

precisely

Spectrum Technology Platform

Web Services Guide

Version 2020.1.0



Table of Contents

1 - Getting Started

| | |
|-----------|----|
| REST..... | 4 |
| SOAP..... | 26 |

2 - Web Services

| | |
|--------------------|-----|
| Enabling CORS..... | 43 |
| REST..... | 44 |
| SOAP..... | 597 |

Appendix

| | |
|---|------|
| Appendix A: | |
| Buffering..... | 1007 |
| Appendix B: | |
| Country Codes..... | 1010 |
| Appendix C: | |
| Validate Address Confidence Algorithm.... | 1035 |

1 - Getting Started

In this section

| | |
|-----------|----|
| REST..... | 4 |
| SOAP..... | 26 |



REST

The REST Interface

Spectrum Technology Platform provides a REST interface to web services. User-defined web services, which are those created in Spectrum Enterprise Designer, support GET and POST methods. Default services installed as part of a module only support GET. If you want to access one of these services using POST you must create a user-defined service in Spectrum Enterprise Designer.

To view the REST web services available on your Spectrum Technology Platform server, go to:

```
http://server:port/rest
```

Note: We recommend that you limit parameters to 2,048 characters due to URL length limits.

Service Endpoints

The endpoint for an XML response is:

```
http://server:port/rest/service_name/results.xml
```

The endpoint for a JSON response is:

```
http://server:port/rest/service_name/results.json
```

Endpoints for user-defined web services can be modified in Spectrum Enterprise Designer to use a different URL.

Note: By default Spectrum Technology Platform uses port 8080 for HTTP communication. Your administrator may have configured a different port.

WADL URL

The WADL for a Spectrum Technology Platform web service is:

```
http://server:port/rest/service_name?_wadl
```

For example:

```
http://myserver:8080/rest/ValidateAddress?_wadl
```

User Fields

You can pass extra fields through the web service even if the web service does not use the fields. These fields are returned, unmodified, in the `user_fields` section of the response. For GET requests, user fields are passed in as a parameter in the URL like any other field. For POST requests, user fields are passed in as part of the `user_fields` element in the XML or JSON request.

Note: User field names may not contain characters that are invalid in XML or JSON element names. For example, spaces are not valid.

Sample REST Request Using GET with XML Response

The following example illustrates how to make a REST request to the `ValidateAddress` service using the GET method requesting a response in XML.

```
http://myserver:8080/rest/ValidateAddress/results.xml?Option.OutputCasing=U&Data.AddressLine1=1825+Kramer+Lane&Data.PostalCode=78759
```

The sample request would result in this response since an XML response was requested:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xml.ValidateAddressResponse
xmlns="http://www.precisely.com/spectrum/services/ValidateAddress">
  <output_port>
    <Address>
      <Confidence>82</Confidence>
      <RecordType>Normal</RecordType>
      <CountryLevel>A</CountryLevel>
      <ProcessedBy>USA</ProcessedBy>
      <MatchScore>0</MatchScore>
      <AddressLine1>1825 KRAMER LN</AddressLine1>
      <City>AUSTIN</City>
      <StateProvince>TX</StateProvince>
      <PostalCode>78758-4260</PostalCode>
      <PostalCode.Base>78758</PostalCode.Base>
      <PostalCode.AddOn>4260</PostalCode.AddOn>
      <Country>UNITED STATES OF AMERICA</Country>
      <user_fields/>
    </Address>
  </output_port>
</xml.ValidateAddressResponse>
```

Sample REST Request Using GET with JSON Response

The following example illustrates how to make a REST request to the ValidateAddress service using the GET method requesting a response in JSON.

```
http://myserver:8080/rest/ValidateAddress/results.json?Option.OutputCasing=U&
Data.
AddressLine1=1825+Kramer+Lane&Data.PostalCode=78759
```

The sample request would result in this response since a JSON response was requested:

```
{
  "nsl.json.ValidateAddressResponse" :
  {
    "nsl.output_port" :
    {
      "nsl.Confidence" : 82,
      "nsl.RecordType" : "Normal",
      "nsl.CountryLevel" : "A",
      "nsl.ProcessedBy" : "USA",
      "nsl.MatchScore" : 0,
      "nsl.AddressLine1" : "1825 KRAMER LN",
      "nsl.City" : "AUSTIN",
      "nsl.StateProvince" : "TX",
      "nsl.PostalCode" : "78758-4260",
      "nsl.PostalCode.Base" : 78758,
      "nsl.PostalCode.AddOn" : 4260,
      "nsl.Country" : "UNITED STATES OF AMERICA"
    }
  }
}
```

JSON POST Request

User-defined web services can be exposed as a REST web service and configured to have a POST method that accepts JSON input. Specify `Content-Type:application/json` and use the following format for JSON POST requests.

Flat Data

Use this format in the body request to send flat data to a web service using POST.

```
{
  "InputStageName":
  {
    "InputDataType": [
      {
        "FieldName1": "FieldValue1",
        "FieldName2": "FieldValue2"
      }
    ]
  }
}
```

```

    }
  ]
}

```

Where:

InputStageName

The name of the input stage as shown on the canvas in Spectrum Enterprise Designer. The default name of the stage is `Input`.

InputDataType

The name given to the record-level entity. This value is specified in the dataflow's Input stage, in the **Data type name** field on the **Input Fields** tab. The default name of the record-level entity is `Row`.

FieldName1 and FieldName2

The names of the input fields defined in the service's Input stage.

FieldValue1 and FieldValue2

Input data that you want to send to the web service in the corresponding field.

List Data

List data consists of hierarchical groupings of fields grouped under a parent field.

Note: In order to use list data as input, the service must be exposed as a REST web service without any GET resources. If the service has a GET resource you will get an error in Spectrum Enterprise Designer when exposing the service because hierarchical fields are not supported for GET.

Use the following format to send list data to a web service using POST.

```

{
  "InputStageName":
  {
    "InputDataType": [
      {
        "ListField1": [
          {"SubfieldName1": "SubfieldValue1"},
          {"SubfieldName2": "SubfieldValue2"}
        ]
      }
    ]
  }
}

```

Where:

InputStageName

The name of the input stage as shown on the canvas in Spectrum Enterprise Designer. The default name of the stage is `Input`.

ListField1

The name of the hierarchical field defined in the service's Input stage.

SubfieldName1 and SubfieldName2

The names of child fields that comprise the list field.

SubfieldValue1 and SubfieldValue2

Input data that you want to send to the web service.

User Fields

You can pass extra fields through the web service even if the web service does not use them. These fields are returned, unmodified, in the `user_fields` section of the response. The user fields you supply in the request do not need to be defined in the service dataflow's Input stage.

```
{
  "InputStageName":
  {
    "InputDataType": [
      {
        "user_fields": [
          {
            "name": "FieldName1",
            "value": "FieldValue1"
          },
          {
            "name": "FieldName2",
            "value": "FieldValue2"
          }
        ]
      }
    ]
  }
}
```

Where:

InputStageName

The name of the input stage as shown on the canvas in Spectrum Enterprise Designer.
The default name of the stage is `Input`.

FieldName1 and FieldName2

The name of the pass-through field.

FieldValue1 and FieldValue2

The data you want to include in the passthrough field.

Options

You can specify options in the request, overriding the default options specified in the service dataflow. For user-defined web services, you can only specify options in the request if the dataflow has been

configured to accept options. To configure a service to accept options in the request, open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

To specify processing options in a request, use this format:

```
"options" : {
  "OptionName1" : "Value1"
},
```

Where:

OptionName1

The name of the option. For a list of valid options for the service see the service's WADL or open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

OptionValue1

A legal value for the option. For a list of legal values, open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

Example JSON Request using POST

The following example demonstrates how to include options, flat fields, a list field, and user-defined fields in a JSON request to a web service using POST.

```
{
  "options" : {
    "OutputCasing" : "U"
  },
  "Input":
  {
    "Address": [
      {
        "AddressLine1": "1825 Kramer Ln",
        "City": "Austin",
        "StateProvince": "TX",
        "Accounts": [
          {
            "AccountNumber": "120993",
            "ExpirationDate": "10-3-2017"
          },
          {
            "AccountNumber": "898732",
            "ExpirationDate": "8-13-2016"
          }
        ]
      },
      {
        "name": "Notel",
        "value": "Prefers decaffeinated coffee"
      }
    ]
  }
}
```

```

    },
    {
      "name": "Note2",
      "value": "Requests east facing window"
    }
  ]
}
]
}
}

```

In this example,

- `OutputCasing` is an option exposed by the web service that controls whether the output is returned in upper case or lower case. In this request, it is set to `U` for upper case.
- `Input` is the label of the Input stage in the dataflow as displayed on the canvas in Spectrum Enterprise Designer.
- `Address` is the name of the record-level entity as specified in the dataflow's Input stage, in the **Data type name** field on the **Input Fields** tab.
- `AddressLine1`, `City`, and `StateProvince` are flat fields.
- `Accounts` is a hierarchical ("list") field containing subfields `AccountNumber` and `ExpirationDate`. There are two accounts included in this example.
- `user_fields` contains user-defined fields that are passed through and returned in the output unmodified by the web service.

XML POST Request

User-defined web services can be exposed as a REST web service and configured to have a POST method that accepts XML input. Specify `Content-Type:application/xml` and use the following format for XML POST requests.

Flat Data

Use this format to send flat data to a web service using POST:

```

<ServiceNameRequest
xmlns:svc="http://www.precisely.com/spectrum/services/ServiceName">
  <svc:Input>
    <svc:Row>
      <svc:Field1>Example value</svc:Field1>
      <svc:Field2>Another example value</svc:Field2>
    </svc:Row>
  </svc:Input>
</ServiceNameRequest>

```

Where:

ServiceName

The name of the web service on the Spectrum Technology Platform server.

Field1 and Field2

The names of the input fields defined in the service's Input stage.

For example, this request sends a first name and last name to a service named CasingExample.

```
<CasingExampleRequest
xmlns:svc="http://www.precisely.com/spectrum/services/CasingExample">
  <svc:Input>
    <svc:Row>
      <svc:FirstName>Alex</svc:FirstName>
      <svc:LastName>Smith</svc:LastName>
    </svc:Row>
  </svc:Input>
</CasingExampleRequest>
```

List Data

List data consists of hierarchical groupings of fields grouped under a parent field.

Note: In order to use list data as input, the service must be exposed as a REST web service without any GET resources. If the service has a GET resource you will get an error in Spectrum Enterprise Designer when exposing the service because hierarchical fields are not supported for GET.

Use the following format to send list data to a web service using POST.

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceNameRequest
xmlns:svc="http://www.precisely.com/spectrum/services/ServiceName">
  <svc:Input>
    <svc:Row>
      <svc>ListField1>
        <svc:DataType>
          <svc:SubField1>Example value</svc:SubField1>
          <svc:SubField2>Example value</svc:SubField2>
        </svc:DataType>
      </svc>ListField1>
    </svc:Row>
  </svc:Input>
</ServiceNameRequest>
```

Where:

ListField1

The name of the hierarchical field defined in the service's Input stage.

DataType

The data type of the list field defined in the service's Input stage.

Subfield1 and Subfield2

The names of child fields that comprise the list field.

For example, this request sends a first name, last name, and a list of phone numbers to a service named CasingExample.

```
<CasingExampleRequest
xmlns:svc="http://www.precisely.com/spectrum/services/CasingExample">
  <svc:Input>
    <svc:Row>
      <svc:FirstName>George</svc:FirstName>
      <svc:LastName>Washington</svc:LastName>
      <svc:PhoneNumbers>
        <svc:PhoneNumbers>
          <svc:HomePhone>123-234-9876</svc:HomePhone>
          <svc:CellPhone>123-678-9012</svc:CellPhone>
          <svc:OfficePhone>123-987-6543</svc:OfficePhone>
        </svc:PhoneNumbers>
      </svc:PhoneNumbers>
    </svc:Row>
  </svc:Input>
</CasingExampleRequest>
```

User Fields

You can pass extra fields through the web service even if the web service does not use them. These fields are returned, unmodified, in the `user_fields` section of the response. The user fields you supply in the request do not need to be defined in the service dataflow's Input stage.

```
<ServiceNameRequest
xmlns:svc="http://www.precisely.com/spectrum/services/ServiceName">
  <svc:Input>
    <svc:Row>
      <svc:user_fields>
        <svc:user_field>
          <svc:name>FieldName</svc:name>
          <svc:value>FieldValue</svc:value>
        </svc:user_field>
      </svc:user_fields>
    </svc:Row>
  </svc:Input>
</ServiceNameRequest>
```

Where:

FieldName

The name of the pass-through field.

FieldValue

The value contained in the pass-through field.

For example, this request sends the spouse's name as a pass-through field. The user field name is `Spouse` and the value of the field is `Martha`.

```
<CasingExampleRequest
xmlns:svc="http://www.precisely.com/spectrum/services/CasingExample">
  <svc:Input>
    <svc:Row>
      <svc:FirstName>George</svc:FirstName>
      <svc:LastName>Washington</svc:LastName>
      <svc:PhoneNumbers>
        <svc:PhoneNumbers>
          <svc:HomePhone>123-123-1234</svc:HomePhone>
          <svc:CellPhone>123-456-4567</svc:CellPhone>
          <svc:OfficePhone>123-678-6789</svc:OfficePhone>
        </svc:PhoneNumbers>
      </svc:PhoneNumbers>
      <svc:user_fields>
        <svc:user_field>
          <svc:name>Spouse</svc:name>
          <svc:value>Martha</svc:value>
        </svc:user_field>
      </svc:user_fields>
    </svc:Row>
  </svc:Input>
</CasingExampleRequest>
```

Options

You can specify options in the request, overriding the default options specified in the service dataflow. For user-defined web services, you can only specify options in the request if the dataflow has been configured to accept options. To configure a service to accept options in the request, open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

To specify processing options in a request, use this format:

```
<ServiceNameRequest
xmlns:svc="http://www.precisely.com/spectrum/services/ServiceName">
  <svc:options>
    <svc:OptionName>OptionValue</svc:OptionName>
  </svc:options>
  <svc:Input>
    <svc:Row> ... </svc:Row>
  </svc:Input>
</ServiceNameRequest>
```

Where:

OptionName

The name of the option. For a list of valid options for the service see the service's WADL or open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

OptionValue

A legal value for the option. For a list of legal values, open the service in Spectrum Enterprise Designer and select **Edit > Dataflow Options**.

