddress	String	The formatted main address line.
formattedLocationAddress	String	The formatted last address line.
precisionCode	String	The returned reverse geocoding result code. The definitions are provided in the appendix:. For US, see **Address Location Codes** on page 182; for all other countries, see **Reverse Geocoding 'R' Result Codes** on page 206.

Name	Type	Description
sourceDictionary	String	Identifies the dictionary that is the source for the candidate information and data. The source dictionary is a 0-based integer value that indicates which configured dictionary the candidate came from. If you only have a single dictionary this will always be "0".
`geometry` object. Returned geocode consisting of the following elements:		
coordinates	Double	The candidate's geocode, specified as x (longitude) and y (latitiude) coordinates separated by a comma.
crs	String	The coordinate reference system used for the candidate's geocode.
type	String	Geometry type. The return value is always `Point`.
`address` object. Returned candidate address which may contain some of the following elements:		
mainAddressLine	String	Candidate address line.
addressLastLine	String	Candidate last address line.
placeName	String	Firm, company, organization, business or building name.
areaName1	String	State, province or region.
areaName2	String	County or district.
areaName3	String	City, town or suburb.
areaName4	String	Locality
postCode1	String	Main postal code.
postCode2	String	Secondary postal code, where one exists.
country	String	Country
addressNumber	String	House or building number.
streetName	String	Street name.

Name	Type	Description
unitType	String	The type of unit, such as Apt., Ste. and Bldg.
unitValue	String	The unit value/number, such as "3B".
customFields	Object	The fields and corresponding values returned are country-specific.
`ranges`: `CandidateRange` object. Contains information about a candidate's ranges, consisting of the following elements:		
placeName	String	If applicable, indicates the name of the candidate's place or building.
lowHouse	String	Indicates the low house number in the candidate's street range.
highHouse	String	Indicates the high house number in the candidate's street range.
side	String	Provides information on the side of street that the candidate's range is located. **LEFT** The range is on the left side of the street. **RIGHT** The range is on the right side of the street. **BOTH** The range is on both the left and right side of the street. **UNKNOWN** No information is available on the side of the street this range is located.
oddEvenIndicator	String	Provides information on the house numbering of the candidate's range. **ODD** The range contains odd house numbers. **EVEN** The range contains even house numbers. **BOTH** The range contains both odd and even house numbers. **IRREGULAR** The range contains both even and odd numbers in an irregular order. **UNKNOWN** No information is available on the odd/even house numbering on this range.
customValues	Map	A map of local values associated with the candidate's range.

Examples

Example: JSON GET Request & Response

The following is an example of a `JSON GET` request for the Reverse Geocode service. Note that a value that is associated with more than one key query parameter can be assigned to the parameters by using the following syntax: `parameter1&parameter2=value`.

```
GET http://myserver:8080/rest/GlobalGeocode/reverseGeocode.json?
x=57.70716&y=12.025594&coordSysName=EPSG:4326&
distance=1&distanceUnits=METERS HTTP/1.1
```

The following shows the `JSON` response returned by the previous request.

```
{
   "totalPossibleCandidates": 1,
   "totalMatches": 1,
   "candidates": [
      {
         "precisionLevel": 1,
         "formattedStreetAddress": "KALLKÄLLEGATAN 34",
         "formattedLocationAddress": "416 54 GÖTEBORG",
         "identifier": null,
         "precisionCode": "RS5A",
         "sourceDictionary": "0",
         "matching": null,
         "geometry": {
            "type": "Point",
            "coordinates": [
               57.712566, 12.025625
            ],
            "crs": {
               "type": "name",
               "properties": {
                  "name": "epsg:4326"
               }
            }
         },
         "address": {
            "mainAddressLine": "KALLKÄLLEGATAN 34",
            "addressLastLine": "416 54 GÖTEBORG",
            "placeName": "",
            "areaName1": "VÄSTRA GÖTALANDS LÄN",
            "areaName2": "GÖTEBORG",
            "areaName3": "GÖTEBORG",
            "areaName4": "",
            "postCode1": "416 54",
            "postCode2": "",
```

```
            "country": "SWE",
            "addressNumber": "34",
            "streetName": "KALLKÄLLE",
            "unitType": null,
            "unitValue": null,
            "customFields": {
               "REVERSE_GEOCODE_DISTANCE_UNIT": "METER",
               "REVERSE_GEOCODE_DISTANCE": "0.94200000000000001"
            }
         },
         "ranges": [
            {
               "placeName": null,
               "lowHouse": "34",
               "highHouse": "34",
               "side": "UNKNOWN",
               "oddEvenIndicator": "EVEN",
               "units": [],
               "customValues": {}
            }
         ]
      }
   ]
}
```

Example: XML GET Request & Response

The following is an example of an XML request for the Reverse Geocode service.

```
GET http://myserver:8080/rest/GlobalGeocode/reverseGeocode.xml?
distanceUnits=METER&distance=100&coordSysName=EPSG:4326&y=51.543396
&x=13.419194 HTTP/1.1
```

The following shows the XML response returned by the previous request.

```
<?xml version="1.0" encoding="UTF-8"?>
<GeocodeServiceResponse>
   <totalPossibleCandidates>1</totalPossibleCandidates>
   <totalMatches>1</totalMatches>
   <candidates>
      <precisionLevel>1</precisionLevel>
      <formattedStreetAddress>Am Weinberg 4</formattedStreetAddress>
      <formattedLocationAddress>
              04924 Uebigau-Wahrenbrück
      </formattedLocationAddress>
      <precisionCode>RS5A</precisionCode>
      <sourceDictionary>0</sourceDictionary>
      <geometry>
         <type>Point</type>
         <coordinates>13.41906511750789</coordinates>
         <coordinates>51.54321229045565</coordinates>
         <crs>
            <type>name</type>
            <properties>
               <name>epsg:4326</name>
            </properties>
         </crs>
      </geometry>
      <address>
         <mainAddressLine>Am Weinberg 4</mainAddressLine>
         <addressLastLine>04924 Uebigau-Wahrenbrück</addressLastLine>
         <placeName />
         <areaName1>Brandenburg</areaName1>
         <areaName2>Elbe-Elster</areaName2>
         <areaName3>Uebigau-Wahrenbrück</areaName3>
         <areaName4>Prestewitz</areaName4>
         <postCode1>04924</postCode1>
         <postCode2 />
         <country>DEU</country>
         <addressNumber>4</addressNumber>
         <streetName>Am Wein</streetName>
         <customFields>
            <entry>
               <key
                   xmlns:xs="http:...
                   xmlns:xsi="http:...
```

```
xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE_UNIT</key>
                <value
                     xmlns:xs="http:...
                     xmlns:xsi="http:...
                     xsi:type="xs:string">METERS</value>
            </entry>
            <entry>
               <key
                   xmlns:xs="http:...
                   xmlns:xsi="http:...
                   xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE</key>
               <value
                   xmlns:xs="http:...
                   xmlns:xsi="http:...
                   xsi:type="xs:string">0.983</value>
            </entry>
         </customFields>
      </address>
      <ranges>
         <lowHouse>4</lowHouse>
         <highHouse>6</highHouse>
         <side>UNKNOWN</side>
         <oddEvenIndicator>EVEN</oddEvenIndicator>
         <customValues />
      </ranges>
   </candidates>
</GeocodeServiceResponse>
```

Example: JSON POST Request & Response

The following is an example of a `JSON POST` request for the Reverse Geocode service.

```
POST http://myserver:8080/rest/GlobalGeocode/reverseGeocode.json?
{
   "preferences": {
      "returnAllCandidateInfo": false,
      "fallbackToGeographic": true,
      "fallbackToPostal": true,
      "maxReturnedCandidates": 1,
      "distance": 100,
      "streetOffset": 7,
      "cornerOffset": 7,
      "matchMode": "UNSPECIFIED",
      "clientLocale": "en-US",
      "clientCoordSysName": "epsg:4326",
      "distanceUnits": "METER",
      "streetOffsetUnits": "METER",
      "cornerOffsetUnits": "METER",
      "mustMatchFields": {
         "matchOnAddressNumber": false,
         "matchOnPostCode1": false,
```

```
         ...

         "matchOnStreetName": false,
         "matchOnStreetType": false,
         "matchOnStreetDirectional": false,
         "matchOnPlaceName": false,
         "matchOnInputFields": false
      },
      "returnFieldsDescriptor": {
         "returnAllCustomFields": false,
         "returnMatchDescriptor": false,
         "returnStreetAddressFields": false,
         "returnUnitInformation": false,
         "returnedCustomFieldKeys": []
      },
      "customPreferences": {},
      "preferredDictionaryOrders": []
   },
   "points": [
      {
         "country": "FRA",
         "geometry": {
            "type": "point",
            "coordinates": [
               2.294449,
               48.85838
            ],
            "crs": {
               "type": "name",
               "properties": {
                  "name": "EPSG:4326"
               }
            }
         }
      }
   ]
}
```

The following shows the `JSON` response returned by the previous request.

```
{
   "responses": [
      {
         "totalPossibleCandidates": 2,
         "totalMatches": 2,
         "candidates": [
            {
               "precisionLevel": 2,
               "formattedStreetAddress": "avenue Anatole France",
               "formattedLocationAddress": "75007 Paris",
               "identifier": null,
               "precisionCode": "RS4A",
```

```
            "sourceDictionary": "1",
            "matching": null,
            "geometry": {
               "type": "Point",
               "coordinates": [
                  2.2948623,
                  48.858486
               ],
               "crs": {
                  "type": "name",
                  "properties": {
                     "name": "epsg:4326"
                  }
               }
            },
            "address": {
               "mainAddressLine": "avenue Anatole France",
               "addressLastLine": "75007 Paris",
               "placeName": "",
               "areaName1": "Ile-de-France",
               "areaName2": "Paris",
               "areaName3": "Paris",
               "areaName4": "7e Arrondissement Paris",
               "postCode1": "75007",
               "postCode2": "",
               "country": "FRA",
               "addressNumber": "",
               "streetName": "Anatole France",
               "unitType": null,
               "unitValue": null,
               "customFields": {
                  "REVERSE_GEOCODE_DISTANCE_UNIT": "METER",
                  "REVERSE_GEOCODE_DISTANCE": "23.3"
               }
            },
            "ranges": []
         },
         {
            "precisionLevel": 2,
            "formattedStreetAddress": "parc du Champ de Mars",
            "formattedLocationAddress": "75007 Paris",
            "identifier": null,
            "precisionCode": "RS4A",
            "sourceDictionary": "1",
            "matching": null,
            "geometry": {
               "type": "Point",
               "coordinates": [
                  2.2948623,
                  48.858486
               ],
               "crs": {
                  "type": "name",
```

```
                  "properties": {
                     "name": "epsg:4326"
                  }
               }
            },
            "address": {
               "mainAddressLine": "parc du Champ de Mars",
               "addressLastLine": "75007 Paris",
               "placeName": "",
               "areaName1": "Ile-de-France",
               "areaName2": "Paris",
               "areaName3": "Paris",
               "areaName4": "7e Arrondissement Paris",
               "postCode1": "75007",
               "postCode2": "",
               "country": "FRA",
               "addressNumber": "",
               "streetName": "du Champ de Mars",
               "unitType": null,
               "unitValue": null,
               "customFields": {
                  "REVERSE_GEOCODE_DISTANCE_UNIT": "METER",
                  "REVERSE_GEOCODE_DISTANCE": "23.3"
               }
            },
            "ranges": []
         }
      ]
   }
]
}
```

Example: XML POST Request & Response

The following is an example of a `XML POST` request for the Reverse Geocode service.

```
POST http://myserver:8080/rest/GlobalGeocode/reverseGeocode.xml?
<?xml version="1.0" encoding="UTF-8"?>
<reverseGeocodeRequest>
   <preferences>
      <returnAllCandidateInfo>false</returnAllCandidateInfo>
      <fallbackToGeographic>true</fallbackToGeographic>
      <fallbackToPostal>true</fallbackToPostal>
      <maxReturnedCandidates>1</maxReturnedCandidates>
      <distance>150.0</distance>
      <streetOffset>7.0</streetOffset>
      <cornerOffset>7.0</cornerOffset>
      <matchMode>UNSPECIFIED</matchMode>
      <clientLocale>en-US</clientLocale>
      <clientCoordSysName>epsg:4326</clientCoordSysName>
      <distanceUnits>Meter</distanceUnits>
      <streetOffsetUnits>Meter</streetOffsetUnits>
      <cornerOffsetUnits>Meter</cornerOffsetUnits>
      <mustMatchFields>
         <matchOnAddressNumber>false</matchOnAddressNumber>
         <matchOnPostCode1>false</matchOnPostCode1>
         <matchOnPostCode2>false</matchOnPostCode2>
         <matchOnAreaName1>false</matchOnAreaName1>
         <matchOnAreaName2>false</matchOnAreaName2>
         <matchOnAreaName3>false</matchOnAreaName3>
         <matchOnAreaName4>false</matchOnAreaName4>
         <matchOnAllStreetFields>false</matchOnAllStreetFields>
         <matchOnStreetName>false</matchOnStreetName>
         <matchOnStreetType>false</matchOnStreetType>
         <matchOnStreetDirectional>false</matchOnStreetDirectional>
         <matchOnPlaceName>false</matchOnPlaceName>
         <matchOnInputFields>false</matchOnInputFields>
      </mustMatchFields>
      <returnFieldsDescriptor>
         <returnAllCustomFields>false</returnAllCustomFields>
         <returnMatchDescriptor>false</returnMatchDescriptor>
         <returnStreetAddressFields>false</returnStreetAddressFields>
         <returnUnitInformation>false</returnUnitInformation>
      </returnFieldsDescriptor>
      <customPreferences />
   </preferences>
   <points>
      <country>AUS</country>
      <geometry>
         <type>point</type>
         <coordinates>151.196036</coordinates>
         <coordinates>-33.879637</coordinates>
         <crs>
            <type>name</type>
```

```
            <properties>
               <name>EPSG:4326</name>
            </properties>
         </crs>
      </geometry>
   </points>
</reverseGeocodeRequest>
```

The following shows the XML response returned by the previous request.

```
<?xml version="1.0" encoding="UTF-8"?>
<GeocodeServiceResponseList>
   <responses>
      <totalPossibleCandidates>2</totalPossibleCandidates>
      <totalMatches>2</totalMatches>
      <candidates>
         <precisionLevel>1</precisionLevel>
         <formattedStreetAddress>
                344 WATTLE CRESCENT
         </formattedStreetAddress>
         <formattedLocationAddress>
                ULTIMO NSW 2007
         </formattedLocationAddress>
         <precisionCode>RS5A</precisionCode>
         <sourceDictionary>0</sourceDictionary>
         <geometry>
            <type>Point</type>
            <coordinates>151.19599158560163</coordinates>
            <coordinates>-33.87967421977337</coordinates>
            <crs>
               <type>name</type>
               <properties>
                  <name>epsg:4326</name>
               </properties>
            </crs>
         </geometry>
         <address>
            <mainAddressLine>344 WATTLE CRESCENT</mainAddressLine>
            <addressLastLine>ULTIMO NSW 2007</addressLastLine>
            <placeName />
            <areaName1>NSW</areaName1>
            <areaName2>COUNCIL OF THE CITY OF SYDNEY</areaName2>
            <areaName3>ULTIMO</areaName3>
            <areaName4 />
            <postCode1>2007</postCode1>
            <postCode2 />
            <country>AUS</country>
            <addressNumber>344</addressNumber>
            <streetName>WATTLE</streetName>
            <customFields>
              <entry>
                 <key
```

```
                    xmlns:xs="http:...
                    xmlns:xsi="http:...

xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE_UNIT</key>
                <value
                    xmlns:xs="http:...
                    xmlns:xsi="http:...
                    xsi:type="xs:string">METERS</value>
              </entry>
              <entry>
                <key
                    xmlns:xs="http:...
                    xmlns:xsi="http:...
                  xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE</key>

                <value
                    xmlns:xs="http:...
                    xmlns:xsi="http:...
                    xsi:type="xs:string">1.49</value>
              </entry>
           </customFields>
        </address>
        <ranges>
           <lowHouse>329</lowHouse>
           <highHouse>367</highHouse>
           <side>UNKNOWN</side>
           <oddEvenIndicator>BOTH</oddEvenIndicator>
           <customValues />
        </ranges>
     </candidates>
     <candidates>
        <precisionLevel>1</precisionLevel>
        <formattedStreetAddress>
               344 WATTLE STREET
        </formattedStreetAddress>
        <formattedLocationAddress>
               ULTIMO NSW 2007
        </formattedLocationAddress>
        <precisionCode>RS5A</precisionCode>
        <sourceDictionary>0</sourceDictionary>
        <geometry>
           <type>Point</type>
           <coordinates>151.19599158560163</coordinates>
           <coordinates>-33.87967421977337</coordinates>
           <crs>
              <type>name</type>
              <properties>
                 <name>epsg:4326</name>
              </properties>
           </crs>
        </geometry>
        <address>
           <mainAddressLine>
```

```
                 344 WATTLE STREET
              </mainAddressLine>
             <addressLastLine>
                 ULTIMO NSW 2007
             </addressLastLine>
             <placeName />
             <areaName1>NSW</areaName1>
             <areaName2>COUNCIL OF THE CITY OF SYDNEY</areaName2>
             <areaName3>ULTIMO</areaName3>
             <areaName4 />
             <postCode1>2007</postCode1>
             <postCode2 />
             <country>AUS</country>
             <addressNumber>344</addressNumber>
             <streetName>WATTLE</streetName>
             <customFields>
               <entry>
                 <key
                     xmlns:xs="http:...
                     xmlns:xsi="http:...

xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE_UNIT</key>
                 <value
                     xmlns:xs="http:...
                     xmlns:xsi="http:...
                     xsi:type="xs:string">METERS</value>
               </entry>
               <entry>
                 <key
                     xmlns:xs="http:...
                     xmlns:xsi="http:...
                    xsi:type="xs:string">REVERSE_GEOCODE_DISTANCE</key>

                 <value
                      xmlns:xs="http:...
                      xmlns:xsi="http:...
                      xsi:type="xs:string">1.49</value>
                lt;/entry>
            </customFields>
         </address>
         <ranges>
            <lowHouse>329</lowHouse>
            <highHouse>367</highHouse>
            <side>UNKNOWN</side>
            <oddEvenIndicator>BOTH</oddEvenIndicator>
            <customValues />
         </ranges>
      </candidates>
   </responses>
</GeocodeServiceResponseList>
```

Interactive Geocoding Requests

For information on GET and POST requests and responses, see the Geocode Service **Geocoding Requests** on page 12

Interactive Geocode Service Request

Global Interactive Geocode GET Request

A `GET` request to the Global Interactive Geocode service enables you to enter an address and get immediate feedback as it tries to find match candidates. The returned point is a postal centroid. The preference options for a `GET` request are a subset of the total available with the `POST` request.

Base URI

```
http://<server>:<port>/Geocode/rest/GlobalGeocode/interactive[.content
 type]
```

where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

[parameters] are described in the following section. Each key/value pair entered in the request is separated by an ampersand.

Parameters

The following table defines the `GET` parameters for the Global Interactive Geocode service. For information on the response, see **InteractiveGeocodeServiceResponse Object** on page 65.

Parameter	Type	Description
areaName1	string	Name of state or province
areaName2	string	Name of district or subdivision

Parameter	Type	Description
areaName3	string	Name of city or town
areaName4	string	Name of locality
coordSysName	string	Coordinate system for the data.
country	string	Name of country
distance	double	Distance from origin to candidate
distanceUnits		FEET,METERS,MILES,KILOMETERS, FOOT,METER,MILE,KILOMETER
lastLine	string	Last line of the address
mainAddress	string	Address to be matched. Can include the entire address or some portion.
maxCands	integer	Number of candidates to return. Default is 10. Maximum is 100.
originXY	List (Double)	comma separated double values for XY. For Example, originXY=-73.70252500000001,42.68323
placeName	string	Name of the point of interest (POI data not included)
postalCode	string	Address postcode

Interactive Geocode Service POST Request

A `POST` request to the Interactive Geocode Service service enables you to enter an address and get immediate feedback as it tries to find match candidates. The returned point is a postal centroid. All the preferences in interactive geocoding can be included in a `POST` request.

Base URI

```
http://<server>:<port>/Geocode/rest/GlobalGeocode/interactive[.content
 type]
```

Where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

Preferences

The format for using these preferences is `preferences.CustomPreferences.[<name of preference>]` or `preferences.[<name of preference>]`.

Parameter	Type	Description
SEARCH_TYPE	string	Custom preference to control search type of interactive requests. default: ADDRESS_COMPLETION possible values: ADDRESS_COMPLETION, POINT_OF_INTEREST_COMPLETION, POINT_OF_INTEREST_NAME_COMPLETION, POINT_OF_INTEREST_CATEGORY_COMPLETION, ALL
COMPRESSED_AREA_RESULT	boolean	default: false COMPRESSED_AREA_RESULT
KEY_CUSTOM_DICTIONARY_USAGE	string	possible values: PREFER_CUSTOM_DICTIONARIES, PREFER_STANDARD_DICTIONARIES, USE_CUSTOM_DICTIONARIES_ONLY, USE_STANDARD_DICTIONARIES_ONLY

Parameter	Type	Description
		USE_STANDARD_DICTIONARIES_ONLY
matchMode	string	default: STANDARD, possible values: RELAXED STANDARD, CLOSE
originXY	List Double	`{ "preferences" : { "originXY" : [-73.70252500000001, 42.68323] }, "address" : { "mainAddressLine" : "350 Jordan Rd" } }`
restrictedSearch	Bounds	`{ "preferences": { "restrictedSearch": {"northEastXY": [-73.70252500000001,42.68323], "southWestXY": [-73.70252500000001,42.68323] } }, "address": { "mainAddressLine": "350 Jordan Rd" } }`

Global Interactive Service Response

InteractiveGeocodeServiceResponse Object

For a list of response elemets from the Interactive Geocode service, see **GeocodeServiceResponse Object** on page 23.