For example, this request sets the option `OutputCasing` to `U`:

```
<AddressValidationRequest
xmlns:svc="http://www.precisely.com/spectrum/services/AddressValidation">

  <svc:options>
    <svc:OutputCasing>U</svc:OutputCasing>
  </svc:options>
  <svc:Input>
    <svc:Row>
      <svc:FirstName>George</svc:FirstName>
      <svc:LastName>Washington</svc:LastName>
      <svc:AddressLine1>123 Main St.</svc:AddressLine1>
      <svc:City>Springfield</svc:City>
      <svc:StateProvince>MO</svc:City>
    </svc:Row>
  </svc:Input>
</AddressValidationRequest>
```

Example XML Request using POST

The following example demonstrates how to include options, flat fields, a list field, and user-defined fields in an XML request to a web service using POST.

```
<CasingExampleRequest
xmlns:svc="http://www.precisely.com/spectrum/services/CasingExample">
  <svc:options>
    <svc:OutputCasing>U</svc:OutputCasing>
  </svc:options>
  <svc:Input>
    <svc:Row>
      <svc:FirstName>George</svc:FirstName>
      <svc:LastName>Washington</svc:LastName>
      <svc:AddressLine1>1073 Maple</svc:AddressLine1>
      <svc:City>Batavia</svc:City>
      <svc:StateProvince>IL</svc:StateProvince>
      <svc:PhoneNumbers>
        <svc:PhoneNumbers>
          <svc:HomePhone>123-123-1234</svc:HomePhone>
          <svc:CellPhone>123-345-3456</svc:CellPhone>
          <svc:OfficePhone>123-456-4567</svc:OfficePhone>
        </svc:PhoneNumbers>
      </svc:PhoneNumbers>
      <svc:user_fields>
        <svc:user_field>
          <svc:name>Spouse</svc:name>
          <svc:value>Martha</svc:value>
        </svc:user_field>
      </svc:user_fields>
    </svc:Row>
  </svc:Input>
</CasingExampleRequest>
```

```

        </svc:user_fields>
    </svc:Row>
</svc:Input>
</CasingExampleRequest>

```

In this example,

- `OutputCasing` is an option exposed by the web service that controls whether the output is returned in upper case or lower case. In this request, it is set to `U` for upper case.
- `Input` is the label of the Input stage in the dataflow as displayed on the canvas in Spectrum Enterprise Designer.
- `Row` is the name of the record-level entity as specified in the Input stage of the dataflow, in the **Data type name** field on the **Input Fields** tab.
- `FirstName`, `LastName`, `AddressLine1`, `City`, and `StateProvince` are flat fields.
- `PhoneNumbers` is a hierarchical ("list") field containing subfields name `HomePhone`, `CellPhone`, and `OfficePhone`.
- `user_fields` contains user-defined fields that are passed through and returned in the output unmodified by the web service.

Web Service Authentication

Spectrum Technology Platform web services require authentication with valid user credentials. There are two methods for authenticating: Basic authentication and authentication by token.

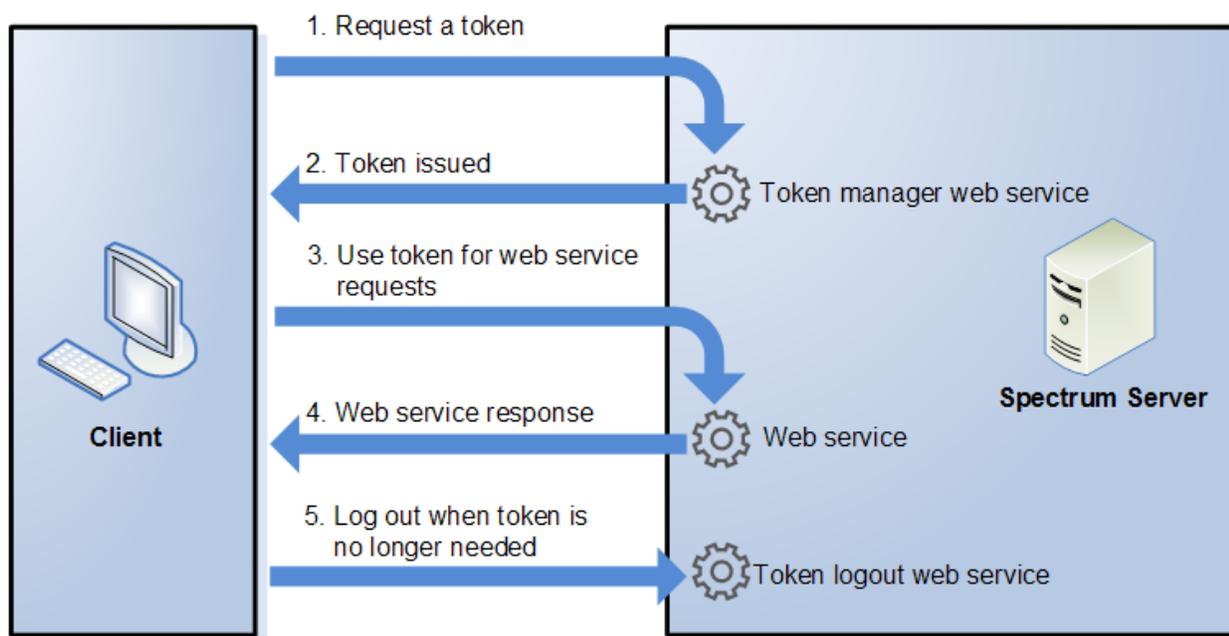
Basic authentication

With Basic authentication, the user ID and password are passed to Spectrum Technology Platform in the HTTP header of each request to the web service. Basic authentication is allowed by default, but your administrator may choose to disable Basic authentication. If Basic authentication is disabled you must use token authentication to access web services.

Authentication by token

With authentication with a token, the requester obtains the token from the Spectrum Technology Platform server, then uses it when sending a request to the web service. Instead of sending user credentials in each request, the token is sent to the server and the server determines if the token is valid.

The diagram below illustrates the process:



1. Obtain a token from the Spectrum Technology Platform server by sending a request to the token manager service.
2. The token manager service issues a token. If you requested a session token it also issues a session ID.
3. Send a request to the desired web service with the token in the HTTP header. For session tokens, include the session ID in the HTTP header.
4. The web service issues a response. You can use the token to make additional web service requests to either the same web service or any other web service on the Spectrum Technology Platform server. There is no limit to the number of web service requests you can make with a token, but if the token has an expiration limit (also known as a time-to-live) it will become invalid after the time-to-live has elapsed. If the token is a session token, it will become invalid after 30 minutes of inactivity.
5. When the token is no longer needed you should log out by sending a request to the token logout web service. This will remove the token from the list of valid tokens on the Spectrum Technology Platform server.

Using Token Authentication

Getting a Token

To get a token, send a request to the `security` web service on the Spectrum Technology Platform server. You can access the `security` WADL here:

```
http://server:port/security/rest?_wadl
```

This web service uses Basic authentication so you must include a valid Spectrum Technology Platform user name and password in the request.

The `security` web service can issue two types of tokens. The token types are:

- Session token
- Open token

Getting a Session Token

A session token is tied to a user session and can only be used by the computer that requested the token. Since it is tied to a session, the token will become invalid if the session is inactive for 30 minutes. A session token is the most secure type of token and is the recommended token type to use to authenticate to Spectrum Technology Platform.

To get a session token, use this URL:

```
http://server:port/security/rest/token/access/session/ttlInMinutes
```

Where:

server

The host name or IP address of your Spectrum Technology Platform server.

port

The HTTP port used by Spectrum Technology Platform. By default this is 8080.

ttlInMinutes

The number of minutes until the token expires, also known as the token time-to-live. If you do not want the token to expire, specify 0.

Here is a sample response:

```
{
  "access_token":
  "eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiaWwiYWxnIjoiZGlyIn0..ESnq4JNEBbVMKycd139z0w.NFXAskVY0seX",
  "session": "09aa1fbb-71j3-43c7-ab8c-d800214283d4",
  "username": "admin"
}
```

The response contains these elements:

| | |
|---------------------|---|
| access_token | The security token. |
| session | The session ID of the session that the token is tied to. The token will only be accepted if this session ID is included in the request. If running with a JavaScript application, you must include a <code>withCredentials: true</code> web request header to ensure the session ID is passed back and forth on all requests. |
| username | The Spectrum Technology Platform user name used to obtain the token. The user name is returned for informational purposes only and is not needed when you use the token. |

Getting an Open Token

An open token is not tied to either a user or a specific computer. It is the least-secure token type.

Important: Avoid using open tokens that do not expire. If an open token is obtained by an unauthorized third party, the token could be used indefinitely, and from any computer, to gain access to your Spectrum Technology Platform server.

To get an open token, use this URL:

```
http://server:port/security/rest/token/access/ttlInMinutes
```

Where:

server

The host name or IP address of your Spectrum Technology Platform server.

port

The HTTP port used by Spectrum Technology Platform. By default this is 8080.

ttlInMinutes

The number of minutes until the token expires, also known as the token time-to-live. If you do not want the token to expire, specify 0.

Here is a sample response:

```
{
  "access_token": "eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiaWwiYWxnIjoiZGlyIn0..fI",
  "username": "admin"
}
```

The response contains these elements:

| | |
|---------------------|--|
| access_token | The security token. |
| username | The Spectrum Technology Platform user name used to obtain the token. The user name is returned for informational purposes only and is not needed when you use the token. |

Using a Token

Once you have obtained a token you can use it to authenticate to a Spectrum Technology Platform web service by including the token in the request. There are two ways of doing this: as an `Authorization` HTTP header or as a `Cookie` HTTP header.

Note: There is no limit to the number of web service requests you can make with a token, but if you requested a token with an expiration, the token will eventually expire. If the token is a session token, it will become invalid after 30 minutes of inactivity.

Using the Token in an Authorization Header

To use the token in the HTTP Authorization header, use the format:

```
Authorization: Bearer Token
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbmMiBMQI4Q0JDLUhmTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
```

If the token is a session token, you must also provide the session identifier in the Cookie header in the form:

```
Cookie: SESSION=SessionID
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbmMiBMQI4Q0JDLUhmTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
Cookie: SESSION=fff96e54-1615-4192-96c1-ea2f133ec6eb
```

Note: The cookie name SESSION must be in all caps.

Using the Token in a Cookie Header

If it is easier to use a cookie rather than the Authorization header you can provide the token in the Cookie header in the form:

```
Cookie: spectrum.authentication.token=Token
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Cookie:
spectrum.authentication.token=eyJlbmMiBMQI4Q0JDLUhmTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
```

If the token is a session token, you must also provide the session identifier in the `Cookie` header in the form:

```
Cookie: SESSION=SessionID
```

Note: The cookie name `SESSION` must be in all caps.

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Cookie:
spectrum.authentication.token=eyJlbmMiBmQI4Q0JDLUhTMjU2I5wiYWxnIjoizGlyIn0..fc6rpRJ-wo
Cookie: SESSION=fff96e54-1615-4192-96c1-ea2f133ec6eb
```

Logging Off

After you are done using a token you should send a request to the `security` web service to remove the token from the list of valid tokens maintained on the Spectrum Technology Platform server. You can access the `security` WADL here:

```
http://server:port/security/rest?wadl
```

To log out, use this URL:

```
http://server:port/security/rest/token/logout
```

Include the token in the `Authorization` HTTP header or in the `Cookie` HTTP header. If the token is a session token, include the session in the `Cookie` header. For more information, see [Using a Token](#) on page 35. This service does not take any parameters.

Exposing a Service as a Web Service

Spectrum Technology Platform services can be made available as RESTful and SOAP web services. To make a service available on your server as a web service:

1. Open Spectrum Enterprise Designer.
2. Open the service that you want to expose as a web service.
3. Go to **Edit > Web Service Options**.
4. To make the service available as a SOAP web service, check the box **Expose as SOAP web service**.

5. To make the service available as a REST web service, check the box **Expose as REST web service** and complete these steps.

- a) If you want to override the default endpoint, specify the endpoint you want to use in the **Path** field.

Specifying a path is optional. By default, a REST web service's endpoint is:

```
http://server:port/rest/service_name/results.qualifier
```

If you want to use a different endpoint, the path you specify is added after the service name. For example, if you specify `Americas/Shipping` in the **Path** field, your JSON endpoint would be something like this:

```
http://myserver:8080/rest/MyService/Americas/Shipping/results.json
```

You can use fields and options from the flow as variable names in the path by clicking the **Insert variable** drop-down menu and selecting the field or option you want to use. The variable is represented in the path using the notation `${Option.Name}` for flow options and `${Data.Name}` for flow fields.

- b) By default REST web services support the GET method and return data in XML and JSON formats. You can define additional HTTP methods and output formats by clicking **Add** to add a resource to the web service.

When you add a resource, you can choose the HTTP method (**GET** or **POST**). The supported data formats are listed below. You may not have all these formats available to you because some formats are only available if you have certain modules installed on your Spectrum Technology Platform server.

XML The default XML format. Use this format if you want to use XML as the format for requests and responses, and there is no specialized XML format for the kind of data you want to process.

JSON The default JSON format. Use this format if you want to use JSON as the format for requests and responses, and there is no specialized JSON format for the kind of data you want to process.

GeoJSON A specialized JSON format that is appropriate for services that handle geographic data. Support is provided only for Geometry and for these native platform types:

- boolean
- double
- float
- integer
- bigdecimal
- long
- string
- date
- time

- datetime
- timespan

If you try to expose a flow with any other type, you will not be able to specify GeoJSON (an error will appear at design-time). Also, GeoJSON only allows a single geometry. If the output contains multiple geometry fields, the system will search for a field called "geometry" followed by a field called "obj." If those fields do not exist, the first geometry field will be selected.

c) Click **OK**.

The new resource is added to the web service.

6. Click **OK** when you are done configuring the web service options.
7. Click the gray light bulb in the tool bar to expose the service.

When a flow is exposed the light bulb button in the Spectrum Enterprise Designer tool bar indicates that the flow is exposed as shown here:



To verify that the service is now exposed as a web service, go to one of the following URLs:

- For REST: `http://server:port/rest`
- For SOAP: `http://server:port/soap`

Where *server* is the name or IP address of your Spectrum Technology Platform server and *port* is the port used for HTTP communication.

Adding POST Support to a REST Web Service

Some Spectrum Technology Platform modules come with standard web services, such as the ValidateAddress web service that comes with the Spectrum Universal Address. These web services support GET only. You can add POST support to these standard web services by creating a user-defined service in Spectrum Enterprise Designer and placing the standard service in it as a stage. Since you can expose user-defined services with POST support, you are in effect creating a web service that exposes the standard service with POST support.

1. Open Spectrum Enterprise Designer.
2. Go to **File > New > Dataflow > Service**.
3. Drag an **Input** and **Output** stage onto the canvas.
4. Drag the service onto the canvas and connect the **Input** and **Output** stages to it.

For example, if you want to expose ValidateAddress with POST support, your dataflow would look like this:



5. Configure each stage.
6. Go to **Edit > Web Service Options**.
7. Check **Expose as REST web service**.
8. Click **Add**, select **POST**, and select the input and output format.
9. Click **OK**, then click **OK** again.

Note: For detailed instructions on configuring web service options, see [Exposing a Service as a Web Service](#) on page 37.

10. Save and expose the service.

You have created a user-defined web service that exposes a standard service as a REST web service that supports POST.

Micro-batch Processing

Micro-batch processing is a technique where you include more than one record in a single service request. By including multiple records in a request instead of issuing separate requests for each record, you can significantly improve performance when processing a large collection of records through a service. Spectrum Technology Platform supports micro-batch processing for REST and SOAP web services.

Micro-batch end point

For AMER, use `amer-microbatch.precisely.com`. It has a 5 minute timeout allowing larger micro-batch sizes.

For APAC and EMEA, use the standard endpoint as they do not have a special micro-batch endpoint. This endpoint has a 30 second timeout, so the number of records in a micro-batch will need to be smaller.

Micro-Batch Size

For AMER using the special micro-batch endpoint, you may put as many records as can fit within a 5 minute timeout. You are charged for each record in the request even if it times out, so it is recommended to choose the number of records that can be processed in 4 minutes in case processing takes longer.

For APAC and EMEA using the standard endpoint, the maximum number of records allowed in a request depends on the service's category.

Using a Record ID

You may find it helpful to assign an ID to each record in a micro-batch so that you can correlate the records in the request with the records returned in the response. Use user fields to do this. For information about user fields, see [The REST Interface](#) on page 4.

Micro-Batch Processing in REST

To perform micro-batch processing with a REST web service, include two or more records as XML or JSON in the body of the request and send the request using the POST method. For more information about sending a POST request to a Spectrum Technology Platform web service, see [JSON POST Request](#) on page 6 and [XML POST Request](#) on page 10.

For example, this request includes two records as XML in the body of the request:

```
POST https://spectrum.precisely.com/rest/ValidateAddressPOST/results.xml
HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/xml
Authorization: Basic YWRtaW46YWRtaW4=
Content-Length: 533
Host: spectrum.precisely.com
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<ValidateAddressPOSTRequest
xmlns:svc="http://www.precisely.com/spectrum/services/ValidateAddressPOST">

  <svc:Input>
    <svc:Row>
      <svc:AddressLine1>3001 Summer</svc:AddressLine1>
      <svc:City>Stamford</svc:City>
      <svc:StateProvince>CT</svc:StateProvince>
    </svc:Row>
    <svc:Row>
      <svc:AddressLine1>33 west monroe</svc:AddressLine1>
      <svc:City>Chicago</svc:City>
      <svc:StateProvince>IL</svc:StateProvince>
    </svc:Row>
  </svc:Input>
</ValidateAddressPOSTRequest>
```

Note: Services do not have POST support enabled by default. To perform micro-batch processing with these services you must enable POST support. For more information, see [Adding POST Support to a REST Web Service](#) on page 22.

Sample .NET Class

The following .NET class calls the `ValidateAddress` web service. It is written in C# on Visual Studio 2010. Proxy class implementations for the web service data types `ValidateAddressClient`, `requestRow`, `context`, `options`, and `responseRow` were generated using Visual Studio .NET's "Add Service Reference" command. It is important to note that in this example the appropriate credentials must be provided or the call will fail.

```
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Net;
using System.Text;
using ConsoleApplication1.ValidateAddress_Reference;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {
            var validateClient = new ValidateAddress {Credentials = new
NetworkCredential("admin", "admin")};

            var address1 = new input_portAddress
            {
                AddressLine1 = "1825B Kramer Lane",
                AddressLine2 = "Suite 100",
                PostalCode = "78758",
                City = "Austin",
                StateProvince = "Texas"
            };

            var address2 = new input_portAddress
            {
                AddressLine1 = "100 Congress",
                PostalCode = "78701",
                City = "Austin",
                StateProvince = "Texas"
            };

            var addresses = new input_portAddress[2];
            addresses[0] = address1;
            addresses[1] = address2;

            var options = new options {OutputCasing = OutputCasing.M};
            output_portAddress[] results =
validateClient.CallValidateAddress(options, addresses);
```

```

        for (int i = 0; i < results.Length; i++)
        {
            System.Console.WriteLine("Record " + (i+1) + ":");
            System.Console.WriteLine("AddressLine1=" +
results[i].AddressLine1);
            System.Console.WriteLine("City=" + results[i].City);
            System.Console.WriteLine("StateProvince=" +
results[i].StateProvince);
            System.Console.WriteLine("PostalCode=" +
results[i].PostalCode + "\n");
        }

        System.Console.Write("Press any key to continue...");
        System.Console.ReadKey();
    }
}

```

SOAP

The SOAP Interface

The Spectrum Technology Platform server provides access to services using SOAP in document/literal mode. Document/literal web services are the WS-I compliant format for web services.

To view the SOAP web services available on your Spectrum Technology Platform server, go to:

`http://server:port/soap`

Note: By default Spectrum Technology Platform uses port 8080 for HTTP communication. Your administrator may have configured a different port.

WSDL URL

The WSDL for a Spectrum Technology Platform web service is:

`http://server:port/soap/service_name?wsdl`

For example:

`http://myserver:8080/soap/ValidateAddress?wsdl`

The web service model is a generic model for all services. The WSDL for any Spectrum Technology Platform web service contains the same definition of data types and operations. What differentiates the WSDL is the target service and the values (options and data) provided at runtime.

User Fields

You can pass extra fields through the web service even if the web service does not use them. These fields are returned, unmodified, in the `<user_fields>` element of the response. For example, this request contains a user field named `id` with a value of 5:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddress">

  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressRequest>
      <val:input_port>
        <val:Address>
          <val:AddressLine1>3001 summer</val:AddressLine1>
          <val:City>stamford</val:City>
          <val:StateProvince>ct</val:StateProvince>
          <val:user_fields>
            <val:user_field>
              <val:name>id</val:name>
              <val:value>5</val:value>
            </val:user_field>
          </val:user_fields>
        </val:Address>
      </val:input_port>
    </val:ValidateAddressRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

The user field is returned, unmodified, in the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ValidateAddressResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/ValidateAddress">

      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>86</ns3:Confidence>
          <ns3:RecordType>HighRise</ns3:RecordType>
          <ns3:RecordType.Default>Y</ns3:RecordType.Default>
          <ns3:CountryLevel>A</ns3:CountryLevel>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:MatchScore>0</ns3:MatchScore>
          <ns3:AddressLine1>3001 Summer St</ns3:AddressLine1>
```

```

    <ns3:City>Stamford</ns3:City>
    <ns3:StateProvince>CT</ns3:StateProvince>
    <ns3:PostalCode>06905-4317</ns3:PostalCode>
    <ns3:PostalCode.Base>06905</ns3:PostalCode.Base>
    <ns3:PostalCode.AddOn>4317</ns3:PostalCode.AddOn>
    <ns3:Country>United States Of America</ns3:Country>
    <ns3:AdditionalInputData.Base/>
    <ns3:POBoxOnlyDeliveryZone/>
    <ns3:user_fields>
      <ns3:user_field>
        <ns3:name>id</ns3:name>
        <ns3:value>5</ns3:value>
      </ns3:user_field>
    </ns3:user_fields>
  </ns3:Address>
</ns3:output_port>
</ns3:ValidateAddressResponse>
</soap:Body>
</soap:Envelope>

```

Note: User field names may not contain characters that are invalid in XML element names. For example, spaces are not valid.

Sample SOAP Request

The following sample SOAP request calls the ValidateAddress service. The sections for options and rows are all dependent on the metadata for that particular web service; therefore, different components will have different metadata entries. Additionally, there is a user_fields section that allows you to pass in field values that will be returned, unmodified, in the response.

```

<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddress">
  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressRequest>
      <val:options>
        <val:OutputFormattedOnFail>Y</val:OutputFormattedOnFail>
      </val:options>
      <val:input_port>
        <val:Address>
          <val:AddressLine1>1525B Kramer Lane</val:AddressLine1>
          <val:AddressLine2>Suite 100</val:AddressLine2>
          <val:PostalCode>78758</val:PostalCode>
        </val:Address>
      </val:input_port>
    </val:ValidateAddressRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

Sample SOAP Response

The sample request above would return the following response.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ValidateAddressResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/ValidateAddress">

      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>88</ns3:Confidence>
          <ns3:RecordType>Normal</ns3:RecordType>
          <ns3:CountryLevel>A</ns3:CountryLevel>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:MatchScore>0</ns3:MatchScore>
          <ns3:AddressLine1>1525B Kramer Ln Ste
100</ns3:AddressLine1>
          <ns3:City>Austin</ns3:City>
          <ns3:StateProvince>TX</ns3:StateProvince>
          <ns3:PostalCode>78758-4227</ns3:PostalCode>
          <ns3:PostalCode.Base>78758</ns3:PostalCode.Base>
          <ns3:PostalCode.AddOn>4227</ns3:PostalCode.AddOn>
          <ns3:Country>United States Of America</ns3:Country>
          <ns3:user_fields/>
        </ns3:Address>
      </ns3:output_port>
    </ns3:ValidateAddressResponse>
  </soap:Body>
</soap:Envelope>
```

Web Service Authentication

Spectrum Technology Platform web services require authentication with valid user credentials. There are two methods for authenticating: Basic authentication and authentication by token.

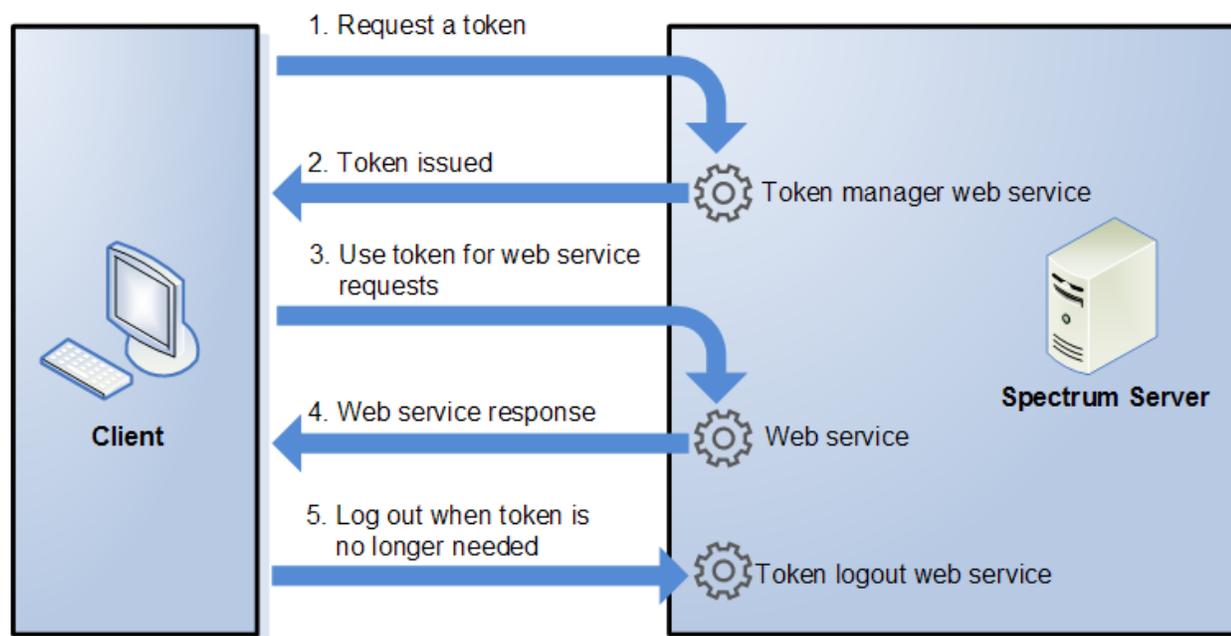
Basic authentication

With Basic authentication, the user ID and password are passed to Spectrum Technology Platform in the HTTP header of each request to the web service. Basic authentication is allowed by default, but your administrator may choose to disable Basic authentication. If Basic authentication is disabled you must use token authentication to access web services.