Examples

Example: JSON POST Request & Response

Interactive Request

```
{
  "address": {
    "mainAddressLine": "13-15 Quai André Citroën",
    "country": null
  },
  "preferences": {
    "maxReturnedCandidates": 10,
    "distanceUnits": "MILES",
    "distance": null,
    "customPreferences": {
      "COMPRESSED_AREA_RESULT": "false",
      "SEARCH_TYPE": "ADDRESS_COMPLETION"
    },
    "returnAllCandidateInfo": true,
    "originXY": []
  }
}
}
```

Interactive Response

```
{
  "totalPossibleCandidates": 1,
  "totalMatches": 1,
  "candidates": [
    {
      "precisionLevel": 0,
      "formattedStreetAddress": "13-15 Quai André Citroën",
      "formattedLocationAddress": "75015 Paris",
      "matching": {
        "matchOnAddressNumber": true,
        "matchOnPostCode1": false,

        ...

        "matchOnStreetType": false,
        "matchOnStreetDirectional": false,
        "matchOnPlaceName": false,
        "matchOnInputFields": false
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          2.275675,
```

```
48.844045
        ],
        "crs": {
          "type": "name",
          "properties": {
            "name": "epsg:4326"
          }
        }
      },
      "address": {
        "mainAddressLine": "",
        "addressLastLine": "",
        "areaName1": "Île-de-France",
        "areaName2": "Paris",
        "areaName3": "Paris",
        "areaName4": "15e Arrondissement",
        "postCode1": "75015",
        "postCode2": "",
        "country": "FRA",
        "addressNumber": "13-15",
        "streetName": "Quai André Citroën",
        "unitType": "",
        "unitValue": "",
        "customFields": {
          "FORMATTED_ADDRESS": "13-15 Quai André Citroën, 75015 Paris",
          "DISTANCE": "-0.0",
          "FEATUREID": "12500001640586",
          "FROM_CUSTOM_DATASET": "false",
          "MATCHED_FROM_ADDRESSNUMBER": "13 15",
          "MATCHED_FROM_STREETNAME": "QI ANDRE CITROEN",
          "DISTANCE_UNIT": "MILES"
        }
      },
      "ranges": []
    }
  ],
  "customValues": {}
}
```


KeyLookup Requests

For information on GET and POST requests and responses, see the Geocode Service **Geocoding Requests** on page 12.

Global Key Lookup Service Request

Global Key Lookup GET Request

The `GET` request enables you to submit a key to geocode against and get back additional information that enhances your records.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/keyLookup[.content type]
```

where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

Parameters

The following table defines the `GET` parameters for the Key Lookup Service service. For information on the response, see **GeocodeServiceResponse Object**.

Parameter	Type	Description
key	string	Key that is being used to geocode.
type	string	Type of key supported, currently PB_KEY and GNAF-PID
country	string	3-letter ISO code that represents the country for which the lookup is being performed. Currently AUS and USA is supported.

Global KeyLookup POST Request

The `POST` request enables you to submit a key to geocode against and get back additional information that enhanced your records.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/keyLookup.[content type]
```

Where:

[.content type] indicates that the specified content type will be used by default. Optional.

json

Default content type is `JSON`, unless superseded by `HTTP` content negotiation

xml

Default content type is `XML`, unless superseded by `HTTP` content negotiation

Sample JSON Request

```
{
  "type" : "PB_KEY",
  "preferences": {
    "maxReturnedCandidates": 10
  },
  "keys": [
    {
      "country" : "USA",
      "value" : "PB12345678"
    }
    ]
}
```

Global Key Lookup Service Response

GlobalKeyLookupGeocodeServiceResponse Object

For a list of response elemets from the Key Lookup service, see **GeocodeServiceResponse Object** on page 23.

Examples

Example: JSON POST Request & Response

Key Lookup Request

```
{
  "keys": [
    {
      "value": "P0000GL638OL",
      "country": "USA"
    }
  ],
  "type": "PB_KEY",
  "preferences": {
    "returnAllCandidateInfo": true
  }
}
```

Key Lookup Response

```
{
  "responses": [
    {
      "totalPossibleCandidates": 1,
      "totalMatches": 1,
      "candidates": [
        {
          "precisionLevel": 16,
          "formattedStreetAddress": "350 JORDAN RD",
          "formattedLocationAddress": "TROY, NY  12180-8352",
          "identifier": "869200424",
          "precisionCode": "S8H--A",
          "sourceDictionary": "2",
          "matching": {
            "matchOnAddressNumber": false,
            "matchOnPostCode1": true,
            "matchOnPostCode2": true,
            "matchOnAreaName1": true,
            "matchOnAreaName2": false,
            "matchOnAreaName3": true,
            "matchOnAreaName4": false,
            "matchOnAllStreetFields": false,
            "matchOnStreetName": true,
            "matchOnStreetType": true,
            "matchOnStreetDirectional": true,
            "matchOnPlaceName": false,
            "matchOnInputFields": false
          },
          "geometry": {
```

```
      "type": "Point",
      "coordinates": [
        -73.700257,
        42.678161
      ],
      "crs": {
        "type": "name",
        "properties": {
          "name": "epsg:4326"
        }
      }
    },
    "address": {
      "mainAddressLine": "350 JORDAN RD",
      "addressLastLine": "TROY, NY  12180-8352",
      "placeName": "",
      "areaName1": "NY",
      "areaName2": "RENSSELAER COUNTY",
      "areaName3": "TROY",
      "areaName4": "",
      "postCode1": "12180",
      "postCode2": "8352",
      "country": "USA",
      "addressNumber": "350",
      "streetName": "JORDAN",
      "unitType": "",
      "unitValue": "",
      "customFields": {
        "ZIP": "12180",
        "CSA_NUMBER": "104",
        "TYPE_SHORT": "RD",
        "THOROUGHFARE_TYPE": "RD",
        "ROAD_CLASS": "01",
        "MATCH_CODE": "V001",
        "DFLT": "Y",
        "COUNTY": "36083",
        "LANGUAGE": "en",
        "PB_KEY": "P0000GL638OL",
        "POINT_ID": "108535989",
        "LAST_LINE": "TROY, NY  12180-8352",
        "CHECK_DIGIT": "2",
        "MM_RESULT_CODE": "S8H--A",
        "METRO_FLAG": "Y",
        "BLOCK": "360830523011022",
        "QCITY": "361305000",
        "ZIP_FACILITY": "P",
        "LON": "-73.700257",
        "LOT_CODE": "A",
        "LOT_NUM": "0063",
        "CTYST_KEY": "V16572",
        "ZIP_CARRTSORT": "D",
        "LORANGE": "350",
        "STREET_SIDE": "L",
        "DATATYPE": "12",
```

```
                "SEG_LORANGE": "350",
                ...

                "LASTLINE_SHORT": "TROY, NY  12180-8352",
                "DPBC": "99",
                "MAIN_ADDRESS": "JORDAN",
                "NAME_SHORT": "JORDAN",
                "CITY_SHORT": "TROY",
                "ZIP9": "121808352",
                "CITY": "TROY",
                "IS_ALIAS": "N01",
                "ZIP10": "12180-8352",
                "ZIP4": "8352",
                "CBSA_NAME": "ALBANY-SCHENECTADY-TROY, NY METROPOLITAN
STATISTICAL AREA",
                "MATCHED_DB": "2",
                "RANGE_PARITY": "E",
                "LAT": "42.678161"
              }
            },
            "ranges": [
              {
                "placeName": "",
                "lowHouse": "350",
                "highHouse": "350",
                "side": "LEFT",
                "oddEvenIndicator": "EVEN",
                "units": [
                  {
                    "placeName": "",
                    "unitType": "",
                    "highUnitValue": "",
                    "lowUnitValue": "",
                    "customValues": {}
                  }
                ],
                "customValues": {}
              }
            ]
          }
        ],
        "customValues": {}
      }
    ]
}
```

Capabilities Service

Capabilities Service Request

Capabilities GET Request

A `GET` request to the `Capabilities` service returns information:

- supported services
- available geocoding engines
- supported countries
- supported operations and associated required and optional inputs
- custom fields

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/capabilities.[content
type]?[query parameters]
```

where:

.*[content type]* indicates that the specified content type will be used by default. Optional.

JSON Default content type is `JSON`, unless superseded by `HTTP` content negotiation

XML Default content type is `XML`, unless superseded by `HTTP` content negotiation

[query parameters] are described in the following section.

Query Parameters

There are several options for the type of information returned based on the query parameters:

- Include a country code to get the capabilities for the specified country;
- Include a country code and an operation to get the description of that operation; or,
- Exclude all query parameters to get the capabilities for all countries.

The query parameters for the `Capabilities` service are defined in the following table.

Name	Description
country	Name of country in **ISO 3166-1** Alpha-2 or Alpha-3 format, or a common name of the country, such as United States of America.
operation	Type of geocoding service operation. One of the following: • `geocode` • `reverseGeocode` • `interactive` • `keyLookup`

Capabilities Service Response

GeocodeCapabilitiesResponse Object

The following table defines the response elements returned from the `Capabilities` service.

Name	Type	Description
serviceName	String	The name of a supported service.
serviceDescription	String	A description of the service.
coreVersion	String	The core version of Spectrum™ Technology Platform.
geocodingEngines	String	The installed country geocode engine(s).
supportedCountries	String	The countries supported by each installed country geocoder engine.
geocoderVersions	Map	The version number of the geocode engine.
`supported Operations` `Operation` object. An array that defines the supported operations for the specified input country or for all countries consisting of the following fields:		
name	String	Name of the operation.

Name	Type	Description
requiredInputs	InputParameter	Lists the required input fields for the operation. Includes the following elements: • `name` (String) • `description` (String) • `type` (String) • `defaultValue` (String) • `lowBoundary` (String) • `highBoundary` (String) • `allowedValuesWithDescriptions` (Map)
optionalInputs	InputParameter	Lists the optional input fields for the operation. Includes the following elements: • `name` (String) • `description` (String) • `type` (String) • `defaultValue` (String) • `lowBoundary` (String) • `highBoundary` (String) • `allowedValuesWithDescriptions` (Map)
outputs	OutputParameter	Lists the operation's output fields. Includes the following elements: • `name` (String) • `description` (String) • `type` (String)

Name	Type	Description
supportLevels	SupportLevel	Lists the support levels for the operation. Includes the following elements: • `supportedDataLevel` (Integer) **`Data Postal Centroid=1`** Postcode centroids are present in dictionaries (does not distinguish post code 2). **`Data Geographic Centroid=2`** Geographic centroids are present in dictionaries (does not distinguish the type of geographic centroid). **`Data Street Segment=4`** Street segment information present in dictionaries. **`Data Address Point=8`** Point level data present in dictionaries. The data level will contain the sum of all available data keys. For example, **Value — Type of data** `15` — all (postal + geographic + segment + point) `14` — all but postal `13` — all but geographic `12` — point + segment `11` — point + geographic + postal `10` — point + geographic `9` — point + postal `8` — point only `7` — all but point `6` — segment + geographic `5` — segment + postal `4` — segment only `3` — postal + geographic `2` — geographic only `1` — postal only • `countries` — (String) Countries • `updatedRequiredInputs` — (InputParameter) Country-specific required input fields • `updatedOptionalInputs` — (InputParameter) Country-specific optional input fields • `updatedOptionalOutputs` — (OutputParameter) Country-specific output fields
`customObjects` list of type `CustomObject`.		

Name	Type	Description
name	String	The name(s)s of the custom object fields that were user-specified in Preferences.
description	String	The description of the user-specified custom object fields.
properties	list of type CustomObjectMember	Where `CustomObjectMember` contains the following elements: • `name` — (String) Indicates name of parameter. • `input` — (InputParameter) Indicates the property is an input parameter. • `output` —(OutputParameter) Indicates the property is an output parameter.

Examples

Capabilities JSON Request & Response

JSON Request

The following is an example of a `JSON` request for the `Capabilities` service. In this example, the request is for the capabilities for Great Britain.

```
GET http://myserver:8080/rest/GlobalGeocode/capabilities.json?
country=GBR HTTP/1.1
```

JSON Response

The following shows the `JSON` response returned by the previous request. This response is an abbreviated view.

```
{
   "serviceName": "GeocodeService",
   "serviceDescription": "Provides a method to geocode and reverse
geocode",
   "coreVersion": "5.1.0.59",
   "geocodingEngines": [
      "World"
   ],
   "supportedCountries": [
      "XWG"
   ],
   "supportedOperations": [
      {
         "name": "geocode",
         "requiredInputs": [
            {
               "name": "address",
               "description": "The input address",
               "type": "Address",
               "defaultValue": null,
               "lowBoundary": null,
               "highBoundary": null,
               "allowedValuesWithDescriptions": {}
            }
         ],
         "optionalInputs": [
            {
               "name": "type",
               "description": "Indicates what kind of geocode
                               to perform",
               "type": "ONEOF",
```

```
                "defaultValue": "address",
                "lowBoundary": null,
                "highBoundary": null,
                "allowedValuesWithDescriptions": {
                   "geographic": "geographic",
                   "postal": "postal",
                   "address": "address",
                   "custom": "custom"
                }
             },
             {
                "name": "preferences",
                "description": "Contains preferences and constraints",
                "type": "Preferences",
                "defaultValue": null,
                "lowBoundary": null,
                "highBoundary": null,
                "allowedValuesWithDescriptions": {}
             }
          ],
          "outputs": [
             {
                "name": "responses",
                "description": "The geocoded address information",
                "type": "Response"
             }
          ],
          "supportLevels": [
             {
                "supportedDataLevel": 3,
                "countries": [
                   "XWG"
                ],
                "updatedRequiredInputs": [],
                "updatedOptionalInputs": [],
                "updatedOptionalOutputs": [
                   {
                      "name": "CITYRANK",
                      "description": "City ranking from 1 (highest)
                        to 10 (lowest).  0 means no rank available",
                      "type": "KEY"
                   }
                ]
             }
          ]
       },
.
.
.

       {
          "name": "responses",
          "description": "Holds results from a geocode
```

```
                        or reverse geocode operation",
         "properties": [
            {
               "name": "totalPossibleCandidates",
               "input": null,
               "output": {
                  "name": "totalPossibleCandidates",
                  "description": "Number of candidate that could
                                  have been returned from this query",
                  "type": "int"
               }
            },
            {
               "name": "totalMatches",
               "input": null,
               "output": {
                  "name": "totalMatches",
                  "description": "Number of candidates that could
                                   have been returned from this query",
                  "type": "int"
               }
            },
            {
               "name": "candidates",
               "input": null,
               "output": {
                  "name": "candidates",
                 "description": "ordered list of matching candidates",
                  "type": "LIST<Candidate>"
               }
            }
         ]
      }
   ],
   "geocoderVersions": {
      "World": "4.5"
   }
}
```

Dictionaries Service

Dictionaries Service Request

Dictionaries GET Request

A `GET` request to the `Dictionaries` service returns information on the configured dictionaries.

Base URI

```
http://<server>:<port>/rest/GlobalGeocode/dictionaries.[content
type]?[query parameters]
```

where:

.[content type] indicates that the specified content type will be used by default. Optional.

JSON Default content type is `JSON`, unless superseded by `HTTP` content negotiation

XML Default content type is `XML`, unless superseded by `HTTP` content negotiation

[query parameters] are described in the following section.

Query Parameters

There are a couple of options for the type of information returned based on the input query parameters:

- Include a country code to get the dictionaries for the specified country; or
- Exclude all query parameters to get a list of all the configured dictionaries.

The query parameters for the `Dictionaries` service are defined in the following table.

Name	Description
country	Name of country in **ISO 3166-1** Alpha-2 or Alpha-3 format, or a common name of the country, such as United States of America.

Dictionaries Service Response

ConfiguredDictionaryResponse Object

The following table defines the response elements returned from the `Dictionaries` service.

Name	Type	Description
customDictionary	Boolean	Indicates if the dictionary is a user-defined dictionary. **true** The dictionary is a custom, user-defined dictionary. **False** The dictionary is not a custom dictionary.
repositoryName	String	The file name of the dictionary.
path	String	The location of the dictionary on the server.
vintage	String	The data vintage from the vendor.
source	String	The data vendor.
description	String	The name of the dictionary.
`countrySupportInfos`, a collection of `CountrySupport` objects. Each comprising the following elements:		
supportedCountries	List <String>	A list of countries supported by the specified dictionary.
supportedDataTypes	List <DataType>	Type of data in dictionary. One of the following: • POINT • STREET • POST_CODE_1 • POST_CODE_2 • AREA_NAME_1 • AREA_NAME_2 • AREA_NAME_3 • AREA_NAME_4

Examples

Dictionaries JSON Request & Response

JSON Request

The following is an example of a `JSON` request for the `Dictionaries` service. In this example, the request is for a list of configured geocoding datasets for France.

```
GET http://myserver:8080/rest/GlobalGeocode/dictionaries.json?
country=FRA HTTP/1.1
```

JSON Response

The following shows the `JSON` response returned by the previous request.

```
{
   "dictionaries": [
      {
         "customDictionary": false,
         "repositoryName": "MAPMARKER_FR_Navteq_2013_Q4",
         "path": null,
         "vintage": "2013.Q4",
         "source": "Navteq",
         "description": "MAPMARKER_FR_Navteq_2013_Q4",
         "countrySupportInfos": [
            {
               "supportedCountries": [
                  "MYT",
                  "REU",
                  "GUF",
                  "GLP",
                  "MTQ",
                  "FRA",
                  "MCO"
               ],
               "supportedDataTypes": [
                  "POST_CODE_1",
                  "AREA_NAME_3",
                  "STREET"
               ]
            }
         ]
      },
      {
         "customDictionary": false,
         "repositoryName": "MAPMARKER_FR_TomTom_2013_12",
         "path": null,
```

```
         "vintage": "2013.12",
         "source": "TomTom",
         "description": "MAPMARKER_FR_TomTom_2013_12",
         "countrySupportInfos": [
            {
               "supportedCountries": [
                  "MYT",
                  "REU",
                  "GUF",
                  "GLP",
                  "MTQ",
                  "FRA",
                  "MCO"
               ],
               "supportedDataTypes": [
                  "POST_CODE_1",
                  "AREA_NAME_3",
                  "STREET"
               ]
            }
         ]
      }
   ]
}
```

Pitney Bowes Geocoding Connector

Introduction

Previously referred to as "PBLocator", the Pitney Bowes Geocoding Connector allows customers to integrate Pitney Bowes geocoding within third-party systems such as ArcGIS™Online or ArcGIS™ Pro Desktop.

You'll need one of these Pitney Bowes geocoding solutions to generate our geocodes:

- Global Geocoding Software Developer Kit (SDK)
- Spectrum Global Geocoding Module
- Location Intelligence GeoCode API (available at locate.pitneybowes.com)

Geocoding Operations

The Pitney Bowes Geocoding Connector supports the following operations:

- `findAddressCandidates`: Geocode one location or address at a time.
- `geocodeAddresses`: Geocode a list of addresses as a batch with a single request.
- `reverseGeocode`: Returns address or place candidates when given an XY location.
- `suggest`: Provides suggested candidates based on user input character-by-character typing

Example URL

The following is a URL example using the Geocoding Connector. The default URL will be the following if deployed at the root of server.

http://localhost:8080/rest/GeocodeService/arcgis/rest/services/PBLocator/GeocodeServer/findAddressCandidates

findAddressCandidates

`findAddressCandidates` geocodes one location/address per request. The input can be a single line or multiline, along with mandatory and optional parameters. It supports following types of location:

- Street Address
- Street Intersection
- Point of Interest
- Administrative Place Names

- Postal codes

Coordinates, as a type of ESRI location, is not supported by Pitney Bowes Geocoding Connector.

Parameters

`findAddressCandidates` uses required and optional parameters in a GET request to geocode a single address.

Parameter	Details
f	Required: The response format in json, html or kmz. For the Gecoding Connector, the supported format is json.
addressField	Required: The address of the location to be geocoded.
countryCode	Required: Defines the source country for the address.
singleLine	Optional: An address as a single string to be geocoded.
neighborhood	Optional: Neighborhood the address is in.
city	Optional: City the address is in.
subregion	Optional: Subregion the address is in.
region	Optional: Region the address is in.
postal	Optional: Postcode for the address.
postalExt	Optional: Additional postcode for the address.
maxLocations	Optional: The maximum number of locations to be returned.
outFields	Optional: The list of fields to be returned.
outSR	Optional: The well-known ID (WKID) of the spatial reference or a spatial reference JSON object for the returned address candidates.
addressField2	Not supported.
addressField3	Not supported.

Parameter	Details
location	Not supported.
category	Not supported.
matchOutofRange	Not supported.
magicKey	Not supported.
locationType	Not supported.
searchExtent	Not supported.
forStorage	Not supported.

geocodeAddresses

Geocode an entire list of addresses in one request using the `geocodeAddresses` operation. Geocoding many addresses at once is also known as batch or bulk geocoding.

- Street Address
- Street Intersection
- Point of Interest
- Administrative Place Names
- Postal codes

Coordinates, as a type of ESRI location, is not supported by the Geocoding Connector.

Parameters

`geocodeAddresses` uses required and optional parameters in a POST request to batch geocode multiple addresses.

Parameter	Details
f	Required: The response format in json, html or kmz. For the Geocoding Connector, the supported format is json.

Parameter	Details
addresses	Required: Addresses for batch geocoding. SingleLine and multiline addresses are supported.
outFields	Optional: The list of fields to be returned.
outSR	Optional: The well-known ID (WKID) of the spatial reference or a spatial reference JSON object for the returned address candidates.
maxLocations	Optional: The maximum number of locations to be returned.
MaxBatchSize	Optional: Defines the limit of addresses that can be geocoded in a single request. We recommend a batch size of 100-200 addresses.
SuggestedBatchSize	Optional: Specifies the optimal number of addresses to include in a single batch request given the power of the server and the bandwidth.
sourceCountry	Optional: Defines the source country for the addresses.
matchOutofRange	Not supported.
locationType	Not supported.

Example

This is an example for the addresses parameter.