Authentication by token

With authentication with a token, the requester obtains the token from the Spectrum Technology Platform server, then uses it when sending a request to the web service. Instead of sending user credentials in each request, the token is sent to the server and the server determines if the token is valid.

The diagram below illustrates the process:



1. Obtain a token from the Spectrum Technology Platform server by sending a request to the token manager service.
2. The token manager service issues a token. If you requested a session token it also issues a session ID.
3. Send a request to the desired web service with the token in the HTTP header. For session tokens, include the session ID in the HTTP header.
4. The web service issues a response. You can use the token to make additional web service requests to either the same web service or any other web service on the Spectrum Technology Platform server. There is no limit to the number of web service requests you can make with a token, but if the token has an expiration limit (also known as a time-to-live) it will become invalid after the time-to-live has elapsed. If the token is a session token, it will become invalid after 30 minutes of inactivity.
5. When the token is no longer needed you should log out by sending a request to the token logout web service. This will remove the token from the list of valid tokens on the Spectrum Technology Platform server.

Using Token Authentication

Getting a Token

To get a token, send a request to the `TokenManagerService` web service on the Spectrum Technology Platform server. You can access the `TokenManagerService` WSDL here:

```
http://server:port/security/TokenManagerService?wsdl
```

This web service uses Basic authentication so you must include a valid Spectrum Technology Platform user name and password in the request.

The `TokenManagerService` web service can issue two types of tokens. The token types are:

- Session token
- Open token

Getting a Session Token

A session token is tied to a user session and can only be used by the computer that requested the token. Since it is tied to a session, the token will become invalid if the session is inactive for 30 minutes. A session token is the most secure type of token and is the recommended token type to use to authenticate to Spectrum Technology Platform.

`TokenManagerService` has two SOAP operations for getting a session token.

| Operation | Description |
|------------------------|---|
| getAccessExpiringToken | Use this operation if you want to specify an expiration time for the token. Here is a sample request: |

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:tok="http://token.security.common.server.platform.spectrum.precisely.com/"

  <soapenv:Header/>
  <soapenv:Body>
    <tok:getAccessExpiringToken>
      <tokenLifeInMinutes>60</tokenLifeInMinutes>

    </tok:getAccessExpiringToken>
  </soapenv:Body>
</soapenv:Envelope>
```

The element `<tokenLifeInMinutes>` specifies the number of minutes until the token expires, also known as the token time-to-live. In this example the token will expire in 60 minutes.

Here is a sample response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"

  <soap:Body>
    <ns2:getAccessExpiringTokenResponse

xmlns:ns2="http://token.security.common.server.platform.spectrum.precisely.com/"

      <return>

<session>ebd7904b-07f6-15c9-82e4-71589131eb01</session>

<token>eyJhbGciOiJIUzI1NiIsInR5cCI6IkpzZW50L3p1ciJ9.eyJpdiI6ImI0UjR1U0p0Z3ppdC9kQlR9</token>

      <username>simon0897</username>
    </return>
  </ns2:getAccessExpiringTokenResponse>
</soap:Body>
</soap:Envelope>
```

| Operation | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-----------------------|---|
| getAccessSessionToken | Use this operation if you want to get a token that will not expire. Note that the token will still become invalid if the session is inactive for 30 minutes, even if the token has not expired. |
|-----------------------|---|

Here is a sample request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:tok="http://token.security.common.server.platform.spectrum.precisely.com/">

  <soapenv:Header/>
  <soapenv:Body>
    <tok:getAccessSessionToken/>
  </soapenv:Body>
</soapenv:Envelope>
```

Here is a sample response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Body>
    <ns2:getAccessSessionTokenResponse

xmlns:ns2="http://token.security.common.server.platform.spectrum.precisely.com/">

      <return>

<session>65822c9b-362e-2e0e-a02a-a50a1a761323</session>

<token>eyJlbmMiOiJMTTQ0UjU1MjU2LiwiYXNjaioiZGlyIn0..CEE-ClVjKTha</token>

        <username>simon0897</username>
      </return>
    </ns2:getAccessSessionTokenResponse>
  </soap:Body>
</soap:Envelope>
```

The response contains these elements:

| | |
|----------------|---|
| token | The security token. |
| session | The session ID of the session that the token is tied to. The token will only be accepted if this session ID is included in the request. If running with a JavaScript application, you must include a <code>withCredentials: true</code> web request header to ensure the session ID is passed back and forth on all requests. |

username The Spectrum Technology Platform user name used to obtain the token. The user name is returned for informational purposes only and is not needed when you use the token.

Getting an Open Token

An open token is not tied to either a user or a specific computer. It is the least-secure token type.

Important: Avoid using open tokens that do not expire. If an open token is obtained by an unauthorized third party, the token could be used indefinitely, and from any computer, to gain access to your Spectrum Technology Platform server.

TokenManagerService has one SOAP operation for getting an open token.

| Operation | Description |
|----------------|---|
| getAccessToken | <p>Use this operation to get an open token. Here is a sample request:</p> <pre><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:tok="http://token.security.common.server.platform.spectrum.precisely.com/" <soapenv:Header/> <soapenv:Body> <tok:getAccessToken/> </soapenv:Body> </soapenv:Envelope></pre> <p>Here is a sample response:</p> <pre><soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" <soap:Body> <ns2:getAccessTokenResponse xmlns:ns2="http://token.security.common.server.platform.spectrum.precisely.com/" <return> <token>eyJlbnMiOiJMTT40QDUhMjU2TiwiYXNjaioiZGlyIn0..f4JwB4qPs0l</token> <username>paul1234</username> </return> </ns2:getAccessTokenResponse> </soap:Body> </soap:Envelope></pre> |

The response contains these elements:

| | |
|-----------------|--|
| token | The security token. |
| username | The Spectrum Technology Platform user name used to obtain the token. The user name is returned for informational purposes only and is not needed when you use the token. |

Using a Token

Once you have obtained a token you can use it to authenticate to a Spectrum Technology Platform web service by including the token in the request. There are two ways of doing this: as an `Authorization` HTTP header or as a `Cookie` HTTP header.

Note: There is no limit to the number of web service requests you can make with a token, but if you requested a token with an expiration, the token will eventually expire. If the token is a session token, it will become invalid after 30 minutes of inactivity.

Using the Token in an Authorization Header

To use the token in the HTTP `Authorization` header, use the format:

```
Authorization: Bearer Token
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbnMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
```

If the token is a session token, you must also provide the session identifier in the `Cookie` header in the form:

```
Cookie: SESSION=SessionID
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbnMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
Cookie: SESSION=fff96e54-1615-4192-96c1-ea2f133ec6eb
```

Note: The cookie name `SESSION` must be in all caps.

Using the Token in a Cookie Header

If it is easier to use a cookie rather than the `Authorization` header you can provide the token in the `Cookie` header in the form:

```
Cookie: spectrum.authentication.token=Token
```

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Cookie:
spectrum.authentication.token=eyJlbmMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
```

If the token is a session token, you must also provide the session identifier in the `Cookie` header in the form:

```
Cookie: SESSION=SessionID
```

Note: The cookie name `SESSION` must be in all caps.

For example,

```
HTTP/1.1
POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Cookie:
spectrum.authentication.token=eyJlbmMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..fc6rpRJ-wo
Cookie: SESSION=fff96e54-1615-4192-96c1-ea2f133ec6eb
```

Logging Off

After you are done using a token you should send a request to the `TokenLogoutService` web service to remove the token from the list of valid tokens maintained on the Spectrum Technology Platform server. You can access the `TokenLogoutService` WSDL here:

```
http://server:port/security/TokenLogoutService?wsdl
```

To log out, send a request to the `TokenLogoutService` web service and include the token in the `Authorization` HTTP header or in the `Cookie` HTTP header. If the token is a session token, include the session in the `Cookie` header. For more information, see [Using a Token](#) on page 35. This service does not take any parameters.

Exposing a Service as a Web Service

Spectrum Technology Platform services can be made available as RESTful and SOAP web services. To make a service available on your server as a web service:

1. Open Spectrum Enterprise Designer.
2. Open the service that you want to expose as a web service.
3. Go to **Edit > Web Service Options**.
4. To make the service available as a SOAP web service, check the box **Expose as SOAP web service**.
5. To make the service available as a REST web service, check the box **Expose as REST web service** and complete these steps.
 - a) If you want to override the default endpoint, specify the endpoint you want to use in the **Path** field.

Specifying a path is optional. By default, a REST web service's endpoint is:

```
http://server:port/rest/service_name/results.qualifier
```

If you want to use a different endpoint, the path you specify is added after the service name. For example, if you specify `Americas/Shipping` in the **Path** field, your JSON endpoint would be something like this:

```
http://myserver:8080/rest/MyService/Americas/Shipping/results.json
```

You can use fields and options from the flow as variable names in the path by clicking the **Insert variable** drop-down menu and selecting the field or option you want to use. The variable is represented in the path using the notation `${Option.Name}` for flow options and `${Data.Name}` for flow fields.

- b) By default REST web services support the GET method and return data in XML and JSON formats. You can define additional HTTP methods and output formats by clicking **Add** to add a resource to the web service.

When you add a resource, you can choose the HTTP method (**GET** or **POST**). The supported data formats are listed below. You may not have all these formats available to you because some formats are only available if you have certain modules installed on your Spectrum Technology Platform server.

- | | |
|-------------|--|
| XML | The default XML format. Use this format if you want to use XML as the format for requests and responses, and there is no specialized XML format for the kind of data you want to process. |
| JSON | The default JSON format. Use this format if you want to use JSON as the format for requests and responses, and there is no specialized JSON format for the kind of data you want to process. |

GeoJSON A specialized JSON format that is appropriate for services that handle geographic data. Support is provided only for Geometry and for these native platform types:

- boolean
- double
- float
- integer
- bigdecimal
- long
- string
- date
- time
- datetime
- timespan

If you try to expose a flow with any other type, you will not be able to specify GeoJSON (an error will appear at design-time). Also, GeoJSON only allows a single geometry. If the output contains multiple geometry fields, the system will search for a field called "geometry" followed by a field called "obj." If those fields do not exist, the first geometry field will be selected.

c) Click **OK**.

The new resource is added to the web service.

6. Click **OK** when you are done configuring the web service options.
7. Click the gray light bulb in the tool bar to expose the service.

When a flow is exposed the light bulb button in the Spectrum Enterprise Designer tool bar indicates that the flow is exposed as shown here:



To verify that the service is now exposed as a web service, go to one of the following URLs:

- For REST: `http://server:port/rest`
- For SOAP: `http://server:port/soap`

Where *server* is the name or IP address of your Spectrum Technology Platform server and *port* is the port used for HTTP communication.

Micro-batch Processing

Micro-batch processing is a technique where you include more than one record in a single service request. By including multiple records in a request instead of issuing separate requests for each record, you can significantly improve performance when processing a large collection of records through a service. Spectrum Technology Platform supports micro-batch processing for REST and SOAP web services.

Micro-batch end point

For AMER, use `amer-microbatch.precisely.com`. It has a 5 minute timeout allowing larger micro-batch sizes.

For APAC and EMEA, use the standard endpoint as they do not have a special micro-batch endpoint. This endpoint has a 30 second timeout, so the number of records in a micro-batch will need to be smaller.

Micro-Batch Size

For AMER using the special micro-batch endpoint, you may put as many records as can fit within a 5 minute timeout. You are charged for each record in the request even if it times out, so it is recommended to choose the number of records that can be processed in 4 minutes in case processing takes longer.

For APAC and EMEA using the standard endpoint, the maximum number of records allowed in a request depends on the service's category.

Using a Record ID

You may find it helpful to assign an ID to each record in a micro-batch so that you can correlate the records in the request with the records returned in the response. Use user fields to do this. For information about user fields, see [The SOAP Interface](#) on page 26.

Micro-Batch Processing in SOAP

To perform micro-batch processing in a SOAP web service, include two or more records in the SOAP request. For example, this request contains two records:

```
POST http://spectrum.example.com:8080/soap/ValidateAddress HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: ""
Authorization: Basic YWRtaW46YWRtaW4=
Content-Length: 782
Host: config813vm0:8080
```

```

Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddress">

  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressRequest>
      <val:input_port>
        <val:Address>
          <val:AddressLine1>1 N. State St.</val:AddressLine1>
          <val:City>Chicago</val:City>
          <val:StateProvince>IL</val:StateProvince>
        </val:Address>
        <val:Address>
          <val:AddressLine1>3001 summer</val:AddressLine1>
          <val:City>stamford</val:City>
          <val:StateProvince>ct</val:StateProvince>
        </val:Address>
      </val:input_port>
    </val:ValidateAddressRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

Sample .NET Class

The following .NET class calls the ValidateAddress web service. It is written in C# on Visual Studio 2010. Proxy class implementations for the web service data types ValidateAddressClient, requestRow, context, options, and responseRow were generated using Visual Studio .NET's "Add Service Reference" command. It is important to note that in this example the appropriate credentials must be provided or the call will fail.

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Net;
using System.Text;
using ConsoleApplication1.ValidateAddress_Reference;

namespace Test
{
    class Program
    {
        static void Main(string[] args)
        {

```

```

        var validateClient = new ValidateAddress {Credentials = new
NetworkCredential("admin", "admin")};

        var address1 = new input_portAddress
        {
            AddressLine1 = "1825B Kramer Lane",
            AddressLine2 = "Suite 100",
            PostalCode = "78758",
            City = "Austin",
            StateProvince = "Texas"
        };

        var address2 = new input_portAddress
        {
            AddressLine1 = "100 Congress",
            PostalCode = "78701",
            City = "Austin",
            StateProvince = "Texas"
        };

        var addresses = new input_portAddress[2];
        addresses[0] = address1;
        addresses[1] = address2;

        var options = new options {OutputCasing = OutputCasing.M};
        output_portAddress[] results =
validateClient.CallValidateAddress(options, addresses);

        for (int i = 0; i < results.Length; i++)
        {
            System.Console.WriteLine("Record " + (i+1) + ":");
            System.Console.WriteLine("AddressLine1=" +
results[i].AddressLine1);
            System.Console.WriteLine("City=" + results[i].City);
            System.Console.WriteLine("StateProvince=" +
results[i].StateProvince);
            System.Console.WriteLine("PostalCode=" +
results[i].PostalCode + "\n");
        }

        System.Console.Write("Press any key to continue...");
        System.Console.ReadKey();
    }
}
}

```

2 - Web Services

In this section

| | |
|--------------------|-----|
| Enabling CORS..... | 43 |
| REST..... | 44 |
| SOAP..... | 597 |



Enabling CORS

Cross-Origin Resource Sharing (CORS) is a W3C standard that allows data sharing between domains. CORS enables web applications running in one domain to access data from another domain.

By enabling CORS on your Spectrum Technology Platform server, you can allow web applications hosted in another domain to access Spectrum Technology Platform web services.

For example, say you have a web application hosted at **webapp.example.com**. This web application contains a JavaScript function that calls a Spectrum Technology Platform web service hosted at **spectrum.example.com**. Without CORS, you would need to use a proxy server to facilitate this request, which would add complexity to your implementation. With CORS, you do not need to use a proxy server. Instead, you can designate **webapp.example.com** as an "allowed origin", thus permitting Spectrum Technology Platform to respond to web service requests that originate from the domain **webapp.example.com**.

To enable CORS on your Spectrum Technology Platform server:

1. Stop the Spectrum Technology Platform server.
2. Open this file in a text editor:

```
SpectrumDirectory/server/conf/spectrum-container.properties
```

3. Edit the following parameters.

spectrum.http.cors.enabled

Set this property to true to enable CORS. The default is false.

spectrum.http.cors.allowedOrigins

A comma separated list of origins that are allowed to access resources on the Spectrum Technology Platform server. The default is `http://localhost:8080,http://localhost:443`, which allows access to resources using the default HTTP port 8080 and the default HTTPS port of 443.

If an allowed origin contains one or more asterisks ("*"), for example `http://*.domain.com`, then asterisks are converted to `.*` and dots characters (".") are escaped to `\.` and the resulting allowed origin is interpreted as a regular expression. Allowed origins can therefore be more complex expressions such as `https?://*.domain.[a-z]{3}` that matches http or https, multiple subdomains and any three-letter top-level domain (such as `.com`, `.net`, `.org`).

spectrum.http.cors.allowedMethods

A comma separated list of HTTP methods that are allowed to be used when accessing resources on the Spectrum Technology Platform server. The default value is `POST,GET,OPTIONS,PUT,DELETE,HEAD`.

spectrum.http.cors.allowedHeaders

A comma separated list of HTTP headers that are allowed when accessing resources on the Spectrum Technology Platform server. The default value is X-PINGOTHER, Origin, X-Requested-With, Content-Type, Accept. If the value is a single asterisk ("*"), all headers will be accepted.

spectrum.http.cors.preflightMaxAge

The number of seconds that preflight requests can be cached by the client. The default value is 1800 seconds, or 30 minutes.

spectrum.http.cors.allowCredentials

Indicates whether the resource allows requests with credentials. The default value is true.

4. Save and close the file.
5. Start the Spectrum Technology Platform server.

REST

Context Graph

Context Graph REST API

Context Graph provides a persistent repository to help you manage and understand your most critical data assets. Context Graph incorporates entities and relationships in models.

Context Graph provides a set of REST APIs that are used to create, read, update, and delete entities and relationships in a Context Graph model. The operations communicate values in JSON name-value pairs. This section provides general information for working with the Context Graph REST APIs, as well as specific reference information for each available operation. The Spectrum server supports both HTTP and HTTPS for the requests. The API includes entity operations and relationship operations.

- [Entity Requests](#) on page 45
- [Relationship Requests](#) on page 54

Entity Requests

Entity requests read, create, update or delete an entity in a Context Graph model.

Table 1: Entity Operations

| Operation | Description |
|--|--|
| Create Entity Operation on page 45 | The create entity operation adds a new entity to a Context Graph model. The entity metadata must already exist in the Context Graph model. |
| Read Entity Operation on page 47 | The read entity operation returns property values for an entity in a Context Graph model. |
| Update Entity Operation on page 51 | The update entity operation replaces property values in an existing entity. |
| Delete Entity Operation on page 53 | The delete entity operation removes an existing entity. |

Create Entity Operation

The create entity operation adds a new entity to a Context Graph model. The entity metadata must already exist in the Context Graph model.

HTTP PUT URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

PUT

`http://server_name:port/rest/DataHub/operations/modelName/entities/entityType/entityLabel`

URL Path Elements

modelName

The name of the Context Graph model.

entityType

An entity type defined in the model

entityLabel

The label for the new entity.

URL PUT Body Format

`Content-Type:application/json {Property Name-Value Pairs}`

Optionally, you can specify a property name and value pair for any existing property in the following format. A property is not created if it is omitted or contains a null or empty value. The property name pairs are formatted as follows:

```
{
  "Property1": "Value1",
  "Property2": "Value2",
  ...
}
```

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Create entity with JSON response

The following request creates a "Place" entity type in the "911" model with the label "FlightSafety International" and adds the following properties: "Latitude", "Location", "Longitude", and "Place".

```
PUT
http://localhost:8080/rest/DataHub/operations/911/entities/Place/FlightSafety%20International
```

Body:

```
{
  "Latitude": "27.6386433",
  "Location": "Vero Beach, Florida",
  "Longitude": "-80.39727",
  "Place": "FlightSafety International",
  "Date": 1275782400000
}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Response:

```
{
  "success": "200 OK"
}
```

Read Entity Operation

The read entity operation returns property values for an entity in a Context Graph model.

HTTP GET URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

GET

`http://server_name:port/rest/DataHub/operations/modelName/entities/entityType/entityLabel?query_parameters`

URL Path Elements

modelName

The name of the Context Graph model.

entityType

An entity type defined in the model

entityLabel

The label for an existing entity in the model.

Query Parameters

The following parameters are specified in the query string, preceded in the URL by the question mark (?). Combine parameter name and value pairs with the ampersand (&).

| Parameter | Type | Required | Description |
|---|---------|----------|--|
| <code>retrieveHistory</code> | Boolean | no | Set this parameter to true to retrieve the entity history. The history includes values for each iteration in the history along with changes. The default is false. |
| <code>includeRelationshipHistory</code> | Boolean | no | This parameter may be used with the <code>retrieveHistory</code> parameter to add the history for connected relationships. Set this parameter to true to include the relationship history. The default is false. |

Return codes

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Response without history

The response returns a name-value pair for each entity property in the following format. Note that numeric values are not enclosed between quotation marks.

```
{
  "result":{
    "property":value,
    "property":value,
    ...
    "property":value
  }
}
```

JSON response without history

The following request reads properties from the "Place" entity type in the "911" model with the label "FlightSafety International".

```
GET
http://localhost:8080/rest/DataHub/operations/911/entities/Place/FlightSafety%20International
```

This results in the following response:

```
{"result":{
  "Latitude":"27.6386433",
  "Location":"Vero Beach, Florida",
  "Longitude":"-80.39727",
  "Place":"FlightSafety International",
  "Date":1275782400000
}}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Response with entity history

The response when you include the retrieveHistory parameter includes values for each historical instance. Inclusion of from and to values in the history depend on whether the eventType is Create, Update, or Delete.

```
{
  "result": {
    "property": value,
    property: value,
    ...
    property: value
  },
  "history": [
```

```

    {
      "stpId": "entityType:entityLabel",
      "eventType": "Create|Update|Delete",
      "versionDatetime": epoch_timestamp,
      "versionDatetimeNanos": epoch_timestamp(nanos),
      "username": "userName",
      "edgeChanges": [],
      "properties": [
        {
          "name": "property",
          "eventType": "Create|Update|Delete",
          "to": value
        },
        ...
        {
          "name": "property",
          "eventType": "Create|Update|Delete",
          "to": value
        }
      ]
    }
  ]
}

```

JSON response with entity history

The following request reads properties from the "John Smith" entity in the Customer Banking model

```
GET http://localhost:8080/rest/DataHub/operations/CustomerBanking_DataType/entities/Person/John%20Smith?retrieveHistory=true&includeRelationshipHistory=false
```

This results in the following response:

```

{
  "result": {
    "USCitizen": true,
    "Name": "John Smith",
    "AccountNumber": 9955420
  },
  "history": [ {
    "stpId": "Person:John Smith",
    "eventType": "Create",
    "versionDatetime": 1594590216325,
    "versionDatetimeNanos": 120456689138214,
    "username": "hub_security_allperm",
    "edgeChanges": [],
    "properties": [
      {
        "name": "AccountNumber",
        "eventType": "Create",
        "to": 9955420
      }
    ]
  }
]
}

```

```

    },
    {
      "name": "Name",
      "eventType": "Create",
      "to": "John Smith"
    },
    {
      "name": "USCitizen",
      "eventType": "Create",
      "to": true
    }
  ]
}

```

Response with entity relationship history

The response when you include the `includeRelationshipHistory` parameter includes values for each connected relationship. Inclusion of `from` and `to` values in the history depend on whether the `eventType` is `Create`, `Update`, or `Delete`.

```

{
  "result": {
    "parameter": value,
    ...
    "parameter": value
  },
  "history": [ {
    "stpId": "entityType:entityLabel",
    "eventType": "Create|Update|Delete",
    "versionDatetime": epoch_timestamp,
    "versionDatetimeNanos": epoch_timestamp(nanos),
    "username": "userName",
    "edgeChanges": [ {
      "stpId": "relationshipLabel",
      "eventType": "Create|Update|Delete",
      "sourceStpId": "entityType:entityLabel",
      "targetStpId": "entityType:entityLabel",
      "versionDatetime": epoch_timestamp,
      "versionDatetimeNanos": epoch_timestamp(nanos),
      "username": "userName",
      "changes": [ {
        "relationshipProperty": "value",
        "eventType": "Create|Update|Delete",
        "to": value
      }
    ]
  }
],
  "properties": []
},

```

JSON response with entity relationship history

The following request reads properties from the "John Smith" entity and connected relationships in the Customer Banking model.

```
GET
http://localhost:8080/rest/DataHub/operations/CustomerBanking_DataType/entities/Person/John%20Smith?retrieveHistory=true&includeRelationshipHistory=true
```

This results in the following response:

```
{
  "result": {
    "USCitizen": false,
    "Name": "John Smith",
    "AccountNumber": 995542
  },
  "history": [ {
    "stpId": "Person:John Smith",
    "eventType": "Update",
    "versionDatetime": 1594398222284,
    "versionDatetimeNanos": 274973121721755,
    "username": "hub_security_allperm",
    "edgeChanges": [ {
      "stpId": "LivesAt",
      "eventType": "Create",
      "sourceStpId": "Person:John Smith",
      "targetStpId": "Address:123 Main Street",
      "versionDatetime": 1594398222284,
      "versionDatetimeNanos": 274973121721755,
      "username": "hub_security_allperm",
      "changes": [ {
        "name": "PostalCode",
        "eventType": "Create",
        "to": 21119
      }
    ]
  }
  ],
  "properties": []
},
```

Update Entity Operation

The update entity operation replaces property values in an existing entity.

HTTP POST URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

```
POST
http://server_name:port/rest/DataHub/operations/modelName/entities/entityType/entityLabel
```

URL Path Elements

modelName

The name of the Context Graph model.

entityType

An entity type defined in the model

entityLabel

The label for an existing entity.

URL POST Body Format

Content-Type:application/json {*Property Name-Value Pairs*}

You can specify a property name and value pair for any existing property in the following format:

```
{
  "Property1": "Value1",
  "Property2": "Value2",
  ...
}
```

At least one property is required to successfully complete the operation. Omitted properties will not be changed.

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Update entity with JSON response

The following request updates property values for the "Place" entity type in the "911" model with the label "FlightSafety International".

POST
 http://localhost:8080/rest/DataHub/operations/911/entities/Place/FlightSafety%20International

Body:

```
{
  "Latitude": "27.6386433",
  "Location": "Vero Beach, Florida",
  "Longitude": "-80.39727",
  "Place": "FlightSafety International",
  "Date": 1275782400000
}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Response:

```
{
  "success": "200 OK"
}
```

Delete Entity Operation

The delete entity operation removes an existing entity.

HTTP DEL URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

DEL

`http://server_name:port/rest/DataHub/operations/modelName/entities/entityType/entityLabel`

URL Path Elements

modelName

The name of the Context Graph model.

entityType

An entity type defined in the model.

entityLabel

The label for an existing entity.

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Delete entity with JSON Response

The following request deletes the "Place" entity type in the "911" model with the label "FlightSafety International".

DEL

`http://localhost:8080/rest/DataHub/operations/911/entities/Place/FlightSafety%20International`

Response:

```
{
  "success": "200 OK"
}
```

Relationship Requests

Relationship requests create, read, update, or delete a relationship in a Context Graph model.

Table 2: Relationship Operations

| Operation | Description |
|--|---|
| Create Relationship Operation on page 54 | The create relationship operation adds a new relationship between two entities in a Context Graph model. The relationship metadata must already exist in the Context Graph model. |
| Read Relationship Operation on page 56 | The read relationship operation returns property values for a relationship in a Context Graph model. |
| Update Relationship Operation on page 60 | The update relationship operation replaces property values in an existing relationship. |
| Delete Relationship Operation on page 62 | The delete relationship operation removes an existing relationship between two entities. |

Create Relationship Operation

The create relationship operation adds a new relationship between two entities in a Context Graph model. The relationship metadata must already exist in the Context Graph model.