```
{
    "records": [
        {
            "attributes": {
                "OBJECTID": 1,
                "Address": "10 greenhill rd",
                "Neighborhood": "",
                "City": "wayville",
                "Subregion": "",
                "Region": "SA",
                "countryCode": "AUS"
               }
          },
          {
                "attributes": {
                "OBJECTID": 2,
```

```
                "singleLine": "10 downing street London SW1A 2AA",
                "countryCode": "GBR"
            }
        },
        {
            "attributes": {
            "OBJECTID": 3,
            "Address": "1600 PENNSYLVANIA AVE NW",
            "Neighborhood": "",
            "City": "Washington",
            "Subregion": "",
            "Region": "D",
            "countryCode": "USA"
            }
        }
    ]
}
```


reverseGeocode

The `reverseGeocode` operation determines the address at a particular x/y location. You pass the coordinates of a point location to the geocoding service, and the service returns the address or place that is closest to the location.

It supports following types of location:

- Street Address
- Street Intersection
- Point of Interest
- Administrative Place Names
- Postal codes

Parameters

`reverseGeocode` uses required and optional parameters in a GET request to retrieve an address from a point location.

Parameter	Details
f	Required: The response format in json, html or kmz. For the Geocoding Connector, the supported format is json.
location	Required: Location of the point to be matched.

Parameter	Details
distance	Optional: The distance in meters from the location to include in the search area.
outSR	Optional: The well-known ID (WKID) of the spatial reference or a spatial reference JSON object for the returned address candidates.
featureTypes	Not supported.
returnIntersection	Not supported.
locationType	Not supported.

suggest

The method `suggest` allows character-by-character autocomplete suggestions to be generated for user input in a client application. This capability facilitates the interactive search user experience by reducing the number of typed characters before a suggested match is obtained.

- Street Address
- Street Intersection
- Point of Interest
- Administrative Place Names
- Postal codes

Parameters

`suggest` uses required and optional parameters in a GET request to return suggested results from character by character input.

Parameter	Details
f	Required: The response format. The supported format is JSON.
text	Required: The input text to be used to find suggested candidates.

Parameter	Details
countryCode	Required: Defines the source country for the address.
searchExtent	Optional: Confine the search results to a specified area. Use as a starting point to expand as needed.
location	Optional: Confine the search results to a specified area. Use as a starting point to expand as needed.
maxSuggestions	Optional: Maximum number of suggestions to be returned. in a response.
matchOutofRange	Not supported.
locationType	Not supported.

A - Country-Specific Preferences and Fields

In this section

Australia (AUS)

Custom Preferences

Australia supports the following **custom preferences** and **output fields**.

Preference	Description
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.
KEY_STREET_FRONTAGE	When set to true, requests GNAF street frontage points. Default = false.
KEY_GNAF_ORIGINAL	When set to true, returns the GNAF original point coordinates. Default = false.
KEY_POSTAL_CODE_OVERRIDE	When set to true, a matching postal code will match even if the city/suburb does not match. Default = false.

Preference	Description
KEY_RETURN_STREET_TYPE_ABBREVS	When set to true, returns the abbreviated street types instead of the Australia default of fully spelled-out type. Default = false.

Custom Output Fields

Output Field	Description
STREET_TYPE_ABB	The abbreviation for the street type, which is spelled out by default.
ORIGINAL_LATITUDE	The original GNAF latitude.
ORIGINAL_LONGITUDE	The original GNAF longitude.
UD_ORIGINAL_LATITUDE	The original latitude returned from a point-based user dictionary.
UD_ORIGINAL_LONGITUDE	The original longitude returned from a point-based user dictionary.
GNAF_PARCEL_ID	The GNAF parcel identifier.
GNAF_PID	The GNAF Persistent Identifier (GNAF PID) is a 14-character alphanumeric string that uniquely identifies each GNAF address. The PID is constructed from a combination of the major address fields of the GNAF Dictionary. An example of a GNAF PID is: GAACT718519668
GNAF_PRINCIPAL_PID	The Persistent Identifier of the principal address.
GEOCONTAINMENT	This specifies whether the returned coordinates are inside or outside the address boundary. Values are YES for coordinates within, or NO for coordinates outside the boundary.
GEOFEATURE	This field returns a geocode feature type if that was not provided in other GNAF fields. GEOFEATURE corresponds to Geocode Types (GEOCODE_TYPE_AUT Codes) that are described in the PSMA Data Product Description Version 2.7 (Aug. 2012).
GNAF_ADDRESS_CLASS	The GNAF address classification.
GNAF_SA1	The GNAF Statistical Area Level 1 (SA1) identifier.

Output Field	Description
LEVEL_NUMBER	The number of a floor or level in a multistory building. For example, **Floor 2**, 17 Jones Street The GNAF database includes level information for some Australian states. Level information may be associated with unit information, but not necessarily. If the GNAF database contains multiple records with the same level, the level information is returned only if the input address contains unique content (such as a unit number). If the GNAF dictionary has level information for an address, that information is returned with the matched candidate. The correct level information is returned (when available) even if the input address did not include level information, or if the input had the wrong level information. If the input address has level information but the GNAF database does not include level information for the matching address, then the input level information is discarded since it is not validated in the GNAF data.
LEVEL_TYPE	The label used for a floor of a multistory building. For example, "Level" or "Floor". In this example, the level type is "Level": Suite 3 **Level 7**, 17 Jones Street In this example, Suite 3 is a unit.
LOT_NUMBER	Lot numbers are returned for GNAF candidates because some rural addresses do not have adequate physical or house number information.
MESH_BLOCK_ID	A Meshblock is the smallest geographic unit for which statistical data is collected by the Australian Bureau of Statistics (ABS). Meshblocks usually contain a minimum of 20 to 50 households. This is about one fifth the size of a Collection District (CD). You can use the Meshblock ID to do additional attributions against your own data.

Key Lookup Service

The Key Lookup Service provides the following capabilities:

- Takes a unique key for an address.
- Supported keys come from USA or AUS GNAF data (for example, P0000GL638OL for USA data and GAACT715000223 for AUS).
- Supported key types are PB_KEY or GNAF_PID
- Returns a geocoded matched candidate

Canada (CAN)

Custom Preferences

Canada supports the following **custom preferences** and **output fields**.

Preference	Description
KEY_MUST_HAVE_LDU	When true, does not return any matches that do not have the full FSA LDU postal code. Default = false.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNIT	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Custom Output Fields

Field Name	Description
CENSUS_CD	The Census Division (CD) in which the address is located.
CENSUS_CMA	The Census Metropolitan Area (CMA) in which the address is located.
CENSUS_CSD	The Census Subdivision (CSD) in which the address is located.
CENSUS_CT	The Census Tract (CT) in which the address is located.

Field Name	Description
CENSUS_DA	The Dissemination Area (DA) in which the address is located.
FORMATTED_STREET_RANGE	The formatted range data for the address. This field is only returned for postal centroid candidates.

France (FRA)

Custom Preferences

France supports the following custom preferences.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Germany (DEU)

Custom Preferences

Germany supports the following custom preferences.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Great Britain (GBR)

Custom Preferences

Great Britain supports the following custom preferences.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

New Zealand (NZL)

Custom Preferences

New Zealand supports the following custom preferences and output fields.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.
ReturnOriginalCoordinates	When set to true, returns the `ORIGINAL_LONGITUDE` and `ORIGINAL_LATITUDE` values in the candidate's custom output fields.

Custom Output Fields

Field	Description
ALIASED_SUBURB	New Zealand Aliased suburb. An alternative to the officially-recognized suburb name.
KEY_UFI	New Zealand UFI. The Unique Identifier (UFI) identifies the street segment that the geocoded address belongs to. UFIs are up to 7-digit numbers, assigned by New Zealand Post, that uniquely identify each postal delivery point. The UFI is always returned if available, but you cannot use the UFI for input.
NZL_MESHBLOCK_ID	New Zealand Meshblock identifier. A Meshblock is the smallest geographic unit for which statistical data is collected by Statistics New Zealand. Meshblocks vary in size from part of a city block to large areas of rural land.
ORIGINAL_LATITUDE	The original latitude value.
ORIGINAL_LONGITUDE	The original longitude value.

Portugal (PRT)

Custom Preferences

Portugal supports the following custom preferences.

Preference	Description
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Singapore (SGP)

Custom Preferences

Singapore supports the following custom preferences.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Sweden (SWE)

Custom Preferences

Sweden supports the following custom preferences.

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

United States (USA)

Custom Preferences

USA supports **custom preferences** and **custom output fields**.

Additional topics for USA geocoding are included following the custom tables.

Table 2: Custom Preferences

Preference	Description
USE_ADDRESS_POINT_INTERPOLATION	When set to true, enables address point interpolation. Default = false. Supported only in forward geocoding. **Note:** The address point interpolation feature requires that you have a point-level geocoding dataset installed.
CALCULATE_CENTERLINE_PROJECTION_OF_POINT	Computes the closest point on the street from the parcel point. Default = disabled. **Note:** This feature requires that a point-level geocoding dataset is installed.
USE_CENTERLINE_OFFSET	When set to true, calculates the centerline offset for point addresses. Default = false. Supported only in forward geocoding. **Note:** The centerline feature requires that you have a point-level geocoding dataset installed. A centerline point match is indicated by a result code beginning with **SC**.
CENTERLINE_OFFSET	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the distance to offset the point from centerline. Default = 0. Supported in forward geocoding only.
CENTERLINE_OFFSET_UNITS	When `USE_CENTERLINE_OFFSET` is enabled, this specifies the unit type for the centerline offset. Valid values = feet, meters. Default = meters. Supported only in forward geocoding.

Preference	Description
FIND_ADDR_POINT_INTERP	When set to `true`, enables address point interpolation. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** This feature does not work with point addresses in Auxiliary files.
FIND_ADDRCODE	When set to `true`, attempts to standardize and find an address geocode. Set this option if you want the address parsed and standardized. If this option is not set, only the input ZIP and ZIP+4 address elements are used. Default = `false`. **Note:** Supported only in forward geocoding.
FIND_ADDRESS_RANGE	When set to `true`, returns address range information. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS and Exact match modes.
FIND_ALTERNATE_LOOKUP	Determines whether the preferred lookup is to look for the streets first or the firms first. Default = `3`. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS match mode. **1** Prefer street lookup: Matches to the address line, if a match is not made, then matches to the `placeName` line. **2** Prefer firm lookup: Matches to the `placeName` line, if a match is not made, then matches to the address line. **3** Street lookup only: Matches to the address line. Default.

Preference	Description
FIND_BUILDING_SEARCH	Controls the ability to search by building name entered in the address line. When set to `true`, enables matching to building names even when no unit numbers are present. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS match mode. This option is also used when matching to POI data, an optional index file included in the Master Location Data. See **Point of Interest Matching** on page 140.
FIND_CENTERLINE_OFFSET	Offset distance from the street center for a centerline match. Any positive integer, which represents number of feet. Default = `0` feet. **Note:** Supported only in forward geocoding. **Note:** This feature requires a points dataset is installed.
FIND_CENTERLN_PROJ_OF_POINT	When set to `true`, computes the closest point on the street from the parcel point. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** This feature requires a points dataset is installed.
FIND_CLOSEST_POINT	When set to `true`, matches to the nearest point address within the search radius, rather than to the closest feature (e.g. street segment or intersection as well as point addresses). Default = `false`. **Note:** Supported only in reverse geocoding. **Note:** This feature requires that at least one points dataset and one streets dataset are installed; otherwise, the match will be made to the closest feature.

Preference	Description
FIND_CORRECT_LASTLINE	When set to `true`, corrects elements of the output lastline, providing a good ZIP Code or close match on the soundex even if the address would not match or was non-existent. Default = `false`. **Note:** Supported only in forward geocoding.
FIND_DB_ORDER	Allows you to specifically set the order in which User Dictionary and GSD data sets are searched. The default search order is: • User Dictionaries • Point GSD files • Street GSD files Enter a list of geocoding dataset index values [starting at 0, separated by semi-colons] indicating which datasets to search and in what order.
FIND_DPV	When set to `true`, enables delivery point validation. Default = `false`. **Note:** Supported only in forward geocoding.
FIND_EXPANDED_SEARCH_RADIUS	Allows the setting of the radius in miles (up to 99) around which your record lies. Must be used with applicable `FIND_SEARCH_AREA` setting. Default radius setting = `25` miles. **Note:** Ignored in CASS match mode.
FIND_EXPND_SRCH_LIM_TO_STATE	When set to `true`, limits an expanded search to a state's border. Must be used with applicable `FIND_SEARCH_AREA` setting. Default = `true`. **Note:** Ignored in CASS match mode.
FIND_FIRST_LETTER_EXPANDED	When set to `true`, enables extra processing for a bad first letter (missing, wrong, etc.). Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in Exact match mode.

Preference	Description
FIND_LACSLINK	When set to `true`, enables LACS^{Link} lookup. Default = `false`. **Note:** Supported only in forward geocoding.
FIND_MIXED_CASE	When set to `true`, returns candidate information in mixed case rather than uppercase. Default = `false`.
FIND_NEAREST_ADDRESS	When set to `true`, enables matching to addresses interpolated on street segments or to point data locations. You can use `FIND_NEAREST_ADDRESS` and `FIND_NEAREST_INTERSECTION` together to specify reverse geocoding to both addresses and intersections. Set the reverse geocoding processing find properties: `FIND_NEAREST_ADDRESS`, `FIND_NEAREST_INTERSECTION`, and/or `FIND_NEAREST_UNRANGED`. Default = `false`. **Note:** Supported only in reverse geocoding.
FIND_NEAREST_INTERSECTION	When set to `true`, enables matching to intersections. You can use `FIND_NEAREST_INTERSECTION` and `FIND_NEAREST_ADDRESS` together to specify reverse geocoding to both addresses and intersections. Set the reverse geocoding processing find properties: `FIND_NEAREST_ADDRESS`, `FIND_NEAREST_INTERSECTION`, and/or `FIND_NEAREST_UNRANGED`. Default = `false`. **Note:** Supported only in reverse geocoding.
FIND_NEAREST_UNRANGED	When set to `true`, enables matching a street segment with no number ranges. Enabled with `FIND_NEAREST_ADDRESS`. Ignored for point data and intersection matches. **Note:** Supported only in reverse geocoding.

Preference	Description
FIND_PREFER_POBOX	When set to `true`, if both a street address and a PO Box are provided in the input address, the PO Box is used for matching. Default = `false`. If both `FIND_PREFER_POBOX` and `FIND_PREFER_STREET` are set to true, then they are ignored, and the default, `FIND_PREFER_STREET` is used. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS match mode.
FIND_PREFER_STREET	When set to `true`, if both a street address and a PO Box are provided in the input address, the street address is used for matching. Default = `false`. If both `FIND_PREFER_POBOX` and `FIND_PREFER_STREET` are set to true, the default, `FIND_PREFER_STREET` is used. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS match mode.
FIND_PREFER_ZIP_OVER_CITY	When set to `true`, prefers candidates matching the input ZIP Code over matches to the input city. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS and Interactive match modes. Interactive match mode attempts to return the best address regardless of this setting.

Preference	Description
FIND_SEARCH_AREA	This option assists in finding a match when the input address contains limited or inaccurate city or ZIP Code information. One of the following options: `0` Searches the specified city. `1` Searches the entire Finance Area for possible streets. Note: This option has no effect when performing a ZIP centroid match. `2` Allows the setting of an expanded search area. When selected, you can use two other options to set how far to expand the search: `FIND_EXPANDED_SEARCH_RADIUS` and `FIND_EXPND_SRCH_LIM_TO_STATE`. **Note:** Ignored in CASS match mode.
FIND_SEARCH_DIST	When `FIND_APPROXIMATE_PBKEY` is enabled, this parameter sets the distance to use when searching for the nearest address record with an associated pbKey™ unique identifier. The allowable range is 0-5280 feet. Default = 150 feet. **Note:** Supported only in forward geocoding.
FIND_STREET_CENTROID	When set to `true`, enables street locator geocoding. When enabled, if an input street address cannot be found using the street number and name, then the input ZIP Code or city/state is searched for the closest match. If the street is located, a geocode is returned along the matched street segment rather than the geocode for the entered ZIP Code or ZIP + 4. When using street locator geocoding, if no exact matching house number is found, a match code of either E029 (no matching range, single street segment found), or E030 (no matching range, multiple street segment) is returned. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in CASS match mode.

Preference	Description
FIND_SUITELINK	When set to `true`, enables Suite[Link] lookup. Default = `false`. **Note:** Supported only in forward geocoding. **Note:** Ignored in Exact match mode.
FIND_Z_CODE	When set to `true`, attempts to find any ZIP centroid match. Default = `true`. **Note:** Supported only in forward geocoding.
FIND_Z5_CODE	When set to `true`, attempts to find a ZIP centroid match (no ZIP + 4 or ZIP+2). Default = `false`. **Note:** Supported only in forward geocoding.
FIND_Z7_CODE	When set to `true`, attempts to find a ZIP+2 centroid match only (no ZIP+4 or ZIP). Default = `false`. **Note:** Supported only in forward geocoding.
FIND_Z9_CODE	When set to `true`, attempts to find a ZIP+4 centroid match only. Default = `false`. **Note:** Supported only in forward geocoding.
RETURN_LAT_LON_AS_INTEGER_VALUE	When set to `true`, returns the `LAT` and `LON` custom fields as integer values in millionths of degrees. Default = `false`, returns the `LAT` and `LON` custom fields as decimal values. **Note:** Supported only in forward geocoding.

Custom Output Fields

This section lists the unique output fields for USA. Unless otherwise noted, these fields can be returned for both forward and reverse geocoding.

The following categories of output fields are defined:

- **Quality Descriptors**
- **Parsed Address**
- **Point**
- **Centerline**
- **Intersection**
- **Census**
- **Postal**
- **DPV**
- **Residential Delivery Indicator (RDI)**
- **LACS^{Link}**
- **Suite^{Link}**
- **Short Address**
- **Segment**
- **Other**

Quality Descriptors Output Fields

The Quality Descriptors output fields provide information about the results of the matching and geocoding processes.

Field Name	Description
Confidence	The Confidence value for USA candidates is available as a custom output field on the candidate. The key is “CONFIDENCE”.
LocationCode	Location codes indicate the accuracy of the assigned geocode. For descriptions of location codes, see **Address Location Codes** on page 182 in the appendix.
MatchCode	Match codes indicate the portions of the address that matched or did not match to the reference file. For descriptions of match codes, see **Match Codes** on page 170 in the appendix.
PostalFallback	USA fallback candidates returns the correct MatchCode and LocationCode indicating a postal fallback, rather than codes indicating a postal level geocode.

Field Name	Description
MMResultCode	The MapMarker result code for this candidate. See result codes in **Global Result Codes** on page 198 in the appendix.

Parsed Address Output Fields

The Parsed Address output fields provide the components of a matched address which have been parsed and standardized by the geocoder.

Field Name	Description
ParsedAddressLine	The address line for single line input addresses.
ParsedCity	The abbreviated city name from the last line of the input or output address; the value from `ParsedCityName` or `ParsedPreferredCity`.
ParsedCountyName	The county name.
ParsedFirmName	The name of firm from the USPS data or the input firm name. Not applicable to street intersection matches.
ParsedHouseNumber	The house number of input or output address. Not applicable to street intersection matches.
ParsedLastLine	The complete last line of the address.
ParsedMainAddress	The complete first line of the address.
Parsed Name	The street name.
ParsedCityName	The city name for the matched address from the City-State record.
ParsedPreferredCity	The preferred city name for the output ZIP Code of the matched address.
ParsedState	The state abbreviation.
ParsedUnitNumber	The unit number. Not applicable to street intersection matches.
ParsedUnitType	The unit type (APT, STE, etc.). Not applicable to street intersection matches.

Field Name	Description
ParsedZip	5-digit ZIP Code. Not applicable to street intersection matches.
ParsedZip4	4-digit ZIP Code extension.
ParsedZip9	9-digit ZIP Code (ZIP + 4).
ParsedZip10	10-digit ZIP Code (ZIP + 4) with dash separator.

Point Output Fields

The Point output fields provide additional information about the geocode resulting from a match using point-level data.

Note: Except where noted, supported only in forward geocoding.

Field Name	Description
APN ID	The Assessor's Parcel Number Identifier. Not applicable to street intersection matches.
NearestDistance	Gives the distance, in feet, from the input location to the matched street segment, point address, or intersection. **Note:** For reverse geocoding only.
Parcen Elevation	The elevation of the geocode at the parcel centroid. Not applicable to street intersection matches.
PBKey	A unique address identifier that is returned when an address match is made using the Master Location Dataset. The pbKey™ unique identifier is used as a lookup key to a GeoEnrichment dataset, in order to return attribute data for the match. **Note:** For forward and reverse geocoding.
Point ID	The unique point ID of the matched record when matched to point-level data. Blank if the matched record is not from point-level data. Not applicable to street intersection matches.

Centerline Output Fields

Centerline matching is used with point-level matching to tie a point-level geocode with its parent street segment. This type of match provides you with additional data about the parent street segment that is not available with only a point-level match. The output information also includes the bearing and distance from the point data geocode to the centerline match.

Note: Supported only in forward geocoding.

Field Name	Description
CenterlineBearing	For centerline candidates, provides the compass direction, in decimal degrees, from the point data match to the centerline match. Measured clockwise from 0 degrees north.
CenterlineLeftBlock	For centerline candidates, the Census block ID from the left side of the street. Not applicable to street intersection matches.
CenterlineRightBlock	For centerline candidates, the Census block ID from the right side of the street. Not applicable to street intersection matches.
CenterlineLeftSFXBlock	For centerline candidates, the current left block suffix for Census 2010 geography. This field will be blank if the matched record is from point-level data.
CenterlineRightSFXBlock	For centerline candidates, the current right block suffix for Census 2010 geography. This field will be blank if the matched record is from point-level data.
CenterLineDatatype	For centerline candidates, the data type used to make the centerline match. **0** USPS **1** TIGER **2** TomTom Streets geocoding dataset **6** NAVTEQ Streets geocoding dataset **7** TomTom Points geocoding dataset **8** Centrus Points geocoding dataset **9** Auxiliary file **10** User Dictionary **11** HERE Points geocoding dataset **12** Master Location Data

Field Name	Description
CenterlineIsAlias	Three characters indicating that a centerline match was located by an index alias. The first character: **N** Normal street match **A** Alias match (including buildings, aliases, firms, etc.) The next 2 characters: **01** Basic index, normal address match **02** USPS street name alias index **03** USPS building index **05** Statewide intersection alias (when using the Usw.gsi, Use.gsi,or Us.gsi file) **06** Spatial data street name alias (when using the Us_pw.gsi, Usw.gsi, Us_pe.gsi, Use.gsi, Us_ps.gsi, Usp.gsi, Us_psw.gsi, or Us_pse.gsi file is required.) **07** Alternate index (when using ZIP9.gsu, ZIP9E.gsu, and ZIP9W.gsu) **08** LACS^{Link} **09** Unused **09** Auxiliary file match **10** Centrus Alias index (when using usca.gsi) **11** POI index (when using poi.gsi) **12** USPS Preferred Alias **13** ZIPMove match (when using us.gsz). The us.gsz data file contains ZIP+4 codes that have been realigned and a change in city name and/or finance number has occurred. Specifically, ZIPMove allows a change in finance area because CASS searching does not allow a change of finance area unless the input city occurs in more than one finance area, or ZIPMove data contains the old address. **14** Expanded Centroids match (when using us_cent.gsc or bldgcent.gsc)
CenterLineLatitide	For centerline candidates, the latitude as integer value in millionths of degrees.
CenterLineLongitude	For centerline candidates, the longitude as integer value in millionths of degrees.

Field Name	Description
CenterlineName	For centerline candidates, the primary street name.
CenterlineNearestDistance	For centerline candidates, gives the distance, in feet, from the point-level match to the centerline match.
CenterlinePostDirectional	For centerline candidates, the street postfix directional. Can be blank, N, S, E, W, NE, NW, SW, or SE.
CenterlinePreDirectional	For centerline candidates, the street prefix directional. Can be blank, N, S, E, W, NE, NW, SW, or SE.
CenterlineQCity	For centerline candidates, the state, city, or finance numbers.
CenterlineRoadClass	For centerline candidates, the road class code: **0** Minor road, main data file **1** Major road, main data file **10** Minor road, supplemental file **11** Major road, supplemental data file
CenterlineSegmentHiRange	For centerline candidates, provides the high house number in the segment.
CenterlineSegmentLoRange	For centerline candidates, provides the low house number in the segment.
CenterlineSegmentDirection	For centerline candidates, gives the direction of the segment: **F** Numbers are forward. **R** Numbers are reversed.
CenterlineSegmentID	For centerline candidates, the unique segment ID from data vendors.
CenterlineSegmentParity	For centerline candidates, provides the segment parity. The parity indicates which side of the street the odd numbers in the segment are located: **L** Left side of the street **R** Right side of the street **B** Both sides of the street **U** Unknown

Field Name	Description
CenterlineType	For centerline candidates, provides the street type.

Intersection Output Fields

The Intersection output fields provide data about the second segment in an intersection match.

Field Name	Description
BlockLeft2	For intersection matches, the Census block ID from the left side of the street for the second segment in an intersection.
BlockRight2	For intersection matches, the Census block ID from the right side of the street for the second segment in the intersection.
BlockSFXLeft2	For intersection matches, the current left block suffix for Census 2010 geography for the second segment in the intersection.
BlockSFXRight2	For intersection matches, the current right block suffix for Census 2010 geography for the second segment in the intersection.
CBSADivisionName2	For intersection matches, the Core Based Statistical Area (CBSA) division name for the second segment in the intersection.
CBSADivisionNumber2	For intersection matches, the Core Based Statistical Area (CBSA) division number for the second segment in the intersection.
CBSAName2	For intersection matches, the name of the Core Based Statistical Area (CBSA) for the second segment in the intersection.
CBSANumber2	For intersection matches, the Core Based Statistical Area (CBSA) number for the second segment in the intersection.
CountyName2	For intersection matches, the County name for the second segment in the intersection.
County2	For intersection matches, the county FIPS code for the second segment in the intersection.
CSAName2	For intersections matches, the Combined Statistical Area (CSA) name for the second segment in the intersection.

Field Name	Description
CSANumber2	For intersection matches, the Combined Statistical Area (CSA) number for the second segment in the intersection.
DataType2	For intersection matches, the type of data used to make the match for the second segment in the intersection. **0** USPS **1** TIGER **2** TomTom Streets geocoding dataset **6** HERE Streets geocoding dataset **7** TomTom Points geocoding dataset **8** Centrus Points geocoding dataset **9** Auxiliary file **10** User Dictionary **11** HERE Points geocoding dataset **12** Master Location Data
MetroFlag2	Indicates whether the Core Based Statistical Area (CBSA) in which the address is located is a metropolitan area or a micropolitan area. One of the following: **Y** The address is located in a Metropolitan Statistical Area. Metropolitan areas have a population greater than 50,000. **N** The address is not located in a Metropolitan Statistical Area. It is located in a micropolitan area. Micropolitan areas have a population between 10,000 and 49,999. ***Blank*** Is blank (the county does not contain a CBSA).
Name2	For intersection matches, the street name for the second segment of the intersection.
PostDirectional2	For intersection matches, the postfix direction of the second street in the intersection. Can be blank, N, S, E, W, NE, NW, SW, or SE.
PreDirectional2	For intersection matches, the prefix direction of the second street in the intersection. Can be blank, N, S, E, W, NE, NW, SW, or SE.

Field Name	Description
RoadClass2	For intersection matches, the road class code of the second segment in the intersection: **0** Minor road, main data file **1** Major road, main data file **10** Minor road, supplemental file **11** Major road, supplemental data file
SegHiRange2	For intersection matches, provides the high house number of the second segment in the intersection.
SegLoRange2	For intersection matches, provides the low house number of the second segment in the intersection.
SegmentDirection2	For intersection matches, gives the direction of the second segment in the intersection: **F** Numbers are forward. **R** Numbers are reversed.
SegmentID2	For intersection matches, the Segment ID (TLID) or unique ID from premium data vendors for the second segment in the intersection.
SegmentParity2	For intersection matches, provides the segment parity for the second segment in the intersection. The parity indicates which side of the street the odd numbers in the segment are located: **L** Left side of the street **R** Right side of the street **B** Both sides of the street **U** Unknown
Type2	For intersection matches, the street type for the second segment in the intersection.

Census Output Fields

Census output fields contain U.S. Census information about the address.

Field Name	Description
Block	15-digit census block ID/census FIPS code, using the syntax `ssccctttttgbbb` where: • **`ss`**—2-digit State FIPS Code • **`ccc`**—3-digit County FIPS Code • **`tttttt`**—6-digit Census Tract FIPS Code (without period) • **`g`**—Single-digit Block FIPS Code • **`bbb`**—Block FIPS Code Not applicable to street intersection matches.
County	The county FIPS code.
StateFIPS	The state FIPS code.

Postal Output Fields

The Postal output fields contain detailed postal information for the address.

Note: Supported only in forward geocoding.

Field Name	Description
AltFlag	Alternate/base record indicator: **A** Alternate **B** Base
CART	Carrier route ID. Not applicable to street intersection matches.
CheckDigit	A one-digit code at the end of a mailing label barcode.
CountyStateKey	USPS city state key (an alphanumeric value that uniquely identifies a locale in the USPS city state product).
DFLT	Indicates the return status of `Highrise DFLT` or `Rural Routes`: **Y** Either `Highrise DFLT` and `Rural Routes` returned Y. ***Blank*** Both `Highrise DFLT` and `Rural Routes` returned N or *Blank*.

Field Name	Description
DPBCCode	Delivery Point Barcode.
EWSMatch	Indicates if an EWS match was made: **Y** Match denied because it matched to EWS data. **Blank** Input record did not match to EWS data.
Govt Flag	The government building indicator: **A** City government building **B** Federal government building **C** State government building **D** Firm only **E** City government building and firm only **F** Federal government building and firm only **G** State government building and firm only A, B, C, E, F, and G are valid for alternate records only (ALT_FLAG=A). D is valid for both base and alternate records.