HTTP PUT URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

```
PUT http://server_name:port/rest/DataHub/operations/modelName/
relationships/relationshipLabel?query_parameters
```

URL Path Elements

modelName

The name of the Context Graph model.

relationshipLabel

The name of the relationship label connecting two entities in a model.

Query Parameters

| Parameter | Type | Required | Description |
|-----------|--------|----------|---|
| sourceID | string | yes | The source ID of the entity that connects a relationship. This parameter specifies an <i>entityType : entityTypeLabel</i> value pair. |
| targetID | string | yes | The target ID of the entity that connects a relationship. This parameter specifies an <i>entityType : entityTypeLabel</i> value pair. |
| uniqueID | string | no | The value that distinguishes a relationship when there are multiple relationships with the same label connecting two entities in a model. |

URL PUT Body Format

Content-Type:application/json {*Property Name-Value Pairs*}

Optionally, you can specify a property name and value pair for any existing property in the following format. A property is not created if it contains a null or empty value. The property name pairs are formatted as follows:

```
{
  "Property1": "Value1",
  "Property2": "Value2",
  ...
}
```

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Create relationship with JSON response

The following request creates a Roomate relationship label in the "911" model between two person entities and adds the following properties: "Date" and "Rank".

```
PUT http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah
```

Body:

```
{
  "Date":1275782400000,
  "Rank":"0"
}
```

Response:

```
{
  "success":"200 OK"
}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Create relationship with same label

The following creates a Roomate relationship label in the "911" model where the relationship ID between two person entities is equal to "4".

```
PUT http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah&uniqueID=4
```

Read Relationship Operation

The read relationship operation returns property values for a relationship in a Context Graph model.

HTTP GET URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

GET

`http://server_name:port/rest/DataHub/operations/modelName/relationships/relationshipLabel?
query_parameters`

URL Path Elements
modelName

The name of the Context Graph model.

relationshipLabel

The name of the relationship label connecting two entities in a model.

Query Parameters

| Parameter | Type | Required | Description |
|-----------------|---------|----------|--|
| sourceID | string | yes | The source ID of the entity that connects a relationship. This parameter specifies an <i>entityType : entityTypeLabel</i> value pair. |
| targetID | string | yes | The target ID of the entity that connects a relationship. This parameter specifies an <i>entityType : entityTypeLabel</i> value pair. |
| uniqueID | string | no | The value that distinguishes a relationship when there are multiple relationships with the same label connecting two entities in a model. |
| retrieveHistory | boolean | no | Set this parameter to true to retrieve the relationship history. The history includes values for each iteration in the history along with changes. The default is false. |

Return codes

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Response without history

The response returns a name-value pair for each relationship property in the following format, when a relationship has *N* properties.

```
{ "result": {
  "Property1": "Value1",
  "Property2": "Value2",
  ...
  "PropertyN": "ValueN"
}}
```

JSON response without history

The following request reads properties from the `Roomate` relationship label in the "911" model between two person entities.

```
GET http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah
```

This results in the following response:

```
{"result":{
  "Date":1275782400000,
  "Rank":"0"
}}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Response with relationship history

The response returns a historical record for changes to properties in the relationship.

```
{"result":{
  "property":"value",
  "property":"value",
  ...
  "property":value
},
"history": [ {
  "stpId": "relationshipLabel",
  "eventType": "Create|Update|Delete",
  "sourceStpId": "entityType:entityLabel",
  "targetStpId": "entityType:entityLabel",
  "versionDatetime": epic_timestamp,
  "versionDatetimeNanos":_epic_timestamp(nanos),
  "username": "userName",
  "changes": [
    {
      "name": "property",
      "eventType": "Create|Update|Delete",
      "to": value
    },
    ...
    {
      "name": "property",
      "eventType": "Create|Update|Delete",
      "to": value
    }
  ]
}
```

```
    }}
  }
```

JSON response with relationship history

The following request reads properties from the `BanksAt` relationship between two entities in the Customer Banking model.

```
GET
http://localhost:8080/rest/DataHub/operations/CustomerBanking_DataType/relationships/BanksAt?sourceID=Person%3AJohn%20Smith&
targetID=Bank%3ABank%20of%20America&retrieveHistory=true
```

This results in the following response:

```
{
  "result": {
    "OpenTime": 49524000,
    "Float": 2123.98,
    "OpenDateTime": 1275831924000,
    "OpenDate": 1275782400000,
    "Long": 2200007654021,
    "Double": 21.05
  },
  "history": [
    {
      "stpId": "BanksAt",
      "eventType": "Create",
      "sourceStpId": "Person:John Smith",
      "targetStpId": "Bank:Bank of America",
      "versionDatettime": 1594398241650,
      "versionDatettimeNanos": 274992488700358,
      "username": "hub_security_allperm",
      "changes": [
        {
          "name": "Double",
          "eventType": "Create",
          "to": 21.05
        },
        {
          "name": "Float",
          "eventType": "Create",
          "to": 2123.98
        },
        {
          "name": "Long",
          "eventType": "Create",
          "to": 2200007654021
        },
        {
          "name": "OpenDate",
          "eventType": "Create",
          "to": 1275782400000
        }
      ]
    }
  ]
}
```

```

    },
    {
      "name": "OpenDateTime",
      "eventType": "Create",
      "to": 1275831924000
    },
    {
      "name": "OpenTime",
      "eventType": "Create",
      "to": 49524000
    }
  ]
}]
}

```

Update Relationship Operation

The update relationship operation replaces property values in an existing relationship.

HTTP POST URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

POST

`http://server_name:port/rest/DataHub/operations/modelName/relationships/relationshipLabel?
query_parameters`

URL Path Elements

modelName

The name of the Context Graph model.

relationshipLabel

The name of the relationship label connecting two entities in a model.

Query Parameters

| Parameter | Type | Required | Description |
|-----------|--------|----------|---|
| sourceID | string | yes | The source ID of the entity that connects a relationship. This parameter specifies an <i>entityType:entityLabel</i> value pair. |
| targetID | string | yes | The target ID of the entity that connects a relationship. This parameter specifies an <i>entityType:entityLabel</i> value pair. |

| Parameter | Type | Required | Description |
|-----------|--------|----------|---|
| uniqueID | string | no | The value that distinguishes a relationship when there are multiple relationships with the same label connecting two entities in a model. |

URL POST Body Format

Content-Type:application/json {*Property Name-Value Pairs*}

Optionally, you can specify a property name and value pair for any existing property in the following format. A property is removed if a null or empty value is specified here. The property name pairs are formatted as follows:

```
{
  "Property1": "Value1",
  "Property2": "Value2",
  ...
}
```

At least one property is required to successfully complete the operation. Omitted properties will not be changed.

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Update relationship with JSON response

The following request updates the date property for a Roomate relationship label in the "911" model.

```
POST http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah
```

Body:

```
{
  "Date":1275782400000
}
```

Response:

```
{
  "success": "200 OK"
}
```

Note: Date, time, and date-time property values are UNIX Epoch time values formatted as long data type in both requests and responses.

Update relationship with same label

The following updates a Roomate relationship label in the "911" model where the relationship ID between two person entities is equal to "4".

```
POST http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah&uniqueID=4
```

Delete Relationship Operation

The delete relationship operation removes an existing relationship between two entities.

HTTP DEL URL Format

The request is specified as follows. The Spectrum server supports both HTTP and HTTPS.

DEL

```
http://server_name:port/rest/DataHub/operations/modelName/relationships/relationshipLabel?
query_parameters
```

URL Path Elements

modelName

The name of the Context Graph model.

relationshipLabel

The name of the relationship label connecting two entities in a model.

Query Parameters

| Parameter | Type | Required | Description |
|-----------|--------|----------|--|
| sourceID | string | yes | The source ID of the entity that connects a relationship. This parameter specifies an <i>entityType</i> : <i>entityLabel</i> value pair. |
| targetID | string | yes | The target ID of the entity that connects a relationship. This parameter specifies an <i>entityType</i> : <i>entityLabel</i> value pair. |

| Parameter | Type | Required | Description |
|-----------|--------|----------|---|
| uniqueID | string | no | The value that distinguishes a relationship when there are multiple relationships with the same label connecting two entities in a model. |

Response

The operation returns the status code "200 OK" when it is successful. The operation returns the status code 500 (Error) when it fails.

Delete relationship label with JSON response

The following request deletes the "Roomate" relationship label in the "911" model between two person entities.

```
DEL http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah
```

Response:

```
{
  "success": "200 OK"
}
```

Delete relationship with same label

The following deletes a Roomate relationship label in the "911" model where the relationship ID between two person entities is equal to "4".

```
DEL http://localhost:8080/rest/DataHub/operations/911/relationships/Roomate?
sourceID=Person:Ahmed%20al-Haznawi&targetID=Person:Ziad%20Jarrah&uniqueID=4
```

Spectrum Enterprise Tax

AssignGeoTAXInfo

AssignGeoTAXInfo identifies the tax districts that apply to a given address. Specifically, AssignGeoTAXInfo returns the following information about an address:

- Latitude/longitude coordinates
- FIPS state codes and county codes

- County names
- MCD/CCD codes and names
- CBSA/CSA codes and names
- Place FIPS and GNIS codes and names
- Incorporated or unincorporated status codes
- Cross-reference tax keys
- Result indicators
- Optionally, the relationship of an address to user-defined polygons

AssignGeoTAXInfo optionally includes enhanced tax jurisdiction information for an address, including:

- **Insurance premium districts**—Areas designated for the collection of taxes imposed on insurance policy premiums based on the policy holder's address. Insurance premium districts are created by state governments.
- **Payroll tax districts**—Areas designated for the collection of taxes imposed on employers to support state or local government facilities and services based on the employee's and/or employer's address. Examples include taxes collected for districts to pay for schools, police, or other services. Payroll tax districts are created by state or local governments.
- **Payroll system tax codes**—Codes that represent specific jurisdictions that collect payroll tax. Using payroll system tax codes has advantages over using the payroll tax district information returned by Assign GeoTAX Info:
 - AssignGeoTAXInfo uses an additional database to determine payroll tax codes, resulting in more accurate payroll tax determination.
 - Many payroll systems use specific codes to determine withholding amounts. Since you can customize the payroll tax codes returned by AssignGeoTAXInfo, you can set up a process where AssignGeoTAXInfo returns the exact payroll tax codes required by your payroll system instead of returning jurisdictional IDs that must then be translated into the codes used by your system.
- **Special purpose tax districts**—Areas designated for the collection of taxes imposed on residents to support specialized services for residents of the district based on the resident's address. Examples include services such as sewer service, transit service, or water resources. Special purpose tax districts are created by legislative action, court action, or public referendums. This optional information requires the use of boundary files which require an additional license. Contact your Precisely sales representative for more information.
- **Sales and Use Tax Rates**—Using the optional Precisely Sales and Use Tax Rate file, AssignGeoTAXInfo can return sales and use tax rates for each of the assigned tax jurisdictions as well as the total tax rate for the assigned locations.

AssignGeoTAXInfo is part of Spectrum Enterprise Tax.

Resource URL

JSON endpoint:

```
http://server:port/rest/AssignGeoTaxInfo/results.json
```

XML endpoint:

```
http://server:port/rest/AssignGeoTaxInfo/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/AssignGeoTAXInfo/results.json?
Data.AddressLine1=1+Global+View&Data.City=Troy&
Data.StateProvince=NY&Data.PostalCode=12180
```

The JSON returned by this request would be:

```
{ "output_port": [{
  "Confidence": "100.0",
  "ProcessedBy": "GTX",
  "Census.MatchCode": "S",
  "Census.MatchLevel": "Street",
  "County.Code": "083",
  "County.Name": "Rensselaer",
  "StateCode": "36",
  "LatLong": "42.683028-073.702968",
  "LatLong.MatchCode": "R",
  "LatLong.MatchLevel": "Rooftop",
  "Latitude": "42.683028",
  "Longitude": "-073.702969",
  "State.Abbreviation": "NY",
  "Place.Code": "00000",
  "Place.IncorporatedFlag": "Uninc",
  "AddressLine1": "1 GLOBAL VW",
  "City": "TROY",
  "StateProvince": "NY",
  "PostalCode": "121808371",
  "AddressMatch.MatchCode": "S80",
  "AddressMatch.LocationCode": "AS0",
  "AddressMatch.LastLine": "TROY, NY 12180-8371",
  "AddressMatch.Zip": "12180",
  "AddressMatch.Zip4": "8371",
  "AddressMatch.GenRC": "S",
  "AddressMatch.DataType": "TOMTOM",
  "MCD.DistanceToBorder": "000002938",
  "Place.DistanceToBorder": "00000000",
  "GNISCode": "000000000",
}] }
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/AssignGeoTAXInfo/results.xml?
Data.AddressLine1=1+Global+View&Data.City=Troy&
Data.StateProvince=NY&Data.PostalCode=12180
```

The XML returned by this request would be:

```
<ns2:xml.AssignGeoTAXInfoResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/AssignGeoTAXInfo">

  <ns2:output_port>
    <ns2:Address>
      <ns2:Confidence>100.0</ns2:Confidence>
      <ns2:ProcessedBy>GTX</ns2:ProcessedBy>
      <ns2:Census.MatchCode>S</ns2:Census.MatchCode>
      <ns2:Census.MatchLevel>Street</ns2:Census.MatchLevel>
      <ns2:County.Code>083</ns2:County.Code>
      <ns2:County.Name>Rensselaer</ns2:County.Name>
      <ns2:StateCode>36</ns2:StateCode>
      <ns2:LatLong>42.683028-073.702968</ns2:LatLong>
      <ns2:LatLong.MatchCode>R</ns2:LatLong.MatchCode>
      <ns2:LatLong.MatchLevel>Rooftop</ns2:LatLong.MatchLevel>
      <ns2:Latitude>42.683028</ns2:Latitude>
      <ns2:Longitude>-073.702969</ns2:Longitude>
      <ns2:State.Abbreviation>NY</ns2:State.Abbreviation>
      <ns2:Place.Code>00000</ns2:Place.Code>
      <ns2:Place.IncorporatedFlag>Uninc</ns2:Place.IncorporatedFlag>

      <ns2:AddressLine1>1 GLOBAL VW</ns2:AddressLine1>
      <ns2:City>TROY</ns2:City>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:PostalCode>121808371</ns2:PostalCode>
      <ns2:AddressMatch.MatchCode>S80</ns2:AddressMatch.MatchCode>

      <ns2:AddressMatch.LocationCode>AS0</ns2:AddressMatch.LocationCode>
      <ns2:AddressMatch.LastLine>TROY, NY
12180-8371</ns2:AddressMatch.LastLine>
      <ns2:AddressMatch.Zip>12180</ns2:AddressMatch.Zip>
      <ns2:AddressMatch.Zip4>8371</ns2:AddressMatch.Zip4>
      <ns2:AddressMatch.GenRC>S</ns2:AddressMatch.GenRC>

      <ns2:AddressMatch.DataTypename>TOMTOM</ns2:AddressMatch.DataTypename>
      <ns2:MCD.DistanceToBorder>000002938</ns2:MCD.DistanceToBorder>

      <ns2:Place.DistanceToBorder>00000000</ns2:Place.DistanceToBorder>

      <ns2:GNISCode>000000000</ns2:GNISCode>
      <ns2:LatLong.StreetMatchCode/>
      <ns2:LatLong.StreetMatchLevel/>
```

```

    </ns2:Address>
  </ns2:output_port>
</ns2:xml.AssignGeoTAXInfoResponse>

```

Request

Parameters for Input Data

The following table provides information about the format of AssignGeoTAXInfo input.

| Parameter | Format | Description |
|-------------------|-----------------|---|
| Data.AddressLine1 | String [100] | First address line |
| Data.AddressLine2 | String [100] | Second address line |
| Data.AddressLine3 | String [100] | Third address line |
| Data.AddressLine4 | String [100] | Fourth address line |
| Data.BufferWidth | String [10] | <p>Specifies the width of the polygon buffers to use for Boundary File processing. The buffer width is used to determine if a point is close to the edge of a polygon. The output field BufferRelation indicates whether or not the point is within the polygon's buffer area. For more information, see Buffering on page 1008.</p> <p>This field overrides the value specified in the <code>Option.DefaultBufferWidth</code> parameter. Specify the border width in the units specified by the <code>Option.DistanceUnits</code> parameter.</p> <p>If you do not specify a buffer width in this input field, the default is used.</p> |
| Data.City | String [50] | City name |

| Parameter | Format | Description |
|---------------------|--------------|---|
| Data.Country | String [var] | The country where the address resides. The data you enter in this field has no impact on processing. It is simply passed through to output. Note: Only US addresses are supported. |
| Data.FirmName | String [var] | The company or firm name. |
| Data.PostalCode | String [9] | Nine-digit ZIP Code |
| Data.StateProvince | String [50] | The state where the address resides. The data you enter in this field has no impact on processing. It is simply passed through to output. |
| Data.UseBufferWidth | Long [10] | Specifies the width of the polygon buffers to use for User-Defined Boundary File processing. The buffer width is used to determine if a point is close to the edge of a polygon. The output field BufferRelation indicates whether or not the point is within the polygon's buffer area. For more information, see Buffering on page 1008. This field overrides the value specified in the <code>Option.DefaultBufferWidth</code> parameter. Specify the border width in the units specified by the <code>Option.DistanceUnits</code> parameter. If you do not specify a buffer width in this input field, the default is used. |

Matching Options

Matching options control the address search methodology and match results handling returned by AssignGeoTAXInfo.

| Parameter | Description |
|--|-------------|
| Optional files: The following options enable the database resource(s) to use in the search process. | |

| Parameter | Description |
|-------------------------------|---|
| Option.UseGeoTaxAuxiliaryFile | <p>Specifies whether or not AssignGeoTAXInfo should attempt a match to the GeoTAX Auxiliary file. The GeoTAX Auxiliary file contains new addresses that have not yet been added to the Master File.</p> <p>Y Use the GeoTAX Auxiliary file for matching. (default)</p> <p>N Do not use the GeoTAX Auxiliary file for matching.</p> |
| Option.UseAuxiliaryFile | <p>Specifies whether to attempt a match to a User Auxiliary file. User Auxiliary files are user-defined files that override results from the master files in street-level matching.</p> <p>Y Use the User Auxiliary file for matching.</p> <p>N Do not use the User Auxiliary file for matching. (default)</p> |
| Option.UseStateProvidedFile | <p>Specifies whether to attempt a match to the state-supplied file. Use this option in combination with <code>FileSearchOrder</code> to specify a state-supplied file to use.</p> <p>State-supplied files are provided by individual state governments. By matching to the state-supplied files, you can remain compliant with tax jurisdiction assignment requirements mandated by new federal and state laws, such as the Mobile Telecommunications Sourcing Act and the Florida state Communications Services Tax Simplification Law.</p> <p>There are two supported file formats: the Florida-native format and the national TS-158 format (ANSI Transaction Set No. 158). The state of Florida provides address files in both the TS-158 and its own native format.</p> <p>If this option is enabled, the address is first matched to the state supplied file. If a state match cannot be found, the master files are used to attempt a match.</p> <p>Y Use the State-supplied file for matching.</p> <p>N Do not use the State-supplied file for matching. (default)</p> <p>Note: You must install the appropriate State-supplied file to use these options. For instructions, see the <i>Spectrum Technology Platform Installation Guide</i>.</p> |
| Option.FileSearchOrder | <p>Specifies which state-supplied file to use. This option only takes effect if you specify <code>Option.UseStateProvidedFile=Y</code>. One of the following:</p> <p>FLOnly Use only the Florida-native formatted file. (default)</p> <p>TSOnly Use only the TS-158 formatted file.</p> |

| Parameter | Description |
|------------------------------------|--|
| Option.UseRelaxedSecondaryMatching | Specifies whether input addresses with secondary information are matched to records without secondary information. This option applies only to Florida-native files. <ul style="list-style-type: none"> Y Use relaxed secondary matching. N Do not use relaxed secondary matching. (default) |

Address Searching and Matching: These options can be enabled for use in the address search and match processes.

| | |
|--------------------|--|
| Option.GsMatchMode | Match modes determine the leniency used to make a match between your input and the reference database. Select a match mode based on the quality of your input and your desired output. For example, if you have an input database that is prone to errors, you may want to select the relaxed match mode. <ul style="list-style-type: none"> 0 - Exact Requires a very tight match. This restrictive mode generates the fewest match candidates, which decreases the processing time. When using this mode, ensure that your input is very clean; free of misspellings and incomplete addresses. 1 - Close Requires a close match and generates a moderate number of match candidates. (default) 2 - Relaxed Allows a loose match and generates the most match candidates, which increases the processing time and results in more multiple matches. Use this mode if you are not confident that your input is clean; free of misspellings and incomplete addresses. This is the only mode that does not respect the street parity when making an address match. |
|--------------------|--|

| | |
|---------------------|--|
| Option.GsSearchArea | The search area options allow for searching the address' finance area or an expanded area specified by distance. <ul style="list-style-type: none"> 1 Searches the entire finance area for a match. A finance area is a region defined by the U.S. Postal Service and typically consists of a set of contiguous ZIP Codes.(default) 2 Searches the area specified by the radius in miles. The search area can be extended up to a 99-mile radius from the centroid of the input ZIP Code to assist in finding a match when the input address contains limited or inaccurate city or ZIP Code information. The expanded area is confined to within the state's borders. |
|---------------------|--|

| | |
|-----------------------|--|
| Option.GsSearchRadius | Radius for search area. <ul style="list-style-type: none"> 1-99 miles Search radius. (default = 0 miles) |
|-----------------------|--|

| Parameter | Description |
|------------------------------------|--|
| Option.GsEnableFirstLetterExpanded | <p>Looks for the correct first letter of a street address if the first letter is missing or incorrect. Spectrum Enterprise Tax searches through the alphabet looking for possible correct first letters to complete the street address.</p> <p>Note: This feature is disabled by default and cannot be enabled in Exact mode.</p> <p>Y Enable first letter change matches.</p> <p>N Do not allow first letter change matches. (default)</p> |
| Option.GsEnableRangedAddress | <p>Matches to a house range input. 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. When this feature is enabled, the address range is geocoded to the interpolated mid-point of the range.</p> <p>Note: This feature is disabled by default and cannot be enabled in Exact mode.</p> <p>Y Allow address range matches.</p> <p>N Do not allow address range matches. (default)</p> |
| Option.GsAlternateLookup | <p>This option allows specifying the preferred way to match when both an address and firm name are provided. The matching method can be set to match to the address rather than the firm name or vice versa. If neither are specified, the default matching method is to match to the address line only.</p> <p>1 Searches for street name, but if there isn't a match, will use the firm name.</p> <p>2 Looks up the firm name, but if there isn't a match, will use the street name.</p> <p>3 Searches only street records. (default)</p> |
| Option.GsMultiMatchResolution | <p>A multi-match occurs when multiple equally-scored matches are found in either the Points or Streets files and cannot be resolved to a single best candidate. There are several choices for handling a multi-match outcome:</p> <p>N No matches are returned. (default)</p> <p>R Return the first match candidate in the list.</p> <p>A The information for all the match candidates is returned.</p> |

Geocoding Options

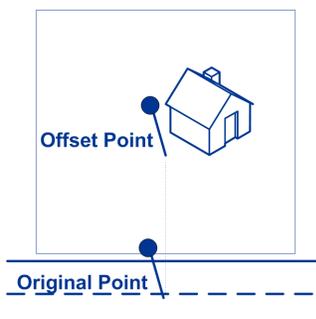
Geocoding is the process of determining the latitude/longitude coordinates of a given address. Address coordinates are used as the basis for determining the tax jurisdictions for an address. Geocoding options control how AssignGeoTAXInfo determines address latitude/longitude coordinates.

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Latitude/Longitude placement: These options can be set for the geocode result.

Option.LatLongOffset Indicates the offset distance in feet from the street center line.

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. Since the building represented by an address is not on the street itself, you do not want the geocode for an address to be a point on the street. Instead, you want the geocode to represent the location of the building which sits next to the street. For example, an offset of 40 feet means that the geocode will represent a point 40 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. The diagram below shows an offset point in relation to the original point.



- 0** No offset. (default)
- 20** Twenty feet offset from street centerline.
- 40** Forty feet offset from street centerline. (recommended)
- 60** Sixty feet offset from street centerline.

| Parameter | Description |
|----------------|---|
| Option.Squeeze | <p>Specifies if the street end points should be "squeezed" when determining the geocode of an address in street-level matching. When <code>Option.Squeeze</code> is enabled, both street and end points are moved closer to the center of the segment by 50 feet. The diagram below compares the end points of a street segment to the squeezed end points of a street segment.</p>  <p>Y Apply squeeze.</p> <p>N Do not apply squeeze. (default)</p> |

| | |
|----------------------|---|
| Option.LatLongFormat | <p>Indicates the desired format for the returned latitude/longitude. One of the following:</p> <p>PreZero Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W.</p> <p>PreZeroDecimal Decimal degrees using directional indicator. For example, 090.000000N180.000000W. (default)</p> <p>Decimal Decimal degrees using signed latitude/longitude. For example, 90.000000-180.000000.</p> <p>DecimalAssumed Decimal degrees using signed latitude/longitude (no decimal point). For example, 90000000-180000000.</p> <p>DegMinSec Degrees, minutes, seconds. For example, 90 00 00N180 00 00W.</p> |
|----------------------|---|

Expanded Geocoding options: These options enable additional geocoding functionality.

| Parameter | Description |
|--|---|
| Option.GsEnableAddressPointInterpolation | <p>Address point interpolation uses a patented process that improves upon regular street segment interpolation by inserting point data into the interpolation process.</p> <p>Note: This feature is only for use with point-level geocoding.</p> <p>A match is first attempted using the loaded points data. If an exact point match is found in the points data, then searching ceases and the point match is returned. If an exact point match was not found, Spectrum Enterprise Tax attempts to find high and low boundary address points to use for address point interpolation.</p> <p>Y Enable address point interpolation.</p> <p>N Disable address point interpolation. (default)</p> |

Minimum geocode quality

| | |
|-----------------------------------|---|
| Option.GsEnableGeographicFallback | <p>The default search does not perform a search of geographic centroids. When enabled, the Geographic Fallback feature locates the first city, county and/or state centroid, and then matches from the set of possible matches found.</p> <p>Y If a definitive match cannot be made, then return the next higher level geographic centroid.</p> <p>N Disable geographic fallback feature. (default)</p> |
|-----------------------------------|---|

| Parameter | Description |
|--|--|
| Option.GsEnableStreetCentroid | <p>If an input street address cannot be found using the street number and name, Spectrum Enterprise Tax then searches the input ZIP Code or city/state for the closest match. If Spectrum Enterprise Tax is able to locate the street, it returns a geocode along the matched street segment rather than the geocode for the entered ZIP Code or ZIP + 4.</p> <p>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) returns. For example, if you enter Main St and there are both an E Main St and a W Main St within the input ZIP Code, then an E030 returns and the location code returned is reflective of the input ZIP Code. The location code returned begins with a "C" when matched to a single street segment, indicated by E029. The Spectrum Enterprise Tax does not change the street name on the output address.</p> <p>Y If a street or point match cannot be made, then return a street level centroid.</p> <p>N Do not return a street level centroid if a match cannot be made. (default)</p> <p>Note: This feature should only be used for exception processing or research. It should not be used in a production process.</p> |
| <p>Boundary matching: These options can be set when matching to a boundary file such as SPD, IPD, PAY, Place and MCD or user-defined.</p> | |
| Option.DistanceUnits | <p>Specifies the units in which to measure distance. One of the following:</p> <p>Feet Distances are measured in feet. (default)</p> <p>Meters Distances are measured in meters.</p> |
| Option.DefaultBufferWidth | <p>Specifies the buffer width to use for tax district boundary files. The tax district boundary files are the Special Purpose District (SPD) file, the Insurance Premium District (IPD) file, the Payroll Tax District (PAY) file, and Place and MCD files.</p> <p>The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>For more information about buffers, see Buffering on page 1008.</p> |

| Parameter | Description |
|-------------------------------|---|
| Option.DefaultUserBufferWidth | <p>Specifies the buffer width to use for user-defined boundary files. Specify the distance in the units of measurement specified in the Distance units option. For information about buffers, see Buffering on page 1008. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>Note: To use buffers, the user-defined boundary file must support buffers.</p> |