HighriseDFLT	Indicates if the match was made to a highrise. **N** Matched to an exact highrise record or a street record. **Y** Did not match to an exact record. Matched to the USPS default highrise record or a street record. Check the input address for accuracy and completeness. ***Blank*** Does not apply to the input address (for example, PO Boxes and General Delivery addresses) or did not find a match.
LotCode	Lot ascending and descending value. Only available for addresses that can be standardized. Blank if running in CASS mode and you have not initialized DPV or the output address does not DPV confirm. **A** Ascending **D** Descending
LotNumber	4-digit eLot number. Requires an input address that can be standardized. Blank if running in CASS mode and you have not initialized DPV or the output address does not DPV confirm.

Field Name	Description
MailStop	Returns address information appearing after mail stop designator words: MSC, MS, MAILSTOP, MAIL STOP, ATTN, ATTENTION.
PMBDesignator	PMB designator.
PMBNumber	PMB number.
RuralRoutes	Match indicator for rural routes. **N** Matched to an exact rural route record. **Y** Did not find an exact record. Matched to the USPS default rural route record. Check the input address for accuracy and completeness. ***Blank*** Does not apply to the input address (for example, street addresses, P.O. Boxes, and General Delivery addresses) or no match found.
URBName	The urbanization name for Puerto Rico.
ZipCarrtSort	Indicates the type of cart sort allowed: **A** Automation cart allowed, optional cart merging allowed. **B** Automation cart allowed, optional cart merging not allowed. **C** Automation cart not allowed, optional cart merging allowed. **D** Automation cart not allowed, optional cart merging not allowed.
ZipClass	ZIP Classification Code: ***Blank*** Standard ZIP Code **M** Military ZIP Code **P** ZIP Code has P.O. Boxes only **U** Unique ZIP Code. (A unique ZIP Code is a ZIP Code assigned to a company, agency, or entity with sufficient mail volume to receive its own ZIP Code.)

Field Name	Description
ZipFacility	Returns the USPS City State Name Facility Code: **A** Airport Mail Facility (AMF) **B** Branch **C** Community Post Office (CPO) **D** Area Distribution Center (ADC) **E** Sectional Center Facility (SCF) **F** Delivery Distribution Center (DDC) **G** General Mail Facility (GMF) **K** Bulk Mail Center (BMC) **M** Money Order Unit **N** Non-Postal Community Name, Former Postal Facility, or Place Name **P** Post Office **S** Station **U** Urbanization

DPV Output Fields

DPV data output fields contain information about a match made using DPV data.

Note: Supported only in forward geocoding.

Field Name	Description
DPVCMRA	Delivery Point Validation CMRA indicator. **Y** Address found in CMRA table. **N** Address not found in CMRA table. ***Blank*** DPV not loaded.

Field Name	Description
DPVConfirm	Indicates if a match occurred for DPV data. **N** Nothing confirmed. **Y** Everything confirmed (ZIP+4, primary and secondary) **S** ZIP+4 and primary (house number) confirmed. **D** ZIP+4 and primary (house number) confirmed and a default match (`HI_RISE_DFLT = Y`, secondary did not confirm. ***Blank*** Non-matched input address to USPS ZIP+4 data, or DPV data not loaded.
DPVFalsePOS	DPV false-positive indicator. **Y** False-positive match found. ***Blank*** False-positive match not found.
DPVFootNote1	Information about the matched DPV records. **AA** ZIP+4 matched. **A1** Failure to match a ZIP+4. ***Blank*** Address not presented to hash table or DPV data not loaded.

Field Name	Description
DPVFootNote2	Information about the matched DPV records. **BB** All DPV categories matched. **CC** DPV matched primary/house number, where the secondary/unit number did not match (present but invalid). **M1** Missing primary/house number. **M3** Invalid primary/house number. **N1** DPV matched primary/house number, with a missing secondary number. **P1** Missing PS, RR, or HC Box number. **P3** Invalid PS, RR or HC Box number. **F1** All military addresses. **G1** All general delivery addresses. **U1** All unique ZIP Code addresses. ***Blank*** Address not presented to hash table or DPV data not loaded. **Note:** A unique ZIP Code is a ZIP Code assigned to a company, agency, or entity with sufficient mail volume to receive its own ZIP Code.
DPVFootNote3	Information about the matched DPV records. **R1** Matched to CMRA but PMB designator not present. **R2** Matched to CMRA and PMB designator present (PMB 123 or #123). ***Blank*** Address not presented to hash table or DPV data not loaded.
DPVNoSTAT	**Y** The address is valid for CDS pre-processing. **N** The address is not valid for CDS pre-processing. ***Blank*** DPV is not loaded or DPV did not confirm.
DPVShutdown	**Y** Address was found in false-positive table. **N** Address was not found in false-positive table. ***Blank*** Address was not presented to hash table or DPV data not loaded.

Field Name	Description	
DPVVacant	**Y**	The address is vacant.
	N	The address is not vacant.
	Blank	DPV is not loaded or DPV did not confirm (so vacancy is irrelevant).

Residential Delivery Indicator (RDI) Output Fields

The Residential Delivery Indicator (RDI™) is a United States Postal Service (USPS®) data product that identifies whether a delivery type is classified as residential or business. If you are shipping to residences, you may lower costs by shipping with the Postal Service™ and avoid residential delivery surcharges typically charged by other shipping companies.

Note: To use RDI, Delivery Point Validation (DPV) must also be enabled and a US Streets dataset loaded.

Field Name	Description
RDIRetCode	USPS Residential Delivery Indicator (RDI) return codes: • Y = Residence • N = Business • Blank = Not processed through RDI.

LACS^{Link} Output Fields

LACS^{Link} data output fields contain information about a match made using the LACS^{Link} dataset.

Note: Supported only in forward geocoding.

Field Name	Description	
LACSLinkFlag	Indicates if the address is marked for conversion.	
	L	Address marked for LACS conversion.
	Blank	Address not marked for LACS conversion.

Field Name	Description
LACSLinkIND	LACS^{Link} indicator. **Y** Matched LACS^{Link} record. **N** LACS^{Link} match NOT found. **F** False-positive LACS^{Link} record. **S** Secondary information (unit number) removed to make a LACS^{Link} match. ***Blank*** Not processed through LACS^{Link}.
LACSLinkRetCode	LACS^{Link} return code. **A** Matched LACS^{Link} record. **00** LACS^{Link} match NOT found. **09** Matched to highrise default, but no LACS^{Link} conversion. **14** Found LACS^{Link} match, but no LACS^{Link} conversion. **92** Secondary information (unit number) removed to make a LACS^{Link} match. ***Blank*** Not processed through LACS^{Link}.
LACSLinkShutdown	**Y** False-positive occurred and LACSLink library shutdown. **N** LACSLink library has not shutdown or not loaded.

Suite^{Link} Output Fields

The Suite^{Link} output fields contain information about a match made using the Suite^{Link} dataset.

Note: Supported only in forward geocoding.

Field Name	Description
SuiteLink_Ret_Code	**A** Suite^{Link} record match. **00** No Suite^{Link} match. ***Blank*** This address was not processed through Suite^{Link}.

Short Address Output Fields

The short address output fields contain abbreviated elements of the matched address.

Note: Supported only in forward geocoding.

Field Name	Description
ShortAddressline	Shortest possible address line that can be constructed from available short street name and other address line components.
ShortCityName	The output city name that appears in `LASTLINE_SHORT`. This value is determined by logic similar to `CITY`. Whenever possible, this city name is 13 characters or less. This output city name is determined by CASS rules. This can be either City State Name, City State Name Abbreviation, or Preferred Last Line City State Name.
ShortLastline	The address last line. Whenever possible, this field is 29 characters or less: • 13-character city name • 2 (comma and space) • 2-character state abbreviation • 2 spaces • 10-digit ZIP Code
ShortStreetName	The short street name used to construct the short address line. All attempts are made to abbreviate this name according to the process specified by the USPS in the 30 Character Abbreviation - Cycle M Flow Chart. If an abbreviated address cannot be constructed that is 30 characters or less, this field then contains the same street name value as the `NAME` field return.
ShortPostDirectional	Postdir from the `ADDRLINE_SHORT` field.
ShortPreDirectional	Predir from the `ADDRLINE_SHORT` field.
ShortStreetType	Street type from the `ADDRLINE_SHORT` field.

Segment Output Fields

Segment output fields contain information on the street segment identified by the data provider.

Field Name	Description
LeftBlockID	Census block ID from the left side of the street. Not applicable to street intersection matches.
RightBlockID	Census block ID from the right side of the street. Not applicable to street intersection matches.
LeftSFXBlock	The current left block suffix for Census 2010 geography. This field will be blank if the matched record is from point-level data.
RightSFXBlock	The current right block suffix for Census 2010 geography. This field will be blank if the matched record is from point-level data.
DataType	The type of data used to make the match. **0** USPS **1** TIGER **2** TomTom Streets geocoding dataset **6** HERE Streets geocoding dataset **7** TomTom Points geocoding dataset **8** Centrus Points geocoding dataset **9** Auxiliary file **10** User Dictionary **11** HERE Points geocoding dataset **12** Master Location Data
DataTypeName	The source data vendor for the candidate match.
HiRange	House number at the high end of the range. Not applicable to street intersection matches.
HighUnit	High unit number for the range. Not applicable to street intersection matches.
HiZip4	High ZIP+4 for the range. Not applicable to street intersection matches.

Field Name	Description
IsStreetAlias	The first character: **N** Normal street match **A** Alias match (including buildings, aliases, firms, etc.) The next 2 characters: **01** Basic index, normal address match **02** USPS street name alias index **03** USPS building index **05** Statewide intersection alias (when using the Usw.gsi, Use.gsi,or Us.gsi file) **06** Spatial data street name alias (when using the Us_pw.gsi, Usw.gsi, Us_pe.gsi, Use.gsi, Us_ps.gsi, Usp.gsi, Us_psw.gsi, or Us_pse.gsi file is required.) **07** Alternate index (when using ZIP9.gsu, ZIP9E.gsu, and ZIP9W.gsu) **08** LACS^{Link} **09** Unused **09** Auxiliary file match **10** Centrus Alias index (when using usca.gsi) **11** POI index (when using poi.gsi) **12** USPS Preferred Alias **13** ZIPMove match (when using us.gsz). The us.gsz data file contains ZIP+4 codes that have been realigned and a change in city name and/or finance number has occurred. Specifically, ZIPMove allows a change in finance area because CASS searching does not allow a change of finance area unless the input city occurs in more than one finance area, or ZIPMove data contains the old address. **14** Expanded Centroids match (when using us_cent.gsc or bldgcent.gsc)
LoRange	House number at the low end of the range. Not applicable to street intersection matches.
LowUnit	Low unit number. Not applicable to street intersection matches.
LoZip4	Low ZIP+4 for this range. Not applicable to street intersection matches.

Field Name	Description
NearestDistance	Gives the distance, in feet, from the input location to the matched street segment, point address, or intersection. **Note:** For reverse geocoding only.
StreetPostDirectional	Postfix direction. Can be blank, N, S, E, W, NE, NW, SW, or SE.
StreetPreDirectional	Prefix direction. Can be blank, N, S, E, W, NE, NW, SW, or SE.
QCity	State, city, or finance numbers.
RangeParity	Indicates the parity of the house number in the range: **E** Even **O** Odd **B** Both
RecType	The range record type: **A** Auxiliary file **F** Firm **G** General Delivery **H** Highrise **P** Post Office/PO Box **R** Rural Route **S** Street **T** TIGER record match **U** User Dictionary Not applicable to street intersection matches.
RoadClass	The road class code: **0** Minor road, main data file **1** Major road, main data file **10** Minor road, supplemental file **11** Major road, supplemental data file Not applicable to street intersection matches.

Field Name	Description
SegmentHighRange	Provides the high house number in the segment.
SegmentLowRange	Provides the low house number in the segment.
SegmentDirection	Gives the direction of the segment: **F** Numbers are forward. **R** Numbers are reversed.
SegmentID	Segment ID (TLID) or unique ID from premium data vendors. Not applicable to street intersection matches.
SegmentParity	Provides the segment parity. The parity indicates which side of the street the odd numbers in the segment are located: **L** Left side of the street **R** Right side of the street **B** Both sides of the street **U** Unknown
StreetSide	The matched address is on the following side of the street: **L** Left side of the street. **R** Right side of the street. **B** Both sides of the street. **U** Unknown side of the street. This is relative to the segment end points and the segment direction (`SEGMENT_DIRECTION`).
ThoroughfareType	Street type.

Other Output Fields

The Other output fields contain additional information about the match.

Field Name	Description
AUXUserData	User data from an auxiliary file. Blank if no auxiliary file.

Field Name	Description
CBSADivisionName	Core Based Statistical Area (CBSA) division name.
CBSADivisionNumber	Core Based Statistical Area (CBSA) division number.
CBSAName	The name of the Core Based Statistical Area (CBSA) in which the address is located. A CBSA is a collective term that refers to both metropolitan and micropolitan areas. A metropolitan area has a population of more than 50,000, and a micropolitan area has a population between 10,000 and 49,999. For more information, see *Metropolitan and Micropolitan Statistical Areas* section of the U.S. Census Bureau website:**http://www.census.gov/population/www/metroareas/metroarea.html**
CBSANumber	Core Based Statistical Area (CBSA) number.
CSAName	Combined Statistical Area (CSA) name.
CSANumber	Combined Statistical Area (CSA) number.
LAT	The latitude of the address.
LON	The longitude of the address.
MatchedDB	Index of geocoding dataset for matched record.
MCDName	Minor Civil Division name from the auxiliary file. Blank if no auxiliary file match.
MCDNumber	Minor Civil Division number from the auxiliary file. Blank if no auxiliary file match.
MetroFlag	Indicates whether the Core Based Statistical Area (CBSA) in which the address is located is a metropolitan area or a micropolitan area. One of the following: **Y** The address is located in a Metropolitan Statistical Area. Metropolitan areas have a population greater than 50,000. **N** The address is not located in a Metropolitan Statistical Area. It is located in a micropolitan area. Micropolitan areas have a population between 10,000 and 49,999. ***Blank*** Is blank (the county does not contain a CBSA).
ResolvedLine	Indicates which line in a 2-line address was used to resolve the address.

Additional Topics for USA Geocoding

MLD Extended Attributes including APN and Elevation

(USA) This feature provides access to extended attributes associated with an addressable location that has a pbKey. When matching addresses with Master Location Data (MLD), Spectrum™ Technology Platform returns additional property information associated with the address, such as Assessor's Parcel Number (APN), Elevation, Address Type, and Lot Size. APN can be used to identify the parcel so the parcel ID can be linked to additional information for the insurance industry, such as property and insurance risk attributes. For more detail, see the full list of output fields below.

Requirements

The following are required to return MLD Extended Attributes:

- Master Location Data dataset.
- Streets dataset.
- MLD Extended Attributes dataset.
- Recommendation: the vintages of the MLD and MLD Extended Attributes datasets be within 4 months of each other.

MLD Extended Attributes Output Fields (optional)

Field	Description
ADDRTYPE	Address Type regarding number of units: S – Single unit M – Multiple units P – Post Office box X – Unknown
APN_ID	Assessor’s parcel number.
INC_IND	Incorporated Place Indicator. I – Incorporated place N – Not an incorporated place X – Unknown

Field	Description
LOTSIZE	Lot size of the parcel expressed in square feet; 0 if none.
LOTSIZE_METERS	Lot size of the parcel expressed in square meters; 0 if none.
MEC_LAT	Latitude of Minimum Enclosing Circle expressed with an implied 6 digits of decimal precision; 0 if none.
MEC_LON	Longitude of Minimum Enclosing Circle expressed with an implied 6 digits of decimal precision; 0 if none.
MEC_RAD	Radius of Minimum Enclosing Circle (in square feet) expressed as a whole number. For example: 1234 means 1,234 feet.
MEC_RAD_METERS	Radius of Minimum Enclosing Circle (in meters) expressed with 1 digit of decimal precision.
PARCEN_ELEVATION	Elevation above sea level (in feet) expressed with 1 digit of decimal precision. For example: 12.5 feet.
RESBUS	Usage Indicator: R – Residential use B – Business use M – Mixed use – residential and business X – Unknown use
TFID	TIGER Face Identifier. This field can be used to match to all Census geocodes using external data; 0 if none.
PLACE	TIGER Place code; 0 if none.
UACE	TIGER Urban Area Identifier. Defines the urban area if any; 0 if none.
UACEPOP	Census population of the urban area; 0 if none.

Field	Description
URBANICITY	Urbanicity Indicator. An indicator that defines, according to the Census, the Urbanicity of the Address using TIGER UACE codes for categorization.

PBKey ZIP Centroid Locations

The default behavior of the geocoder is to return matches from Master Location Data for addressable locations that have an address-level geocode. The PBKey ZIP Centroid feature enables returning ZIP centroids when matching to MLD. For addresses that don't have a high-quality location, this provides access to the pbKey™ unique identifier, which can be used to unlock additional information about an address using GeoEnrichment data, as well as to realize operational processing efficiencies. This allows maximum address coverage and integrity in geocoding. The inclusion of these addresses enables providing a higher match rate, lower false-positive match rate, and access to the pbKey™ unique identifier for all known addresses in the US. Default=disabled. To enable, use the appropriate CLI command. For more information, see **setting Command**.

Note: This feature is only available with Master Location Data.

Point of Interest Matching

The optional Point Of Interest (POI) Index file (*poi.gsi*) included with the Master Location Data and HERE Point Addresses datasets provides expanded support in alias name matching.

To enable POI matching:

1. Add the MLD or HERE Point Addresses data as a Database Resource.
2. Ensure `FIND_BUILDING_SEARCH` is set to `true`. The POI Index file will automatically be searched when this option is enabled and a firm, building or POI name is specified in the `mainAddress` input field.
3. If an alias match is made to the POI Index file, the `IsStreetAlias` output field, or, in the case of a centerline match, `CenterlineIsAlias` field, returns `A11`.

Expanded Centroids

In some cases, more than one point-level geocode is available for an address matched in Master Location Data (MLD). For more information about the different types of point-level geocodes, see the "`APnn`" definitions in **Address Location Codes** on page 182. When more than one point-level geocode is available from MLD data, only the highest quality geocode is returned with the matched address data.

The Expanded Centroids feature is available with MLD and the optional Master Location Structure Centroid Data Set (MLDB). If an address match is found in MLD, and the MLDB data set is added as a database resource, the MLDB data set is searched for additional geocodes for the matched address. If additional geocodes are found for the matched address, these are returned.

The returned location code for an Expanded Centroids match will have an "APnn" value with a data type of "MASTER LOCATION".

An Expanded Centroids match is indicated by an "A14" value for the IsAlias return field. The returned location code for an Expanded Centroids match will have an "APnn" value with a data type of "MASTER LOCATION".

Extended Match Codes

Extended Match Codes return additional information about any changes in the house number, unit number and unit type fields. In addition, it can indicate whether there was address information that was ignored. The Extended Match Code is only returned for address-level matches (match codes that begin with A, G, H, J, Q, R, S, T or U), in which case a 3rd hex digit is appended to the match code (see **Match Codes** on page 170).

> **Note:** A typical match code contains up to 4 characters: a beginning alpha character followed by 2 or 3 hex digits. The third hex digit is only populated for intersection matches or as part of the Extended Match Code.
>
> For information about the 3rd hex digit values for:
>
> - Intersection matches, see **Definitions for 1st-3rd hex digit match code values** on page 173
> - Extended Match Codes, see **Definitions for Extended Match Code (3rd hex digit values)** on page 175

The return of the Extended Match Code is enabled by default and cannot be modified.

Extended Match Code return values

"Address information ignored" is specified when any of these conditions apply:

- The input address is a dual address (two complete addresses in the input address). For example, "4750 Walnut St. P.O Box 50".
- The input last line has extra information that is not a city, state or ZIP Code, and is ignored. For example, "Boulder, CO 80301 USA", where "USA" is ignored when matching.

The table below provides descriptions of the Extended Match Code 3rd hex digit return values.

Input Addressline	Output Addressline	Extended Code	Description
4750 WALNUT ST STE 200	4750 WALNUT ST STE 200	0	Matched on all address information on line, including Unit Number and Unit Type if included.

Input Addressline	Output Addressline	Extended Code	Description
4750 WALNUT ST C/O JOE SMITH	4750 WALNUT ST	1	Matched on Unit Number and Unit Type if included. Extra information on address line ignored. Extra information not considered for matching is not returned.
4750 WALNUT ST UNIT 200	4750 WALNUT ST STE 200	2	Matched on Unit Number. Unit Type changed.
4750 WALNUT ST UNIT 200 C/O JOE SMITH	4750 WALNUT ST STE 200	3	Matched on Unit Number. Unit Type changed. Extra information on address line ignored. Extra information not considered for matching is not returned.
4750 WALNUT ST STE 2-00	4750 WALNUT ST STE 200	4	Unit Number changed or ignored.
4750 WALNUT ST STE 2-00 C/O JOE SMITH	4750 WALNUT ST STE 200	5	Unit Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
4750 WALNUT ST STE 400	4750 WALNUT ST STE 400	6	Unit Number changed or ignored. Unit Type changed or ignored. In this example, Suite 400 is not valid for the input address, but the address match is not prevented because of an invalid unit number.
4750 WALNUT ST UNIT 2-00 C/O JOE SMITH	4750 WALNUT ST STE 200	7	Unit Number changed or ignored. Unit Type changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
47-50 WALNUT ST STE 200	4750 WALNUT ST STE 200	8	Matched on Unit Number and Unit Type if included. House number changed or ignored.
47-50 WALNUT ST STE 200 C/O JOE SMITH	4750 WALNUT ST STE 200	9	Matched on Unit Number and Unit Type if included. House number changed or ignored. Extra information not considered for matching is not returned.
47-50 WALNUT ST UNIT 200	4750 WALNUT ST STE 200	A	Matched on Unit Number. Unit Type changed. House Number changed or ignored.
47-50 WALNUT ST UNIT 200 C/O JOE SMITH	4750 WALNUT ST STE 200	B	Matched on Unit Number. Unit Type changed. House Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.

Input Addressline	Output Addressline	Extended Code	Description
47-50 WALNUT ST STE 20-0	4750 WALNUT ST STE 200	C	House Number changed or ignored. Unit Number changed or ignored.
47-50 WALNUT ST STE 20-0 C/O JOE SMITH	4750 WALNUT ST STE 200	D	House Number changed or ignored. Unit Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
47-50 WALNUT ST UNIT 20-0	4750 WALNUT ST STE 200	E	House Number changed or ignored. Unit Number changed or ignored. Unit Type changed or ignored.
47-50 WALNUT ST UNIT 2-00 C/O JOE SMITH	4750 WALNUT ST STE 200	F	House Number changed or ignored. Unit Number changed or ignored. Unit Type changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.

City-only Lastline Matching

City-only lastline matching permits address matching with only a city in the input lastline. The city should be provided using either the `mainAddress` (using single-line address input), `LastLine` or `areaName3` input fields.

With city-only lastline input, all of the states are searched in which the input city exists. Therefore, there is the possibility of an increase in multimatches (return of E023 or E030 Match Codes) when matching with city-only input instead of city+state input.

Restrictions:

- City-only lastline input matching is not supported in CASS mode.
- City-only lastline is not supported when matching to User Dictionaries.
- When matching using city-only lastline, the `FIND_PREFER_ZIP_OVER_CITY` setting is ignored.
- It is strongly recommended to not use city-only lastline matching in Relaxed match mode to avoid the return of false-positive matches.

Correct Lastline

When `FIND_CORRECT_LASTLINE` is set to `true`, the elements of the output lastline are corrected, providing a good ZIP Code or close match on the soundex even if the address did not match or was non-existent. This feature is disabled by default.

The feature works when `FIND_ADDRCODE` is `true` and the address does not match a candidate or when `FIND_Z_CODE` is `true` and only lastline information is input.

For example, when `FIND_ADDRCODE` = `true`

Address: 0 MAIN
Lastline: BOLDER CA 80301

Returns:

MATCH_CODE=E622
LASTLINE=BOULDER, CO 80301
CITY=BOULDER STATE=CO
ZIP=80301

For example, `FIND_Z_CODE` = `true`

Address: Lastline: BOLDER CA 80301

Returns:

MATCH_CODE=Z6
LASTLINE=BOULDER, CO 80301
CITY=BOULDER STATE=CO
ZIP=80301

When Correct Lastline is enabled, the following elements are corrected:

- **City correction** - The city correction is based on input ZIP Code unless a match to city and state exists in which case both search areas are retained. The input state must be correct or spelled out correctly when no ZIP Code is input. The returned location code and coordinates are based on the output ZIP Code.
  - Input city is incorrect:

    HAUDENVILLE MA 01039
    Returns LASTLINE=HAYDENVILLE, MA 01039 LAT= 42396500 LON= -72689100

- **State correction** - State is abbreviated when spelled out correctly or corrected when a ZIP Code is present. There are some variations of the input state which are recognized, ILL, ILLI, CAL, but not MASS. The US geocoder does not consider the abbreviation of the variation a change, so ILL to IL is not identified as a change in the match code. In addition, the output of the ZIP Code for a single ZIP Code city is not considered a change.
  - Input city exists:

    Bronx NT, 10451
    Returns LASTLINE= BRONX, NY 10451

    Bronx NT
    Returns LASTLINE= BRONX NT
    No ZIP Code for correction.

  - Input city does not exist - preferred city for ZIP Code returned:

60515
Returns LASTLINE=DOWNERS GROVE, IL 60515
MATCH_CODE=E622

ILLINOIS 60515 (or ILL 60515 or IL 60515 or ILLI 60515)
Returns LASTLINE=DOWNERS GROVE, IL 60515
MATCH_CODE=E222

- **ZIP Code correction** - The ZIP Code is corrected only when a valid city/state is identified and has only one ZIP Code.
  - Exists on input:

    HAUDENVILLE MA 01039
    Returns LASTLINE=HAYDENVILLE, MA 01039

  - Incorrect on input - ZIP Code correction is not performed, both search areas are retained:

    HAUDENVILLE MA 01030
    Returns LASTLINE=HAYDENVILLE, MA 01030
    City and ZIP Code do not correspond.

  - Does not exist on input:

    DOWNRS GROVE, IL
    Returns LASTLINE=DOWNERS GROVE, IL
    City with multiple ZIP Codes.

    LILSE IL
    Returns LASTLINE=LISLE, IL 60532
    City with a single ZIP Code.

    DOWNERS GROVE LL
    Returns LASTLINE=DOWNERS GROVE LL,
    No ZIP Code for correction.

    DOWNRS GROVE, LL
    Returns LASTLINE=DOWNRS GROVE, LL
    No ZIP Code for correction.

    LILSE ILLINOIS
    Returns LASTLINE= LISLE, IL 60532
    Correct spelled out state.

    LISLE ILLINOS
    Returns LASTLINE= LISLE ILLINOS
    Incorrect spelled out state, no ZIP Code for correction.

Note: For information about the returned match codes, see **Correct Lastline Match Codes** on page 178.

Address Range Matching

Some business locations are identified by address ranges. For example, a shopping plaza could be addressed as 10-12 Front St. This is how business mail is typically addressed to such a business location. These address ranges can be geocoded to the interpolated mid-point of the range.

Address ranges are different from hyphenated (dashed) addresses that occur in some metropolitan areas. For example, a hyphenated address in Queens County (New York City) could be 243-20 147 Ave. This represents a single residence (rather than an address range) and is geocoded as a single address. If a hyphenated address returns as an exact match, Spectrum™ Technology Platform does not attempt to obtain an address range match.

Address range matching is not available in Exact or CASS modes, since an address range is not an actual, mailable USPS® address. The following fields are not returned by address range geocoding:

- ZIP + 4® (in multiple segment cases)
- Delivery point
- Check digit
- Carrier route
- Record type
- Multi-unit
- Default flag

Address range matching works within the following guidelines:

- There must be two numbers separated by a hyphen.
- The first number must be lower than the second number.
- Both numbers must be of the same parity (odd or even) unless the address range itself has mixed odd and even addresses.
- Numbers can be on the same street segment or can be on two different segments. The segments do not have to be contiguous.
- If both numbers are on the same street segment, the geocoded point is interpolated to the approximate mid-point of the range.
- If the numbers are on two different segments, the geocoded point is based on the last valid house number of the first segment. The ZIP Code and FIPS Code are based on the first segment.
- In all cases, odd/even parity is evaluated to place the point on the correct side of the street.

Unsupported Preferences

- When matching using the `mustMatchFields` settings, the matchOnAreaName2 and matchOnAreaName4 preferences are not supported.

B - Custom Dataset Builder

In this section

Custom Dataset Builder

Pitney Bowes Custom Dataset Builder is a stand-alone command line utility that allows customers to create custom datasets and integrate address data with the Spectrum Global Geocoding Module (GGM). Custom datasets thus created can complement the GGM's standard datasets. In addition, the utility allows enhancing and optimizing geocoding behaviors to accommodate features unique to a particular dataset.

Custom Dataset Builder Workflow

- Unpack the package. Confirm source data meet requirements.
- Determine supported countries and languages. Create a sample geocoding configuration for one or all supported countries. Customize the geocoding configuration.
- Build a custom dataset. Integrate datasets with Spectrum.

Features

The Custom Dataset Builder supports:

- Forward geocoding of street and address points for supported countries, including data unique to a particular dataset.
- Reverse geocoding for all supported countries with the exception of the United States.
- Data integration for more than 100 countries and languages supported by the Global Geocoding Module. See **Custom Dataset Builder Supported Countries**.
- MapInfo TAB source file input format (Native and NativeX) is supported.

Limitations

Before you begin using the Custom Dataset Builder, please consider the following:

- The Custom Dataset Builder does not support TAB file names that contain special characters, □, #, $, or %, for example. If a TAB file you intend to use with the Custom Dataset Builder contains special characters, you need to rename it.
- The Custom Dataset Builder does not support the byte order mark (BOM) Unicode character. Some editors, Notepad, for instance, add a BOM when you save text as UTF-8; therefore, when editing a JSON file, you should use an editor that does not add a BOM, Notepad++ for example.

- Data creation for a country using multiple Tab files is not supported.
- The optional parameter `-usePackagedLib` will only work with Spectrum version 2019.1 or higher and SPD bundles OCT2019 or higher.

Source Data Requirements

Source data must conform to the following requirements:

- The source file must be a MapInfo Tab (Native or NativeX).
- The data's source records should contain either point geometries or line geometries (segmented data). In case latitude and longitude are available in tabular format (separate columns) in the tab file, you will need to generate geometries using these columns.
- The data has to be in a schema that contains all required fields, which are mapped during the dataset building process. If a value of a required field is empty for a particular record, then that record will not be imported into the dataset.
- The search area code (SAC) field should preferably be mapped to the postcode, as this serves the most logical grouping for most of the cases – not required for United States, Germany and Canada.
- Spectrum 2019.1 or higher is required and the vintage of the SPD bundles must be OCT2019 or later.

Getting Started with Custom Dataset Builder

The Custom Dataset Builder is available when you install Spectrum Technology Platform and the Global Geocoding Module. It is located in `server\modules\GlobalGeocode\customdatasetbuilder`. Documentation (this document) is located in the directory.

Custom Dataset Builder Commands

> **Note:** Before using the Custom Dataset Builder commands, identify the paths to all installed Global Geocoding Module datasets.

Custom Dataset Builder commands are executed from the command line from the root of the installed location of the tool. Each command has a leading – (hyphen). The available commands are:

- **help**: Provides the user a list of commands which Custom Dataset Builder offers to the customer, and educates them on how to utilize those to onboard their data effectively

```
java -Xmx512m -jar cdb-<version>.jar –help
```

- **findCountryWithLanguage**: This command enables you to understand what countries are supported by the Custom Dataset Builder to create custom data. In addition, it provides information about the language of the data. Both are being written to a text file (placed parallel to the cdb-<version>.jar) that the user can use later to generate the initial configuration per country per supported language.

```
java -Xmx512m -jar cdb<version>.jar
-findCountryWithLanguage-folderLocation="$folderLocation" -usePackagedLib
```

Parameters for the findCountryWithLanguage command:

- folderLocation: value will be parent folder location where all the SPD's are placed in extracted format
- usePackagedLib: optional parameter [required for USA] that uses the libraries bundled with the Custom Dataset Builder tool instead of using the library from the SPD.

- **createConfig**: This command enables the user to create a country-specific configuration, which is a JSON file, which contains the step-by-step mapping of the user data information to onboard their data.

```
java -Xmx512m -jar cdb<version>.jar-createConfig
-folderLocation="$folderLocation"
-country="$country_code"-dataType="$data_type" -language="$language_code"
 -userProfile="basic/advance"-usePackagedLib
```

Parameters for the createConfig command

- folderLocation: value will be parent folder location where all the SPD's are placed in extracted format
- country: mandatory information, which is required for generating any of the configuration which user needs to provide and country for which the configuration file needs to be created. Country information is passed as a 3-letter ISO code only.
- language: optional field, which provides the user to specify the language of the data user wants to onboard. In addition, this field needs to match with the current offering by Pitney Bowes geocoding software. By default the value of this field is set to "en" – which is Latin or plain English.
- datatype: optional parameter which specifies the type of data being intended to onboard – Ranged data maps to "Street" while the point data maps to "AP". Example: datatype=Street
- userProfile: optional parameter. Based upon the technical aspect of the user, the configuration can be basic or advanced. Basic being default. Basic creates default JSON without having any advanced configuration in it.

Advanced persona of the user profile has the entire configuration as offered by the basic, but also offers some additional config elements and is not supported for DEU, CAN and USA.

- usePackagedLib: optional parameter [required for USA] that uses the libraries bundled with the Custom Dataset Builder tool instead of using the library from the SPD.

- **buildAll**: Similar to the createConfig command, the buildAll command enables the user to create a configuration of all the supported countries and languages at once.

```
java -Xmx512m -jar cdb-^<version>.jar -buildAll -folderLocation=
"$folderLocation" -userProfile="basic/advance" -usePackagedLib
```

Parameters for the buildAll command

- folderLocation: value will be parent folder location where all the SPD's are placed in extracted format
- userProfile: optional parameter. Based upon the technical aspect of the user, the configuration can be basic or advanced. Basic being default. Basic creates default JSON without having any advanced configuration in it.

  Advanced persona of the user profile has the entire configuration as offered by the basic, but also offers some additional config elements and is not supported for DEU, CAN and USA.

- usePackagedLib: optional parameter [required for USA] that uses the libraries bundled with the Custom Dataset Builder tool instead of using the library from the SPD.

- **createDictionary**: Once the user is done with all the relevant configuration as described in the createConfig command, the command lets the user initiate the build process of onboarding the user data into the Pitney Bowes geocoding software consumable format.

```
java -Xmx512m -jar cdb<version>.jar -createDictionary
-folderLocation="$folderLocation"
-configFilePath="$configFilePath"  -usePackagedLib
```

Parameters for the createDictionary command

- configFilePath: argument is the absolute path of the JSON file.
- folderLocation: value will be parent folder location where all the SPD's are placed in extracted format
- usePackagedLib: optional parameter [required for USA] that uses the libraries bundled with the Custom Dataset Builder tool instead of using the library from the SPD.

Building a Custom Dataset

Building a custom dataset involves using customized JSON files as input and executing the build command that creates the binary files that compose the dataset as output.

To build a custom dataset execute the following command from the command prompt.

```
java -Xmx512m -jar cdb-<version>.jar -createDictionary
-configFilePath="$configFilePath"
```

The Custom Dataset Builder builds the dataset and places it in the folder you specified.

Integration with Spectrum

After building a custom dataset and putting it in the destination folder for the country to which it applies, you can select it for use using Spectrum Management Console.

Creating a Configuration File for a Single Country

Creating a sample configuration file for a single country establishes a default JSON file you can use to modify and build a custom dataset.

To create a sample configuration file for a country, execute the following command at the command prompt.

```
java -Xmx512m -jar cdb-&version>.jar -createConfig
-folderLocation="$folderLocation" -country="$country_code"
-dataType="$data_type" -language="$language_code"
-userProfile="basic/advance"
```

Refer to **Custom Dataset Builder Commands** for details about individual parameters.

The Custom Dataset Builder creates the JSON file for that country.

FRA Configuration

Custom Dataset Builder supports data creation for France, the French-administered territories of Guadeloupe (GLP), French Guiana (GUF), Martinique (MTQ), Mayotte (MYT), Réunion (REU), and the country of Monaco (MCO) using the similar command as used for other countries.

When geocoding any address from the territories, provide all the relevant settings as you would for France (including the country code FRA, not the territory code). Matching candidates are returned from those territories along with the country code of its parent (i.e., FRA).

Requirements:

- Data must be in TAB format (native or nativeX).

Result:

- When using custom datasets for street geocoding, the result code contains a "U" for user datasets to distinguish it from "A" when candidates are from the standard address datasets. For example: S5HPNTSCZ**U** instead of S5HPNTSCZ**A**.

Limits:

- Data created with Custom Dataset Builder does not support interactive geocoding at this time.

USA Configuration

You need to provide certain values in `USA_DataManagerSettings.properties` for creating custom datasets with USA data. The properties file is located alongside the cdb-<version>.jar.

```
java -Xmx512m -jar cdb-<version>.jar -createDictionary
-folderLocation="$folderLocation" -configFilePath="$configFilePath"
-usePackagedLib
```

> **Note:** USA data creation requires a streets dataset is installed. Additionally, you must use the -usePackagedLib parameter with Spectrum 2019.1 or higher is required and the vintage of the SPD bundles must be OCT2019 or later.

DICTIONARY_PATH1: value will be the path of the folder where the USA address dictionaries are present in extracted format.

LIB_PATH: value will be the path of the OS-specific DLL's available in the GGM module's bin.

Example: LIB_PATH="..\Spectrum\server\modules\GlobalGeocode\bin"

usePackagedLib: this parameter is required for USA data creation. It uses the libraries bundled with the Custom Dataset Builder tool instead of using the library from the SPD.

Requirements:

- Spectrum 2019.1 or higher.
- The vintage of the SPD bundles must be OCT2019 or later.
- A streets dataset must be installed.
- Data must be in TAB format (native or nativeX).

Result:

- When using custom datasets for street geocoding, the result code contains a "U" for user datasets to distinguish it from "A" when candidates are from the standard address datasets. For example: S5HPNTSCZ**U** instead of S5HPNTSCZ**A**.

Limits:

- Data created with Custom Dataset Builder does not support interactive geocoding at this time.

Creating a Configuration File for All Supported Countries

Creating a sample configuration file for all countries establishes default JSON files which can be modified and used to build custom datasets.

To create sample configuration files for all supported countries, execute the following command at the command prompt.

```
 java -Xmx512m -jar cdb-<version>.jar -buildAll -folderLocation=
"$folderLocation" -userProfile="basic/advance"
```

Customizing a Geocoding Configuration

Customizing a geocoding configuration involves modifying the configuration's properties in the sample JSON files. The JSON files utilize two property types:

- Build-time properties are used during both data creation and geocoding.
- Run-time properties are applicable only during geocoding.

> **Note:** A custom geocoding configuration only applies to a specific dataset. It does not affect the geocoding behavior of other datasets.

To customize a geocoding configuration:

1. Open the JSON file you want to edit in a text file editor.
2. Modify the necessary property key values.
3. Close the file.

Review the following sets of properties for potential customization in a dataset's JSON file.

configuration

This set of build time properties defines the dataset's configuration. The properties are:

- `country` – This property identifies the country to which the dataset applies. The value is a three-letter ISO country code in all capital letters. For example: AUT.
- `dataName` – This property indicates the dataset's name. Possible values are AP and STREET in uppercase. AP represents address points. STREET represents street data.
- `dataProviderName` – This property identifies the vendor that is the source of the data. Recommended not to change this property
- `dataReader` – This property identifies the data reader. The value is Tab. Recommended not to change this property
- `dictionaryType` – This property identifies the dataset type. Values is Street. Recommended not to change this property.
- `dataLanguage` – This property indicates the language the dataset uses. The value is a two-letter abbreviation. For example: en. Recommended not to change this value

The following is an example of the configuration properties:

```
"Configuration": {
        "country": "AUT",
        "dataName": "STREET",
        "dataProviderName": "TA",
        "dataReader": "Tab",
        "dictionaryType": "Street",
        "dataLanguage": "en"
        }
```

Note: Both properties and values are in quotation marks.

field

This set of build time properties defines the dataset's field formats. The properties are:

- `StreetName`: Indicates the street column
- `PostCode`: Indicates the postcode column
- `AreaName1`: This property indicates the stateprovince column
- `AreaName1` (USA specific): Mapped to a column which is a state abbreviation
- `AreaName2`: This property indicates the county column
- `AreaName3` : This property indicates the city column

- `AreaName4`: This property indicates the locality column
- `StartingAddressNumber` :This property indicates the starting number for address number ranges for the left and right sides of a road.
- `EndingAddressNumber`: This property indicates the ending number for address number ranges for the left and right sides of a road.
- `StreetSideIndicator`: This property indicates even and/or odd address number structures for the left and right sides of the road. The column being mapped should have one of the following values as supplied in the table below. Any other value being mapped may result in a data creation error.

Column Value	Description	Example
0 or 1	No address number range	
2	Even Ranges From Left – To Left (2-10)	2,4,6,8,10
3	Odd ranges From Left – To Left (1-9)	1,3,5,7,9
4	Mixed From Left – To Left (1-10)	1,2,3,4,5,6,7,8,9,10

- `geometry_name`: This property pairs the key GeometryName with the value “GEOM.”
- `StreetAdditionalFields`: This property indicates whether or not additional street candidate information is necessary
- `RangeAdditionalFields`: This property indicates whether or not additional range information is necessary.
- `UnitAdditionalFields`: This property identifies an additional field at the unit level.
- `PostalAdditionalFields`: This property identifies an additional field at the administrative boundary level.

  The following properties are subordinate to the properties above.

  - `Comments`: Description about the property
  - `keys`: This property identifies single or multiple keys for a particular field. It nests under any of the above properties. Recommended not to change existing keys as generated. Addition is allowed for the additional fields at different levels.
  - `values` : This property indicates the name of the column to which the field is mapped in the custom data source. It nests under any of the above properties.

- `altValues`: An optional field, and indicates alternate value to the key being mapped.

Canada-specific `altValue`: The postal code in Canada comprises 6 digits. Of these 6 digits, the 1st three digits are mapped to `values`, the last three digits are mapped to `altValues`.

```
"PostCode" : {
   "keys": ["LeftPostCode", "RightPostCode"],
   "values": ["PostalCode", "PostalCode"],
   "altValues”: ["PostalCode_AddOn", PostalCode_AddOn"]
   }
"PostCode" : {
   "keys": ["LeftPostCode", "RightPostCode"],
   "values": ["PostalCode", "PostalCode"],
   "altValues”: ["PostalCode_AddOn", PostalCode_AddOn"]
   }
 "PostCode" : {
      "Comments" : "Mapping for Post Code and Extended Post Code from
source data.",
      "keys" : [ "LeftPostCode", "RightPostCode" ],
      "values" : [ "Left_postalcode_5", "Right_postalcode_5" ],

      "altValues" : [ "Left_postalcode_3", "Right_postalcode_3" ]
    },
```

Singapore: Postal codes are mapped in 2 columns. The first column contains the initial 2 digit postcode. The second column contains the last 4 digits.

For address points the `value` and `altValue` are postcode2, postcode4

For street data: l_postcode2/l_postcode4/r_postcode2/r_postcode4

```
"PostCode" : {
      "Comments" : "Mapping for Post Code and Extended Post Code from
source data.",
      "keys" : [ "LeftPostCode", "RightPostCode" ],
      "values" : ["l_postcode2", "r_postcode2"],
      "altValues" : ["l_postcode4", "r_postcode4"]
    },
```

The following is an example of the field properties:

```
"field": {
    "StreetName": {
"Comments" : "Mapping for Street Name and Street Name Alias from source
 data.",
     "keys" : "StreetName",
     "values" : "STRASSE",
 "altValues" : ""
    },
    "PostCode" : {
"Comments" : "Mapping for Post Code and Extended Post Code from source
 data.",
```

```
      "keys" : [ "LeftPostCode", "RightPostCode" ],
      "values" : [ "PLZ", "PLZ" ],
      "altValues" : [ "", "" ]
    },
    "AreaName3" : {
      "keys" : [ "LeftAreaName3", "RightAreaName3" ],
      "values" : [ "ORT", "ORT" ],
      "altValues" : [ "", "" ]
    },
    "AreaName4" : {
      "keys" : [ "LeftAreaName4", "RightAreaName4" ],
      "values" : [ "ORTSTEIL", "ORTSTEIL" ],
      "altValues" : [ "", "" ]
    },
    "StreetSideIndicator" : {
   "Comments" : "Mapping for Street Side Indicator from source data.",
      "keys" : [ "LeftStreetSideIndicator", "RightStreetSideIndicator"
 ],
      "values" : [ "", "" ]
    },
    "StartingAddressNumber" : {
  "Comments" : "Mapping for Starting Address Number from source data.",

      "keys" : [ "FromLeftStartingAddressNumber",
"FromRightStartingAddressNumber" ],
      "values" : [ "HAUSNR_VON", "HAUSNR_VON" ]
    },
    "EndingAddressNumber" : {
   "Comments" : "Mapping for Ending Address Number from source data.",
      "keys" : [ "ToLeftEndingAddressNumber",
"ToRightEndingAddressNumber" ],
      "values" : [ "HAUSNR_VON", "HAUSNR_VON" ]
    },
 "StreetAdditionalFields" : {
   "Comments" : "Mapping for Additional Fields at Street level from
source data.",
      "keys" : [ "sub_locality", "sub_town" ],
      "values" : [ "ORTSTEIL", "ORT" ]
    },
    "RangeAdditionalFields" : {
  "Comments" : "Mapping for Additional Fields at Range level from source
 data.",
      "keys" : [ "RangeIdentifier" ],
      "values" : [ "ORTSTEIL" ]
    },
 "UnitAdditionalFields" : {
  "Comments" : "Mapping for Additional Fields at Unit level from source
 data.",
      "keys" : [ "UnitIdentifier" ],
      "values" : [ "ORTSTEIL" ]
    },
 "PostalAdditionalFields" : {
   "Comments" : "Mapping for Additional Fields at Postal level from
```

```
source data.",
      "keys" : [ "PostalIdentifier" ],
      "values" : [ "ORTSTEIL" ]
    },
    "geometry_name" : {
   "Comments" : "Mapping for Geometry from source data.",
      "keys" : "GeometryName",
      "values" : ""
    }
  }
```

dataReader

This set of build time properties defines the dataset's data reader property. The properties are:

- `tab`: This property indicates the reader is a TAB file reader.

  The following properties are subordinate to the property above.

  - `TABFile`: This property identifies the TAB file. It nests under the tab property.
  - `inputPath`: This property indicates the path to the custom source data. For the tab property, this is the complete file path.

The following is an example of the `dataReader` properties:

```
"dataReader": {
 "Comments": "Mapping for input file path and TAB file name."
    "tab" : {
      "inputPath" : "<InputTabFileFolder>/AUT_TAB",
      "TABFile" : "AT_scheme_dummy_sample1"
    }
  }
```

output

This build time property defines the output path for the custom dataset. For example:

```
"output" : {
    "outputPath" : "<FolderLocation>/AUT_UD"
  }
```

errata

This build time property defines the field mapping for creating the search area code (Sac).

For example:

```
"errata" : {
"SacFromFile" : [ "PostalCode", "PostalCode" ]
}
```

`SacFromFile` in the JSON must be numeric. Postal codes are numeric for most countries and can be used to define the SacFromFile. For those countries where postal codes are not numeric, the Custom Dataset Builder requires a field that can provide a logical grouping.

advancedConfigs

This set of run time properties defines custom configuration values. It includes the subsets Abbreviations, Post_StreetTypes, and Pre_StreetTypes, which in turn, contain keys and editable values.

This configuration is a only available with `userProfile`=”`Advance`”.

Abbreviations

This property allows configuring country-specific abbreviations.

Example:

```
"Abbreviations" : [ "Wien:Wien", "Freih:Frh", "LIMITED:LTD",
"INDUSTRIES:IND", "FOOTBALL:F", "OÖ:Oberösterreich", "haus:hs",
"Hauptbahnhof:Hbf", "Sankt%:St", "European+Economic+Interest+Group:EEIG",
 "Dekan:Dek", "BUILDING:BLD", "NÖ:Niederösterreich"]
```

`SacFromFile` in the JSON must be numeric. Postal codes are numeric for most countries and can be used to define the SacFromFile. For those countries where postal codes are not numeric, the Custom Dataset Builder requires a field that can provide a logical grouping.

Post_StreetType

This property allows configuring country-specific street types which are often written after the street names.

Example:

```
"Post_StreetTypes" : [
"CHAUSEE:chaussee,CHAUSSEE,CHAUSSEE.,CHAUSSE,CHAUSSE.,CHAUSE,CH.,CHAUSS.,CHAUS.,CHAUS,CHAUSS",
 "PROM:PROM,promenade,Prom.", "WEG:WEG,weg,Weg.", "DAMM:DAM,damm,Damm.",
 "RING:RNG,ring,Ring.", "BOULEVARD:BD,boulevard,boulevard.,BD.",
"GASSE:GA,gasse,Gasse.,g.", "PLATZ:PL,platz,platz.,PL.",
"PROMENADE:PROM,promenade",
"STRAßE:STR,STRAßE,STRASSE,STRASS,STRASE,STRABE,STREET" ]
```

Pre_StreetTypes

This property allows configuring country-specific street types which often are written before the street names.

```
"Pre_StreetTypes" : [ "Rue:R." ]
```

The following shows the `advancedConfigs` properties:

```
"advancedConfigs" : {
          "Post_StreetTypes" : [
"CHAUSEE:chaussee,CHAUSSEE,CHAUSSEE.,CHAUSSE,CHAUSSE.,CHAUSE,CH.,CHAUSS.,CHAUS.,CHAUS,CHAUSS",
 "PROM:PROM,promenade,Prom.", "WEG:WEG,weg,Weg.", "DAMM:DAM,damm,Damm.",
 "RING:RNG,ring,Ring.", "BOULEVARD:BD,boulevard,boulevard.,BD.",
"GASSE:GA,gasse,Gasse.,g.", "PLATZ:PL,platz,platz.,PL.",
"PROMENADE:PROM,promenade",
"STRAßE:STR,STRAßE,STRASSE,STRASS,STRASE,STRABE,STREET" ],
          "Abbreviations" : [ "Wien:Wien", "Freih:Frh", "LIMITED:LTD",
 "INDUSTRIES:IND", "FOOTBALL:F", "OÖ:Oberösterreich", "haus:hs",
"Hauptbahnhof:Hbf", "Sankt%:St", "European+Economic+Interest+Group:EEIG",
 "Dekan:Dek", "BUILDING:BLD", "NÖ:Niederösterreich" ],
          "Pre_StreetTypes" : [ "Rue:R." ]
```

How to Access User-defined Fields

Additional fields can be mapped in the configuration JSON and made available while geocoding through the Global Geocoding Module.

To access user-defined fields:

1. In Enterprise Designer, create a dataflow using the GlobalGeocode stage.
2. In the Write to File Options under the Fields tab, add the field using the Add button. Be sure to use the same name as defined in the JSON.
3. Save the dataflow and geocode the address. You will see the user-defined field in the output.

Supported Countries for Custom Dataset Builder

Country	ISO Country Code
Albania	ALB
Algeria	DZA
Angola	AGO
Argentina	ARG
Aruba	ABW
Australia	AUS
Austria	AUT
Bahamas	BHS
Bahrain	BHR
Barbados	BRB
Belarus	BLR
Belgium, Luxembourg	BEL
Belize	BLZ

Country	ISO Country Code
Benin	BEN
Bermuda	BMU
Bolivia	BOL
Bosnia and Herzegovina	BIH
Botswana	BWA
Brazil	BRA
Brunei Darussalam	BRN
Bulgaria	BGR
Burkina Faso	BFA
Burundi	BDI
Cameroon	CMR
Canada	CAN
Chile	CHL
China	CHN
Colombia	COL
Congo-Brazzaville	COG
Congo-Kinshasa	COD
Costa-Rica	CRI
Croatia	HRV
Cuba	CUB

Country	ISO Country Code
Cyprus	CYP
Czech Republic	CZE
Denmark	DNK
Dominican Republic	DOM
Ecuador	ECU
Egypt	EGY
El Salvador	SLV
Estonia	EST
Finland	FIN
France, French Guiana, Guadeloupe, Martinique, Mayotte, Monaco, Réunion	FRA, GLP, GUF, MCO, MTQ, MYT, REU
Gabon	GAB
Germany	DEU
Ghana	GHA
Greece	GRC
Guatemala	GTM
Guyana	GUY
Honduras	HND
Hong Kong	HKG
Hungary	HUN

Country	ISO Country Code
Iceland	ISL
India	IND
Indonesia	IDN
Iraq	IRQ
Ireland	IRL
Italy, Vatican City, San Marino	ITA, VAT, SMR
Jamaica	JAM
Japan	JPN
Jordan	JOR
Kenya	KEN
Korea	KOR
Kosovo	XKX
Kuwait	KWT
Latvia	LVA
Lebanon	LBN
Lesotho	LSO
Lithuania	LTU
Macau	MAC
Macedonia	MKD
Malawi	MWI

Country	ISO Country Code
Malaysia	MYS
Mali	MLI
Malta	MLT
Mauritania	MRT
Mauritius	MUS
Mexico	MEX
Montenegro	MNE
Morocco	MAR
Mozambique	MOZ
Namibia	NAM
Netherlands	NLD
New Zealand	NZL
Nicaragua	NIC
Niger	NER
Nigeria	NGA
Norway	NOR
Oman	OMN
Panama	PAN
Paraguay	PRY
Peru	PER

Country	ISO Country Code
Philippines	PHL
Poland	POL
Portugal	PRT
Qatar	QAT
Romania	ROU
Russia	RUS
Rwanda	RWA
Saint Kitts and Nevis	KNA
Saudi Arabia	SAU
Senegal	SEN
Serbia	SRB
Singapore	SGP
Slovakia	SVK
Slovenia	SVN
South Africa	ZAF
Spain, Andorra, Gibraltor	ESP AND GIB
Suriname	SUR
Swaziland	SWZ
Sweden	SWE
Switzerland, Liechtenstein	CHE LIE

Country	ISO Country Code
Taiwan	TWN
Tanzania	TZA
Thailand	THA
Togo	TGO
Trinidad and Tobago	TTO
Tunisia	TUN
Turkey	TUR
Uganda	UGA
Ukraine	UKR
United Arab Emirates	ARE
United Kingdom	GBR
United States	USA
Uruguay	URY
Venezuela	VEN
Viet Nam	VNM
Yemen	YEM
Zambia	ZMB
Zimbabwe	ZWE

C - Result Codes

In this section

Match and Location Codes for USA

Match Codes

The geocoder returns match codes indicating the address portions that matched or did not match to the database.

If the geocoder cannot make a match, the match code begins with "E" and the remaining digits indicate why the address did not match. For a description of the "Ennn" codes, see **"Ennn" Match Codes for No Match** on page 177. The digits do not specifically refer to which address elements did not match, but rather why the address did not match.

The following table contains the match code values. For a description of the hex digits for the match codes, see **Definitions for 1st-3rd hex digit match code values** on page 173.

Code	Description
Ahhh	Same as Shhh, but indicates match to an alias name record or an alternate record.
Chh	The street address did not match, but the geocoder located a street segment based on the input ZIP Code or city.
D00	Matched to a small town with P.O. Box or General Delivery only.
Ghhh	Matched to an auxiliary file.
Hhhh	The house number was changed.
Jhhh	Matched to a user-defined dictionary.

Code	Description
Nxx	Matched to the nearest address. Used with reverse geocoding. The following are the only values for N: **NS0** Nearest street center match (nearest street segment interpolated) **NS1** Nearest unranged street segment **NP0** Nearest point address **NX0** Nearest intersection
P	Successful reverse APN lookup.
Qhhh	Matched to USPS range records with unique ZIP Codes. CASS rules prohibit altering an input ZIP if it matches a unique ZIP Code value.
Rhhh	Matched to a ranged address.
Shhh	Matched to USPS data. This is considered the best address match, because it matched directly against the USPS list of addresses. S is returned for a small number of addresses when the matched address has a blank ZIP + 4.
Thhh	Matched to a street segment record.
Uhhh	Matched to USPS data but cannot resolve the ZIP + 4 code without the firm name or other information. CASS mode returns an E023 (multiple match) error code.
Vhhh	Matched to MLD and DVDMLDR using Reverse PBKey Lookup. For match code values, see
Xhhh	Matched to an intersection of two streets, for example, "Clay St & Michigan Ave." The first hex digit refers to the last line information, the second hex digit refers to the first street in the intersection, and the third hex digit refers to the second street in the intersection. **Note:** The USPS does not allow intersections as a valid deliverable address.
Yhhh	Same as Xhhh, but an alias name record was used for one or both streets.

Code	Description
Z[1]	No address given, but verified the provided ZIP Code .

[1] Zh may be returned if `FIND_CORRECT_LASTLINE` is set to `true`.

Definitions for 1st-3rd hex digit match code values

The table below contains the description of the hex digits for the match code values.

Note: A typical match code contains up to 4 characters: a beginning alpha character followed by 2 or 3 hex digits. The third hex digit is only populated for intersection matches or as part of the Extended Match Code.

- For intersection matches, use the table below for the 3rd hex digit definitions.
- For Extended Match Code, see **Definitions for Extended Match Code (3rd hex digit values)** on page 175.

Code	In first hex position means:	In second and third hex position means:
0	No change in last line.	No change in address line.
1	ZIP Code changed.	Street type changed.
2	City changed.	Predirectional changed.
3	City and ZIP Code changed.	Street type and predirectional changed.
4	State changed.	Postdirectional changed.
5	State and ZIP Code changed.	Street type and postdirectional changed.
6	State and City changed.	Predirectional and postdirectional changed.
7	State, City, and ZIP Code changed.	Street type, predirectional, and postdirectional changed.
8	ZIP + 4 changed.	Street name changed.
9	ZIP and ZIP + 4 changed.	Street name and street type changed.

Code	In first hex position means:	In second and third hex position means:
A	City and ZIP + 4 changed.	Street name and predirectional changed.
B	City, ZIP, and ZIP + 4 changed.	Street name, street type, and predirectional changed.
C	State and ZIP + 4 changed.	Street name and postdirectional changed.
D	State, ZIP, and ZIP + 4 changed.	Street name, street type, and postdirectional changed.
E	State, City, and ZIP + 4 changed.	Street name, predirectional, and postdirectional changed.
F	State, City, ZIP, and ZIP + 4 changed.	Street name, street type, predirectional, and postdirectional changed.

Definitions for Extended Match Code (3rd hex digit values)

Extended Match Codes return additional information about any changes in the house number, unit number and unit type fields in the matched address, as well as whether there was address information that was ignored. This additional information is provided in a 3rd hex digit that is appended to match codes for address-level matches only - A, G, H, J, Q, R, S, T or U (see **Match Codes** on page 170).

> **Note:** A typical match code contains up to 4 characters: a beginning alpha character followed by 2 or 3 hex digits. The third hex digit is only populated for intersection matches or as part of the Extended Match Code.

"Address information ignored" is specified when any of these conditions apply:

- The input address is a dual address (two complete addresses in the input address). For example, "4750 Walnut St. P.O Box 50".
- The input last line has extra information that is not a city, state or ZIP Code, and is ignored. For example, "Boulder, CO 80301 USA", where "USA" is ignored when matching.

For more information, see **Extended Match Codes** on page 141.

The table below provides the descriptions for the Extended Match Code 3rd hex digit return values:

Code	In 3rd hex position means:
0	Matched on all address information on line, including Unit Number and Unit Type if included.
1	Matched on Unit Number and Unit Type if included. Extra information on address line ignored. Extra information not considered for matching is not returned.
2	Matched on Unit Number. Unit Type changed.
3	Matched on Unit Number. Unit Type changed. Extra information on address line ignored. Extra information not considered for matching is not returned.
4	Unit Number changed or ignored.
5	Unit Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
6	Unit Number changed or ignored. Unit Type changed or ignored.
7	Unit Number changed or ignored. Unit Type changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
8	Matched on Unit Number and Unit Type if included. House Number changed or ignored.

Code	In 3rd hex position means:
9	Matched on Unit Number and Unit Type if included. House Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
A	Matched on Unit Number. Unit Type changed. House Number changed or ignored.
B	Matched on Unit Number. Unit Type changed. House Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
C	House Number changed or ignored. Unit Number changed or ignored.
D	House Number changed or ignored. Unit Number changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.
E	House Number changed or ignored. Unit Number changed or ignored. Unit Type changed or ignored.
F	House Number changed or ignored. Unit Number changed or ignored. Unit Type changed or ignored. Extra information on address line ignored. Extra information not considered for matching is not returned.

"Ennn" Match Codes for No Match

The following table describes the values returned when the application cannot find a match or an error occurs.

Code	"nnn" Value	Description
Ennn[2]		Indicates an error, or no match. This can occur when the address entered does not exist in the database, or the address is badly formed and cannot be parsed correctly. The last three digits of an error code indicate which parts of an address the application could not match to the database.
	nnn = 000	No match made.
	nnn = 001	Low level error.
	nnn = 002	Could not find data file.
	nnn = 003	Incorrect GSD file signature or version ID.
	nnn = 004	GSD file out of date. Only occurs in CASS mode.
	nnn = 010	No city and state or ZIP Code found.
	nnn = 011	Input ZIP not in the directory.
	nnn = 012	Input city not in the directory.
	nnn = 013	Input city not unique in the directory.
	nnn = 014	Out of licensed area. Only occurs if using Group1 licensing technology.
	nnn = 015	Record count is depleted and license has expired.
	nnn = 020	No matching streets found in directory.
	nnn = 021	No matching cross streets for an intersection match.
	nnn = 022	No matching segments.

Code	"nnn" Value	Description
	`nnn = 023`	Unresolved match.
	`nnn = 024`	No matching segments. (Same as 022.)
	`nnn = 025`	Too many possible cross streets for intersection matching.
	`nnn = 026`	No address found when attempting a multiline match.
	`nnn = 027`	Invalid directional attempted.
	`nnn = 028`	Record also matched EWS data, therefore the application denied the match.
	`nnn = 029`	No matching range, single street segment found.
	`nnn = 030`	No matching range, multiple street segments found.

Correct Lastline Match Codes

As mentioned in **Correct Lastline** on page 143, when set to `true`, `FIND_CORRECT_LASTLINE` corrects elements of the output lastline, providing a good ZIP Code or close match on the soundex even if the address would not match or was non-existent.

The feature works when `FIND_ADDRCODE` is `true` and the address does not match a candidate or when `FIND_Z_CODE` is `true` and only lastline information is input.

Code	Value	Description
`Zh`		No address given, but verified the provided ZIP Code.
	`h = 0`	No change in lastline.
	`h = 1`	ZIP Code changed.
	`h = 2`	City changed.

[2] Ennn may be returned if FIND_CORRECT_LASTLINE is set to `true`. For more information, see **Correct Lastline Match Codes** on page 178.

Code	Value	Description
	`h = 3`	City and ZIP Code changed.
	`h = 4`	State changed.
	`h = 5`	State and ZIP Code changed.
	`h = 6`	State and City changed.
	`h = 7`	State, City, and ZIP Code changed.
	`h = 8`	ZIP + 4 changed.
	`h = 9`	ZIP and ZIP + 4 changed.
	`h = A`	City and ZIP + 4 changed.
	`h = B`	City, ZIP, and ZIP + 4 changed.
	`h = C`	State and ZIP + 4 changed.
	`h = D`	State, ZIP, and ZIP + 4 changed.
	`h = E`	State, City, and ZIP + 4 changed.
`Ehnn`		Indicates an error, or no match. This can occur when the address entered does not exist in the database, or the address is badly formed and cannot be parsed correctly. The second digit of the error code is a hex digit which details the changes that were made to the last line information to correct the lastline. The last two digits of an error code indicate which parts of an address the application could not match to the database.
	`h = 0`	No change in lastline.
	`h = 1`	ZIP Code changed.
	`h = 2`	City changed.
	`h = 3`	Record also matched EWS data, therefore the application denied the match.

Code	Value	Description
	`h = 4`	State changed.
	`h = 5`	State and ZIP Code changed.
	`h = 6`	State and City changed.
	`h = 7`	State, City, and ZIP Code changed.
	`h = 8`	ZIP + 4 changed.
	`h = 9`	ZIP and ZIP + 4 changed.
	`h = A`	City and ZIP + 4 changed.
	`h = B`	City, ZIP, and ZIP + 4 changed.
	`h = C`	State and ZIP + 4 changed.
	`h = D`	State, ZIP, and ZIP + 4 changed.
	`h = E`	State, City, and ZIP + 4 changed.
	`nn = 00`	No match made.
	`nn = 01`	Low level error.
	`nn = 02`	Could not find data file.
	`nn = 03`	Incorrect GSD file signature or version ID.
	`nn = 04`	GSD file out of date. Only occurs in CASS mode.
	`nn = 10`	No city and state or ZIP Code found.
	`nn = 11`	Input ZIP Code not in the directory.
	`nn = 12`	Input city not in the directory.

Code	Value	Description
	nn = 13	Input city not unique in the directory.
	nn = 14	Out of licensed area. Only occurs if using Group1 licensing technology.
	nn = 15	Record count is depleted and license has expired.
	nn = 20	No matching streets found in directory.
	nn = 21	No matching cross streets for an intersection match.
	nn = 22	No matching segments.
	nn = 23	Unresolved match.
	nn = 24	No matching segments. (Same as 022.)
	nn = 25	Too many possible cross streets for intersection matching.
	nn = 26	No address found when attempting a multiline match.
	nn = 27	Invalid directional attempted.
	nn = 28	Record also matched EWS data, therefore the application denied the match.
	nn = 29	No matching range, single street segment found
	nn = 30	No matching range, multiple street segments found

Location Codes

Location codes indicate the locational accuracy of the assigned geocode. Note that an accurately placed candidate is not necessarily an ideal candidate. Examine the match codes and/or result codes in addition to location codes to best evaluate the overall quality of the candidate.

Address Location Codes

Location codes that begin with an "A" are address location codes. Address location codes indicate a geocode made directly to a street network segment (or two segments, in the case of an intersection).

An address location code has the following characters.

1st character	Always an "A" indicating an address location.	
2nd character	May be one of the following	
	C	Interpolated address point location
	G	Auxiliary file data location
	I	Application infers the correct segment from the candidate records
	P	Point-level data location
	R	Location represents a ranged address
	S	Location on a street range
	X	Location on an intersection of two streets
3rd and 4th character	Digit indicating other qualities about the location.	

Address Location Code Descriptions

Code		Description
AGn		Indicates an auxiliary file for a geocode match where "n" is one of the following values:
	n = 0	The geocode represents the center of a parcel or building.
	n = 1	The geocode is an interpolated address along a segment.
	n = 2	The geocode is an interpolated address along a segment, and the side of the street cannot be determined from the data provided in the auxiliary file record.
	n = 3	The geocode is the midpoint of the street segment.
APnn		Indicates a point-level geocode match representing the center of a parcel or building, where "nn" is one of the following values:
	nn = 00	User Dictionary centroid. Geocode returned by a User Dictionary.
	nn = 02	Parcel centroid Indicates the center of an accessor's parcel (tract or lot) polygon. When the center of an irregularly shaped parcel falls outside of its polygon, the centroid is manually repositioned to fall inside the polygon as closely as possible to the actual center.
	nn = 04	Address points Represents field-collected GPS points with field-collected address data.

Code	Description
nn = 05	Structure point Indicates a location within a building footprint polygon that is associated with the matched address. Usually, residential addresses consist of a single building. For houses with outbuildings (detached garages, sheds, barns, etc.), the structure point will typically fall on the primary structure. Condominiums and duplexes have multiple, individual addresses and may have multiple structure points for each building. Multi-unit buildings are typically represented by a single structure point associated with the primary/base address, rather than discrete structure points for each unit. Shopping malls, industrial complexes, and academic or medical center campuses are commonly represented by a single structure point associated with the primary/base address for the entire complex. When multiple addresses are assigned to multiple buildings within one complex, multiple structure points may be represented within the same complex.
nn = 07	Manually placed Address points are manually placed to coincide with the midpoint of a parcel's street frontage at a distance from the center line.
nn = 08	Front door point Represents the designated primary entrance to a building. If a building has multiple entrances and there is no designated primary entrance or the primary entrance cannot readily be determined, the primary entrance is chosen based on proximity to the main access street and availability of parking.
nn = 09	Driveway offset point Represents a point located on the primary access road (most commonly a driveway) at a perpendicular distance of between 33-98 feet (10-30 meters) from the main roadway.

Code	Description
nn = 10	Street access point Represents the primary point of access from the street network. This address point type is located where the driveway or other access road intersects the main roadway.
nn = 21	Base parcel point When unable to match to an input unit number, or when the unit number is missing from an address location with multiple units, the "base" parcel information is returned, the address is not standardized to a unit number, and additional information, such as an Assessor's Parcel Number, is not returned.
nn = 22	Backfill address point The precise parcel centroid is unknown. The address location assigned is based on two known parcel centroids.
nn = 23	Virtual address point The precise parcel centroid is unknown. The address location assigned is relative to a known parcel centroid and a street segment end point.
nn = 24	Interpolated address point The precise parcel centroid is unknown. The address location assigned is based on street segment end points.
AIn	The correct segment is inferred from the candidate records at match time.
ASn	House range address geocode. This is the most accurate geocode available.
AIn and ASn, and ACnh share the same values for the 3rd character "n" as follows:	
n = 0	Best location.

Code		Description
	`n = 1`	Street side is unknown. The Census FIPS Block ID is assigned from the left side; however, there is no assigned offset and the point is placed directly on the street.
	`n = 2`	Indicates one or both of the following: • The address is interpolated onto a TIGER segment that did not initially contain address ranges. • The original segment name changed to match the USPS spelling. This specifically refers to street type, predirectional, and postdirectional. **Note:** Only the second case is valid for non-TIGER data because segment range interpolation is only completed for TIGER data.
	`n = 3`	Both 1 and 2.
	`n = 7`	Placeholder. Used when starting and ending points of segments contain the same value and shape data is not available.
`ACnh`		Indicates a point-level geocode that is interpolated between 2 parcel centroids (points), a parcel centroid and a street segment endpoint, or 2 street segment endpoints.
	The `ACnh` 4th character "h" values are as follows:	
	`h = 0`	Represents the interpolation between two points, both coming from User Dictionaries.
	`h = 1`	Represents the interpolation between two points. The low boundary came from a User Dictionary and the high boundary, from a non-User Dictionary.
	`h = 2`	Represents the interpolation between one point and one street segment end point, both coming from User Dictionaries.

Code	Description
h = 3	Represents the interpolation between one point (low boundary) and one street segment end point (high boundary). The low boundary came from a User Dictionary and the high boundary from a non-User Dictionary.
h = 4	Represents the interpolation between two points. The low boundary came from a non-User Dictionary and the high boundary from a User Dictionary.
h = 5	Represents the interpolation between two points, both coming from non-User Dictionaries.
h = 6	Represents the interpolation between one point (low boundary) and one street segment end point (high boundary). The low boundary came from a non-User Dictionary and the high boundary from a User Dictionary.
h = 7	Represents the interpolation between one point and one street segment end point and both came from non-User Dictionaries.
h = 8	Represents the interpolation between one street segment end point andone point, both coming from User Dictionaries.
h = 9	Represents the interpolation between one street segment end point (low boundary) andone point (high boundary). The low boundary came from a User Dictionary and the high boundary from a non-User Dictionary.
h = A	Represents the interpolation between two street segment end points, both coming from User Dictionaries.
h = B	Represents the interpolation between two street segment end points. The low boundary came from a User Dictionary and the high boundary from a non-User Dictionary.

Code		Description
	h = C	Represents the interpolation between one street segment end point (low boundary) and one point (high boundary). The low boundary came from a non-User Dictionary and the high boundary from a User Dictionary.
	h = D	Represents the interpolation between one street segment end point and one point, both coming from non-User Dictionary.
	h = E	Represents the interpolation between two street segment end points. The low boundary came from a non-User Dictionary and the high boundary from a User Dictionary.
	h = F	Represents the interpolation between two street segment end points, both coming from non-User Dictionaries.
ARn		Ranged address geocode, where "n" is one of the following:
	n = 1	The geocode is placed along a single street segment, midway between the interpolated location of the first and second input house numbers in the range.
	n = 2	The geocode is placed along a single street segment, midway between the interpolated location of the first and second input house numbers in the range, and the side of the street is unknown. The Census FIPS Block ID is assigned from the left side; however, there is no assigned offset and the point is placed directly on the street.
	n = 4	The input range spans multiple USPS segments. The geocode is placed on the endpoint of the segment which corresponds to the first input house number, closest to the end nearest the second input house number.
	n = 7	Placeholder. Used when the starting and ending points of the matched segment contain the same value and shape data is not available.

Code	Description
AXn	Intersection geocode, where "n" is one of the following:
n = 3	Standard single-point intersection computed from the center lines of street segments.
n = 8	Interpolated (divided-road) intersection geocode. Attempts to return a centroid for the intersection.

Street Centroid Location Codes

Location codes that begin with "C" are street centroid location codes. Street centroid location codes indicate the Census ID accuracy and the position of the geocode on the returned street segment. Street centroids may be returned if the street centroid fallback option is enabled and an address-level geocode could not be determined.

A street centroid location code has the following characters.

1st character	Always "C" indicating a location derived from a street segment.
2nd character	Census ID accuracy based on the search area used to obtain matching Street Segment.
3rd character	Location of geocode on the returned street segment.

The following table contains the values and descriptions for the location codes.

Character position	Code	Description
2nd Character		
	B	Block Group accuracy (most accurate). Based on input ZIP Code.
	T	Census Tract accuracy. Based on input ZIP Code.
	C	Unclassified Census accuracy. Normally accurate to at least the County level. Based on input ZIP Code.
	F	Unknown Census accuracy. Based on Finance area.
	P	Unknown Census accuracy. Based on input City.
3rd Character		
	C	Segment Centroid.

Character position	Code	Description
	L	Segment low-range end point.
	H	Segment high-range end point.

ZIP + 4 Centroid Location Codes

Location codes that begin with a "Z" are ZIP + 4 centroid location codes. ZIP + 4 centroids indicate a geocode could not be determined for the address, so the location of the center of the address's ZIP + 4 was returned instead. ZIP + 4 centroid location codes indicate the quality of two location attributes: Census ID accuracy and positional accuracy.

A ZIP + 4 centroid location code has the following characters.

1st character	Always "Z" indicating a location derived from a ZIP centroid.
2nd character	Census ID accuracy.
3rd character	Location type.
4th character	How the location and Census ID was defined. Provided for completeness, but may not be useful for most applications.

Character Position	Code	Description
2nd Character		
	B	Block Group accuracy (most accurate).
	T	Census Tract accuracy.
	C	Unclassified Census accuracy. Normally accurate to at least the County level.
3rd Character		
	5	Location of the Post Office that delivers mail to the address, a 5-digit ZIP Code centroid, or a location based upon locale (city). See the 4th character for a precise indication of locational accuracy.

Character Position	Code	Description
	7	Location based upon a ZIP + 2 centroid. These locations can represent a multiple block area in urban locations, or a slightly larger area in rural settings.
	9	Location based upon a ZIP + 4 centroid. These are the most accurate centroids and normally place the location on the correct block face. For a small number of records, the location may be the middle of the entire street on which the ZIP + 4 falls. See the 4th character for a precise indication of locational accuracy.
4th Character		
	A	Address matched to a single segment. Location assigned in the middle of the matched street segment, offset to the proper side of the street.
	a	Address matched to a single segment, but the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	B	Address matched to multiple segments, all segments have the same Block Group. Location assigned to the middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	b	Same as methodology "B" except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.

Character Position	Code	Description
	C	Address matched to multiple segments, with all segments having the same Census Tract. Returns the Block Group representing the most households in this ZIP + 4. Location assigned to t he middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	c	Same as methodology "C" except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	D	Address matched to multiple segments, with all segments having the same County. Returns the Block Group representing the most households in this ZIP + 4. Location assigned to the middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	d	Same as methodology "D" except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	E	Street name matched; no house ranges available. All matched segments have the same Block Group. Location placed on the segment closest to the center of the matched segments. In most cases, this is on the mid-point of the entire street.

Character Position	Code	Description
	F	Street name matched; no house ranges available. All matched segments have the same Census Tract. Location placed on the segment closest to the center of the matched segments. In most cases, this is on the mid-point of the entire street.
	G	Street name matched (no house ranges available). All matched segments have the same County. Location placed on the segment closest to the center of the matched segments. In most cases, this is on the mid-point of the entire street.
	H	Same as methodology "G", but some segments are not in the same County. Used for less than .05% of the centroids.
	I	Created ZIP + 2 cluster centroid as defined by methodologies "A", "a", "B", and "b". All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.
	J	Created ZIP + 2 cluster centroid as defined by methodologies "A", "a", "B", "b", "C", and "c". All centroids in this ZIP + 2 cluster have the same Census Tract. Location assigned to the ZIP + 2 centroid.
	K	Created ZIP + 2 cluster centroid as defined by methodologies "A", "a", "B", "b", "C", "c", "D", and "d". Location assigned to the ZIP + 2 centroid.
	L	Created ZIP + 2 cluster centroid as defined by methodology "E". All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.

Character Position	Code	Description
	M	Created ZIP+2 cluster centroid as defined by methodologies "E" and "F". All centroids in this ZIP + 2 cluster have the same Census Tract. Location assigned to the ZIP + 2 centroid.
	N	Created ZIP + 2 cluster centroid as defined by methodologies "E", "F", "G", and "H". Location assigned to the ZIP + 2 centroid.
	O	ZIP Code is obsolete and not currently used by the USPS. Historic location assigned.
	V	Over 95% of addresses in this ZIP Code are in a single Census Tract. Location assigned to the ZIP Code centroid.
	W	Over 80% of addresses in this ZIP Code are in a single Census Tract. Reasonable Census Tract accuracy. Location assigned to the ZIP Code centroid.
	X	Less than 80% of addresses in this ZIP Code are in a single Census Tract. Census ID is uncertain. Location assigned to the ZIP Code centroid.
	Y	Rural or sparsely populated area. Census code is uncertain. Location based upon the USGS places file.
	Z	P.O. Box or General Delivery addresses. Census code is uncertain. Location based upon the Post Office location that delivers the mail to that address.

Geographic Centroid Location Codes

Location codes that begin with "G" are geographic centroid location codes. Geographic centroids may be returned if the street centroid fallback option is enabled and an address-level geocode could not be determined. Geographic centroid location codes indicate the quality a city, county, or state centroid.

A geographic centroid location code has the following characters.

1st character	Always "G" indicating a location derived from a geographic centroid.
2nd character	Geographic area type. One of the following: **M** Municipality (for example, a city) **C** County **S** State

Global Result Codes

Forward Geocoding Result Codes

Result Code General Descriptions

The following table provides general descriptions for the returned result codes.

Result Code	Description
Street level geocoded candidates return a result code beginning with the letter **S**. The second character in the code indicates the positional accuracy of the resulting point for the geocoded record. For information on the specific S result codes supported for your country, see **Single Match 'S' Result Codes** on page 202.	
S8	Single match with the point located at either the single point associated with an address point candidate or at an address point candidate that shares the same house number. No interpolation is required.
S7	Single match with the point located at an interpolated point along a street segment. Both a point dictionary and a street segment dictionary must be available. Because known point data is available, the S7 interpolation is more accurate than an S5 result.
S6	Single match, point located at point ZIP centroid.
S5	Single match with the point located at a street address position. Because only street segment data is available, the interpolation is not as accurate as an S7 return The S5 code is followed by letters and dashes indicating match precision.
S4	Single match with the point located at a street centroid.
S3	Single match with the point located at a ZIP + 4® centroid. This is the same quality match as a Z3 result.
S2	Single match with the point located at a ZIP + 2 centroid. This is the same quality match as a Z2 result.
S1	Single match with the point located at a ZIP Code centroid. This is the same quality match as a Z1 result.

Result Code	Description
S0	Single match, however, no coordinates are available (this is a very rare occurrence).
SX	Single match with the point located at street intersection.
SC	Single match where the original point has been moved a specified distance (usually along a perpendicular line) toward or away from the associated street segment. This result code can be returned only when both a point dictionary and a street segment dictionary are available and when the centerline offset feature is used.
SL	India only. A street level match at the sublocality (block or sector) level. ublocality. An SL result code also requires a match on other geographic input fields (city, district, or state).

Result Code	Description

For **s** (street geocoded) result codes, eight additional characters describe how closely the address matches an address in the database. The characters appear in the order listed in the following table. Any non-matched components are represented by a dash.

For example, the result code S5--N-SCZA represents a single match that matched the street name, street post directional, town and postcode. The dashes indicate that there was no match on house number, street prefix direction, or thoroughfare type. The match came from the Street Range Address database. This record would be geocoded at the street address position of the match candidate.

H	House number match.
P	Street prefix (pre-directional). P is present if any of these conditions are satisfied: • The candidate pre-directional matches the input pre-directional. • The candidate post-directional matches the input pre-directional after pre- and post-directionals are swapped. • The input does not have a pre-directional.
N	Street name match.
T	Street/thoroughfare type match.

Result Code	Description
S	Street post-directional S in result code is present if any of these conditions are satisfied: • The candidate post-directional matches the input post-directional. • The candidate pre-directional matches the input post-directional after pre- and post-directionals are swapped. • The input does not have a post-directional.
C	areaName3 match (this is usually the city or town).
Z	Postal code match.
A or U	Match to Address Dictionary or User Dictionary.

Result Code	Description

Matches in the **z** category indicate that a match was made at the postcode level. A postcode match is returned in either of these cases:

- You specified to match to postal code centroids. The resulting point is located at the postal code centroid with the following possible accuracy levels.
- There is no street level match and you specified to fall back to postal code centroid.

Note: Refer to the section covering your country to locate the specific meanings of postCode1 & 2.

Z6	Z6 results are matched to a point ZIP centroid. Point ZIPs are 5-digit The Z6 code indicates that these special ZIPs are actual point locations, not an area. Point ZIPs include unique single sites, buildings, or organizations.
Z3	Z3 results are matched to ZIP + 4 or postCode2 centroid locations.
Z2	Z2 results are matched to ZIP + 2 or partial postCode2 centroid locations.
Z1	Z1 results are matched to ZIP Code or (postCode1) centroid locations.

Geographic level geocoded candidates return a result code beginning with the letter **G**. The numbers following the G in the result code provides more detailed information on the accuracy of the candidate.

Note: Refer to the section covering your country to locate the specific meanings of areaName1-4.

Result Code	Description
G1	State/Province (areaName1) match with the point located at the state centroid.
G2	County/Region (areaName2) match with the point located at the county centroid.
G3	City/Town (areaName3) match with the point located at the city centroid.
G4	Suburb/village (areaName4) match with the point located at the suburb/village centroid.

Single Match 'S' Result Codes

The following table shows the support for the S category result codes by country. For detailed descriptions of the 'S' result codes, see **Forward Geocoding Result Codes** on page 198. These descriptions apply to the vast majority of the countries. The exceptions are listed and described in the sections below the following table for:

- **Australia**
- **Canada**
- **United States**

A bullet "•" indicates the S code is supported. A blank cell indicates the S code is not supported.

Country Name	S8	S7	S6	S5	S4	S3	S2	S1	S0	SX	SC	SG
Australia (AUS)	•	•		•	•				•		•	•
Canada (CAN)	•	•		•	•	•		•	•		•	
Denmark (DNK)	•	•		•	•					•		
Germany (DEU)	•	•		•	•					•		
Great Britain (GBR)	•	•		•	•				•	•		
New Zealand (NZL)	•	•		•	•					•		
United States (USA)	•	•	•	•	•	•	•	•	•	•	•	
All other countries	•	•		•	•				•	•	•	

United States — 'S' Precision Code Descriptions

The following table provides 'S' precision code descriptions for the USA.

Precision Code	Description
Street level geocoded candidates return a Precision Code beginning with the letter S. The second character in the code indicates the positional accuracy of the resulting point for the geocoded record.	
S8	Single match, point located at either the single point associated with an address point candidate or at an address point candidate that shares the same house number. No interpolation is required.
S7	Single match, located at an interpolated point along a street segment. Both a point/parcel dictionary and a street segment dictionary must be available. Because known point data is available, the S7 interpolation is more accurate than an S5 result.
S6	Single match, point located at point ZIP centroid.
S5	Single match, point located at a street address position. Because only street segment data is available, the interpolation is not as accurate as an S7 return.
S4	Single match, point located at a street centroid.
S3	Single match, point located at ZIP + 4®. This is the same quality match as a Z3 result.
S2	Single match, point located at ZIP + 2 centroid. single match, point located at ZIP + 2 centroid. This is the same quality match as a Z2 result.
S1	Single match, point located at ZIP Code centroid. This is the same quality match as a Z1 result.
S0	Single match, however, no coordinates are available (this is a very rare occurrence).
SX	Single match, point located at street intersection.
SC	Single match where the original point has been moved a specified distance (usually along a perpendicular line) toward or away from the associated street segment. This result code can be returned only when both a point geocoding dataset and a street segment geocoding dataset are available and when the centerline offset feature is used.

Australia — 'S' Result Code Descriptions

The following table provides 'S' result code descriptions for Australia.

Result Code	Description
Street level geocoded candidates return a result code beginning with the letter S. The second character in the code indicates the positional accuracy of the resulting point for the geocoded record.	
S8	Single match, point located at either the single point associated with an address point candidate or at an address point candidate that shares the same house number. No interpolation is required.
S8.......G	The S8.......G result code is used for single matches with GNAF Reliability levels of 1or 2 (the highest level of GNAF Reliability.
S7	Single match, located at an interpolated point along the candidate's street segment. When the potential candidate is not an address point candidate and there are no exact house number matches among other address point candidates, the S7 result is returned using address point interpolation.
S7.......G	The S7.......G result code is used for single matches with GNAF Reliability level of 3.
S5	Single match, point located at a street address position.
S4	Single match, point located at the center of a shape point path (shape points define the shape of the street polyline).
S4.......G	The S4.......G result code is used for single matches with a GNAF Reliability level of 4 (associated with a unique road feature.)
S0	Single match, however, no coordinates are available (this is a very rare occurrence).
SX	Single match with the point located at street intersection.
SC	Single match where the original point has been moved a specified distance (usually along a perpendicular line) toward or away from the associated street segment. This result code can be returned only when both a point geocoding dataset and a street segment geocoding dataset are available and when the centerline offset feature is used.
SG	Single match with point at the centre of a locality (areaName3) or Locality level geocode derived from topographic feature. An SG result code is associated with GNAF Reliability Level 5 (locality or neighbourhood) or with Level 6 (unique region.)

Canada — 'S' Result Code Descriptions

The following table provides 'S' result code descriptions for Canada.

Result Code	Description
Street level geocoded candidates return a result code beginning with the letter S. The second character in the code indicates the positional accuracy of the resulting point for the geocoded record.	
S8	Single match, point located at either the single point associated with an address point candidate or at an address point candidate that shares the same house number. No interpolation is required.
S7	Single match, located at an interpolated point along the candidate's street segment. When the potential candidate is not an address point candidate and there are no exact house number matches among other address point candidates, the S7 result is returned using address point interpolation.
S5	Single match, point located at a street address position.
S4	Single match, point located at the center of a shape point path (shape points define the shape of the street polyline).
S3	Single match, point located at postal centroid of FSALDU
S1	Single match, point located at postal centroid of FSA
S0	Single match, however, no coordinates are available (this is a very rare occurrence).
SC	Single match where the original point has been moved a specified distance (usually along a perpendicular line) toward or away from the associated street segment. This result code can be returned only when both a point geocoding dataset and a street segment geocoding dataset are available and when the centerline offset feature is used.

Reverse Geocoding 'R' Result Codes

Matches in the `R` category indicate that the record was matched by reverse geocoding. The first three characters of the `R` result code indicate the type of match found. `R` geocode results include an additional letter to indicate the dictionary from which the match was made. This is always an `A`, indicating address dictionary; reverse geocoding is supported by the address dictionary only (not user dictionaries.)

Reverse Geocoding 'R' Result Code Descriptions

Reverse Geocoding Code	Description
`RG0`	Geographic level: Country level (typically only used for small island countries where no other administrative divisions are in the data).
`RG1`	Geographic level: State or province level. Corresponds to G1 in forward geocoding.
`RG2`	Geographic level: District (state or province subdivision) level. Corresponds to G2 in forward geocoding.
`RG3`	Geographic level: City or town level. Corresponds to G3 in forward geocoding.
`RG4`	Geographic level: Locality (city/town subdivision). Corresponds to G4 in forward geocoding.
`RG5`	Geographic level: Locality subdivision.
`RS4A`	Street centroid candidate for reverse geocoding. Candidate returned from address dictionary.
`RS5A`	Interpolated street candidate for reverse geocoding. Candidate returned from address dictionary.

Reverse Geocoding Code	Description
RS7G	For Australia only: Candidate returned from Australia GNAF dictionary with GNAF Reliability level of 3.
RS8A	Point/parcel level precision for reverse geocoding. Candidate returned from address dictionary.
RS8G	For Australia only: Point/parcel level precision. Candidate returned from Australia GNAF dictionary with GNAF Reliability level of 1 or 2.
RZ	Postal level: A postal level reverse geocode in World Boundary Reverse returns a precision code of “RZ”.

D - ISO 3166-1 Country Codes

In this section

ISO 3166-1 Country Codes

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
ALBANIA	AL	ALB
ALGERIA	DZ	DZA
AMERICAN SAMOA	US	USA
ANDORRA	AD	AND
ANGOLA	AO	AGO
ARGENTINA	AR	ARG
ARUBA	AW	ABW
AUSTRALIA	AU	AUS
AUSTRIA	AT	AUT
BAHAMAS	BS	BHS
BAHRAIN	BH	BHR
BARBADOS	BB	BRB
BELARUS	BY	BLR
BELGIUM	BE	BEL
BELIZE	BZ	BLZ
BENIN	BJ	BEN
BERMUDA	BM	BMU

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
BOLIVIA	BO	BOL
BOSNIA AND HERZEGOVINA	BA	BIH
BOTSWANA	BW	BWA
BRAZIL	BR	BRA
BRUNEI DARUSSALAM	BN	BRN
BULGARIA	BG	BGR
BURKINA FASO	BF	BFA
BURUNDI	BI	BDI
CAMEROON	CM	CMR
CANADA	CA	CAN
CHILE	CL	CHL
CHINA	CN	CHN
COLOMBIA	CO	COL
CONGO	CG	COG
CONGO, DEMOCRATIC REPUBLIC OF THE	CD	COD
COSTA RICA	CR	CRI
CROATIA (LOCAL NAME: HRVATSKA)	HR	HRV
CUBA	CU	CUB
CYPRUS	CY	CYP
CZECH REPUBLIC	CZ	CZE

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
DENMARK	DK	DNK
DOMINICAN REPUBLIC	DO	DOM
ECUADOR	EC	ECU
EGYPT	EG	EGY
EL SALVADOR	SV	SLV
ESTONIA	EE	EST
FINLAND	FI	FIN
FRANCE	FR	FRA
FRENCH GUYANA	GF	GUF
GABON	GA	GAB
GERMANY	DE	DEU
GHANA	GH	GHA
GREAT BRITAIN	GB	GBR
GREECE	GR	GRC
GUADELOUPE	GP	GLP
GUAM	US	USA
GUATEMALA	GT	GTM
GUYANA	GY	GUY
HONDURAS	HN	HND
HONG KONG	HK	HKG

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
HUNGARY	HU	HUN
ICELAND	IS	ISL
INDIA	IN	IND
INDONESIA	ID	IDN
IRAQ	IQ	IRQ
IRELAND	IE	IRL
ITALY	IT	ITA
JAMAICA	JM	JAM
JAPAN	JP	JPN
JORDAN	JO	JOR
KENYA	KE	KEN
KOREA, SOUTH	KR	KOR
KOSOVO	XK	XKX
KUWAIT	KW	KWT
LATVIA	LV	LVA
LEBANON	LB	LBN
LESOTHO	LS	LSO
LIECHTENSTEIN	LI	LIE
LITHUANIA	LT	LTU
LUXEMBOURG	LU	LUX

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
MACAO	MO	MAC
MACEDONIA, REPUBLIC OF	MKD	MKD
MALAWI	MW	MWI
MALAYSIA	MY	MYS
MALI	ML	MLI
MALTA	ML	MLT
MARTINIQUE	MQ	MTQ
MAURITANIA	MR	MRT
MAURITIUS	MU	MUS
MAYOTTE	YT	MYT
MEXICO	MX	MEX
MONACO	MC	MCO
MONTENEGRO	ME	MNE
MOROCCO	MA	MAR
MOZAMBIQUE	MZ	MOZ
NAMIBIA	NA	NAM
NETHERLANDS	NL	NLD
NEW ZEALAND	NZ	NZL
NICARAGUA	NI	NIC
NIGER	NE	NER

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
NIGERIA	NG	NGA
NORTH MARIANA ISLANDS	US	USA
NORWAY	NO	NOR
OMAN	OM	OMN
PALAU	US	USA
PANAMA	PA	PAN
PARAGUAY	PY	PRY
PERU	PE	PER
PHILIPPINES	PH	PHL
POLAND	PL	POL
PORTUGAL	PT	PRT
PUERTO RICO	US	USA
QATAR	QA	QAT
REUNION	RE	REU
ROMANIA	RO	ROU
RUSSIAN FEDERATION	RU	RUS
RWANDA	RW	RWA
SAINT KITTS AND NEVIS	KN	KNA
SAUDI ARABIA	SA	SAU
SENEGAL	SN	SEN

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
SERBIA	RS	SRB
SINGAPORE	SG	SGP
SLOVAKIA (SLOVAK REPUBLIC)	SK	SVK
SLOVENIA	SI	SVN
SOUTH AFRICA	ZA	ZAF
SPAIN	ES	ESP
SURINAME	SR	SUR
SWAZILAND	SZ	SWZ
SWEDEN	SE	SWE
SWITZERLAND	CH	CHE
TAIWAN	TW	TWN
TANZANIA	TZ	TZA
THAILAND	TH	THA
TOGO	TG	TGO
TRINIDAD AND TOBAGO	TT	TTO
TUNISIA	TN	TUN
TURKEY	TR	TUR
UGANDA	UG	UGA
UKRAINE	UA	UKR
UNITED ARAB EMIRATES	AE	ARE

Country Name	ISO 3166-1 Alpha-2 Country Code	ISO 3166-1 Alpha-3 Country Code
UNITED KINGDOM	GB	GBR
UNITED STATES	US	USA
URUGUAY	UY	URY
VENEZUELA	VE	VEN
VIETNAM	VN	VNM
VIRGIN ISLANDS	US	USA
WORLD GEOCODER	XW	XWG
YEMEN	YE	YEM
ZAMBIA	ZM	ZMB
ZIMBABWE	ZW	ZWE

E - Error Codes

In this section

Exception Codes

If the server throws an exception, the REST web service will return the exception code and an accompanying exception message over the network to the client. The exception code provides a general error description; the exception message provides a more specific indication of the cause of the exception.

In the following example a GET request to the Geocode service contains an incorrect geocodeType "a".

```
GET http://10.24.48.217:8082/Geocode/rest
/GeocodeService/geocode.json?mainAddress=
330%20Front%20St.%20W%20TORONTO%20ON%20M5V%203B7
&country=can&geocodeType=a HTTP/1.1
```

The server returns the following error:

```
HTTP/1.1 400 Bad Request
            Server: Apache-Coyote/1.1
            exceptionCode: INVALID_CLIENT_INPUT
            exceptionMsg: Invalid geocodeType value: A
            Date: Wed, 20 Sep 2017 14:33:03 GMT
            Content-Type: application/json
            Content-Length: 99
            Connection: close


{"errors":[{"errorCode":"INVALID_CLIENT_INPUT","errorDescription":"Invalid
 geocodeType value: A"}]}
```

Exception Codes (datatype = String)	Description
REQUIRED_PARAMETER_MISSING	A required parameter is missing.
DATA_NOT_LICENSED	The license file for an address dictionary is not installed.
INTERNAL_ERROR	A general error occurred with the geocoding engine.
MAPMARKER_EXCEPTION	A general exception occurred in the MapMarker geocoding engine.
MAPMARKER_FATAL_EXCEPTION	A fatal exception occurred in the MapMarker geocoding engine.

Exception Codes (datatype = String)	Description
INVALID_CLIENT_INPUT	An invalid input was encountered in the request.
NO_COUNTRY_SPECIFIED	The country field is missing from the request.
COUNTRY_NOT_SUPPORTED	The requested operation is not supported for the specified country.
GEOSTAN_FATAL_EXCEPTION	A fatal exception occurred in the GeoStan geocoding engine.

Notices

USPS® Notices

Pitney Bowes Inc. holds a non-exclusive license to publish and sell ZIP + 4® databases on optical and magnetic media. These trademarks are owned by the United States Postal Service: CASS, CASS Certified, DPV, eLOT, FASTforward, First-Class Mail, Intelligent Mail, LACS^{Link}, NCOA^{Link}, PAVE, PLANET Code, Postal Service, POSTNET, Post Office, RDI, Suite^{Link}, United States Postal Service, Standard Mail, United States Post Office, USPS, ZIP Code, and ZIP + 4. This list is not exhaustive of the trademarks belonging to the Postal Service.

Pitney Bowes Inc. is a non-exclusive licensee of USPS® for NCOA$^{Link®}$ processing.

Prices for Pitney Bowes products, options, and services are not established, controlled, or approved by USPS® or United States Government. When utilizing RDI™ data to determine parcel-shipping costs, the business decision on which parcel delivery company to use is not made by the USPS® or United States Government.

Data Provider and Related Notices

Data Products contained on this media and used within Pitney Bowes Software applications are protected by various trademarks and by one or more of these copyrights:

© Copyright United States Postal Service. All rights reserved.

© 2014 TomTom. All rights reserved. TomTom and the TomTom logo are registered trademarks of TomTom N.V.

© 2016 HERE Fuente: INEGI (Instituto Nacional de Estadística y Geografía) - Based upon electronic data © National Land Survey Sweden.

© Copyright United States Census Bureau

© Copyright Nova Marketing Group, Inc.

Portions of this program are © Copyright 1993-2019 by Nova Marketing Group Inc. All Rights Reserved

© Copyright Second Decimal, LLC

© Copyright Canada Post Corporation - Data is from a compilation in which Canada Post Corporation is the copyright owner.

© 2007 Claritas, Inc.

The Geocode Address World data set contains data licensed from the GeoNames Project (**www.geonames.org**) provided under the Creative Commons Attribution License ("Attribution License") located at **http://creativecommons.org/licenses/by/3.0/legalcode**. Your use of the GeoNames data (described in the Spectrum™ Technology Platform User Manual) is governed by the terms of the Attribution License, and any conflict between your agreement with Pitney Bowes and the Attribution License will be resolved in favor of the Attribution License solely as it relates to your use of the GeoNames data.

3001 Summer Street
Stamford CT 06926-0700
USA

www.pitneybowes.com