Output Data Options

Data options control the data returned by AssignGeoTAXInfo.

| Parameter | Description |
|-------------------------------|---|
| Option.GeoTAXOutputRecordType | <p>Select one or more of the following to obtain the type of data you want returned. If you do not want all of the fields in a record type returned, you can specify the individual fields to return by specifying them in the <code>Option.OutputFields</code> option.</p> <ul style="list-style-type: none"> • C—Census • L—Latitude/Longitude • T—Tax Jurisdiction • U—User-defined boundary file • W—Payroll System Tax Codes • X—Auxiliary File • B—PB Software Sales and Use Tax Rate file <p>You can also specify one, and only one, of the following:</p> <p>I Insurance Premium Tax District (IPD)</p> <p>R Payroll Tax District (PAY)</p> <p>S Special Purpose Tax District (SPD)</p> <p>For a description of the fields in each output group, see Response on page 611.</p> <p>Note: If you specify <code>w</code>, to obtain the best payroll system tax code match possible.</p> |

| Parameter | Description |
|---------------------|---|
| Option.TaxKey | <p>If you use third-party tax compliance software from Vertex or Sovos, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following:</p> <p>N Do not return either the Sovos or Vertex jurisdiction codes (default).</p> <p>T Return the Sovos jurisdiction code for the address.</p> <p>V Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Precisely.</p> <p>Note: The Special Purpose District data is needed to achieve the best results from this option.</p> |
| Option.TaxRate | <p>Indicates the sales and use tax rate type to return or none:</p> <p>N Do not return sales and use tax rates. (default)</p> <p>G Return the General sales and use tax rates.</p> <p>A Return the Automotive sales and use tax rates.</p> <p>C Return the Construction sales and use tax rates.</p> <p>M Return the Medical sales and use tax rates.</p> <p>Note: The Special Purpose District data is needed to achieve the best results from this option.</p> |
| Option.OutputFields | <p>Indicates the individual output fields you want returned. You can use this field instead of the Output Record Type to limit the output to those fields that are important to your current data needs.</p> <p>For a list of the fields included in each data type, see Response on page 611.</p> |

Output Format

Output format options control how AssignGeoTAXInfo formats output data.

| Parameter | Description |
|---------------------|--|
| Option.OutputCasing | <p>Specifies the casing of these output fields: County.Name, CBSA.Name, MCD.Name, Place.Name, IPDn.DistrictName, PAYn.DistrictName, SPDn.DistrictName, and PTCn.PayrollDescription.</p> <p>One of the following:</p> <p>M Returns the output in mixed case (default). For example: Rensselaer.</p> <p>U Returns the output in upper case. For example: RENSSELAER.</p> |

Response

Address Match Results

The table below lists the fields returned from the address matching and geocoding process.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose the associated output data options (for example, census or tax jurisdiction data output options). Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------|--|---|
| AddressMatch.DataTypeName* | 20 | <p>Indicates the file from which the match was obtained. One of the following:</p> <ul style="list-style-type: none"> • USPS • TIGER • TOMTOM - Streets • NAVTEQ - Streets • TOMTOM_POINT • CENTRUS_POINT • NAVTEQ_POINT • MASTER LOCATION - Master Location Data • STATE_FILE • USER_AUXILIARY • LANDMARK_AUXILIARY |
| AddressMatch.Firm* | 41 | The name of the business if the address is a business address. |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AddressMatch.GenRC* | 2 | <p>General Return Code indicating the type of match.</p> <p>5 ZIP Code match</p> <p>9 ZIP+4 Code match</p> <p>A User Auxiliary file match</p> <p>C Street Centroid match</p> <p>F Geographic Fallback match</p> <p>G State-supplied file match</p> <p>I Intersection match</p> <p>L Landmark Auxiliary file match</p> <p>M Multiple match (multi-match)</p> <p>O Input Latitude/Longitude coordinates match</p> <p>P Address point match</p> <p>S Street address match</p> <p>U GeoTAX Auxiliary file match</p> <p>X Aborted processing or expired database</p> <p>Blank Did not match</p> |
| AddressMatch.Lastline* | 61 | The complete matched last address line (city, state, and postal code). |
| AddressMatch.LocationCode* | 5 | <p>The Location Code indicates the methodology used to complete the geocode and may also provide some information about the quality of the geocode.</p> <p>For the list of location codes, see Location Codes.</p> |
| AddressMatch.MatchCode* | 5 | <p>The Match Code indicates the portions of the address that matched or did not match to the reference file.</p> <p>For the list of match codes, see Match Codes.</p> |
| AddressMatch.NumCandidates* | 2 | When there are multiple equally-scored matches, returns the number of multiple match candidates found. |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------|--|--|
| AddressMatch.PBKey | 14 | <p>A unique address identifier that is returned when an address match is made using the Master Location Database. The pbKey™ unique identifier is used as a lookup key to a GeoEnrichment database, in order to return attribute data for the match.</p> <p>The AddressMatch.PBKey field has "P" as the leading character, for example: P00001XSF11F.</p> |
| AddressMatch.Urbanization* | 31 | Urbanization name. Used for addresses in Puerto Rico. |
| AddressMatch.Zip* | 6 | The matched address five-digit ZIP Code. |
| AddressMatch.Zip4* | 5 | The matched address four-digit ZIP Code extension. |
| Census.MatchCode* | 2 | <p>The level of match obtained against the databases.</p> <p>5 ZIP Code level match</p> <p>9 ZIP + 4 Code level match</p> <p>A User Auxiliary file match</p> <p>C Street centroid match</p> <p>F Geographic fallback match</p> <p>G State-supplied file match</p> <p>I Intersection match</p> <p>L Landmark Auxiliary file match</p> <p>M Multiple match (multi-match)</p> <p>O Input latitude/longitude coordinates match</p> <p>P Address point match</p> <p>S Street address match</p> <p>U GeoTAX Auxiliary file match</p> <p>X Aborted processing or expired database</p> <p>Blank Did not match</p> |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Census.MatchLevel* | 19 | The level of match obtained against the databases. |
| | | AbortedExpiredData Aborted processing or expired database |
| | | Aux2 GeoTAX Auxiliary file match |
| | | Auxiliary Auxiliary street match |
| | | FallbackGeographic Geographic fallback match |
| | | Gov State file address match |
| | | Intersection Intersection match |
| | | LatLonInput Input latitude/longitude coordinates match |
| | | LandmarkAux Landmark Auxiliary file match |
| | | MultiMatch Multiple match |
| | | Point Address point match |
| | | Street Street address match |
| | | StreetCentroid Street centroid match |
| | | ZIP ZIP Code level match |
| | | ZIP+4 ZIP + 4 Code level match |
| | | NoMatch Did not match |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| Confidence* | 4 | <p>Indicates the confidence in the output provided; from 0 to 100. The higher the score, the higher the confidence in the match. Calculated based on the match results for individual output fields, using the following algorithm:</p> $\text{Census.MatchCode} + \text{LatLong.StreetMatchCode} + \text{LatLong.MatchCode}$ <p>The maximum confidence score is 100, so if this calculation results in a value greater than 100, the Confidence score is returned as 100.</p> <p>The following values are used:</p> <hr/> <ul style="list-style-type: none"> • Census.MatchCode <ul style="list-style-type: none"> • 5 = 45 • 9 = 75 • A = 85 • C = 55 • F = 45 • G = 85 • I = 85 • L = 85 • M = 0 • O = 85 • P = 100 • S = 85 • U = 85 • X = 0 • null = 0 <hr/> <ul style="list-style-type: none"> • LatLong.StreetMatchCode <ul style="list-style-type: none"> • H = 5 • L = 15 • S = -10 • Z = -5 • null = 0 <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|---|--|--|
| | | <ul style="list-style-type: none"> • LatLong.MatchCode <ul style="list-style-type: none"> • 2 = 0 • 4 = 10 • B = 0 • C = 0 • I = 10 • L = 15 • O = 15 • R = 15 • S = -10 • T = -2 • U = 15 • Z = -5 • null = -100 |
| County.Code * | 4 | Extracted from the Census.BlockCode. |
| County.Name * | 26 | Name of the county. |
| GNISCode * | 10 | Unique nine-digit Geographic Names Information System (GNIS) code. |
| Standardized input address fields - for field information, see Input Address on page 628. | | |
| MCD.DistanceToBorder * | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb file. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|---|
| MCD.PointStatus* | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched address point to the polygon defined by the Cousub.txt file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.Code* | 6 | Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.ClassCode* | 3 | Place class code. Place class codes are used to determine the proper taxing jurisdictions |
| Place.DistanceToBorder* | 10 | Returns the distance between the matched address point to the polygon defined by the Place.txt file. |
| Place.IncorporatedFlag* | 8 | <p>Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Inc Incorporated place code.</p> <p>Uninc Unincorporated place code.</p> <p>Unknown Incorporation status unknown.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|---|
| Place.LastAnnexedDate* | 8 | Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information. |
| Place.LastUpdatedDate* | 8 | Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place. |
| Place.LastVerifiedDate* | 8 | Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. |
| Place.Name* | 41 | The name of the "place" where the address is located. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.PointStatus* | 2 | Returns the result for a comparison between the matched address point to the polygon defined by the Place.txb file. For more information on buffers, see Buffering on page 1008. Note: The buffer distance for Place.txb is internally set to zero and cannot be modified. P The point is in the polygon. I The point is in the buffer area inside the polygon. B The point is in the buffer area and outside of the polygon. blank Polygon not found. |
| ProcessedBy* | 4 | Always returns GTX. |
| State.Abbreviation* | 3 | Two-character state abbreviation. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| StateCode* | 3 | Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode. |

Auxiliary File

The table below lists the output fields that contain Auxiliary file data. To include Auxiliary file data in the output, set `Option.GeoTAXOutputRecordType = X`. The following table lists the output fields that contain tax jurisdiction data.

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AuxiliaryData.AuxiliaryFile | 301 | Data retrieved as a result of an auxiliary match from the user-defined area of the auxiliary file. |
| AuxiliaryData.StateFile | 201 | Data retrieved as a result of a state match. Data content and format vary depending on the state file used. |

Census

The census output fields contains census information from the U.S. Census, including Minor Civil Divisions (MCDs) and Census County Division (CCD) names and codes. MCDs are the primary political or administrative divisions of a county, representing many kinds of legal entities with a variety of governmental and administrative functions. CCDs are established in states where there are no legally established MCDs. The Census Bureau recognizes MCDs in 28 states and has established CCDs in 21 states. The District of Columbia has no primary divisions, and the city of Washington, DC is considered equivalent to an MCD for data presentation purposes.

Census data also contains the Federal Information Processing Standards (FIPS) codes for each state and county. The FIPS State Code and the FIPS County Code are both used by the Census Bureau to identify these geographic units.

The table below lists the output fields that contain census data. To include census data in the output, set `Option.GeoTAXOutputRecordType = C`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include census data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|---|
| Census.Block | 4 | Census Block ID. |
| Census.BlockGroup | 2 | Census Block Group code. |
| Census.MatchCode* | 2 | The level of match obtained against the databases. 5 ZIP Code level match 9 ZIP + 4 Code level match A User Auxiliary file match C Street centroid match F Geographic fallback match G State-supplied file match I Intersection match L Landmark Auxiliary file match M Multiple match (multi-match) O Input latitude/longitude coordinates match P Address point match S Street address match U GeoTAX Auxiliary file match X Aborted processing or expired database Blank Did not match |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Census.MatchLevel* | 19 | The level of match obtained against the databases. AbortedExpiredData Aborted processing or expired database Aux2 GeoTAX Auxiliary file match Auxiliary Auxiliary street match FallbackGeographic Geographic fallback match Gov State file address match Intersection Intersection match LatLonInput Input latitude/longitude coordinates match LandmarkAux Landmark Auxiliary file match MultiMatch Multiple match Point Address point match Street Street address match StreetCentroid Street centroid match ZIP ZIP Code level match ZIP+4 ZIP + 4 Code level match NoMatch Did not match |
| Census.Tract | 7 | Six-digit tract number extracted from the Census.BlockCode. |
| County.Code* | 4 | Extracted from the Census.BlockCode. |
| County.Name* | 26 | Name of the county. |
| MCD.Code | 6 | Minor Civil Division/Census County Division (MCD/CCD) Code. |
| MCD.Name | 41 | Minor Civil Division/Census County Division (MCD/CCD) name. |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------|--|---|
| MCD.PointStatus* | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched address point to the polygon defined by the Cousub.txt file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| MCD.DistanceToBorder* | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txt file. |
| MCD.Confidence | 4 | Provides the percentage overlap of the geocode to the MCD polygon layer. The returned percentage value describes the probability that the point falls in the MCD. |
| CBSA.Code | 6 | Core Based Statistical Area (CBSA) code. |
| CBSA.Name | 76 | Core Based Statistical Area (CBSA) name. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| CBSA.MetroFlag | 2 | Indicates if the CBSA is a "Metropolitan Statistical Area" or a "Micropolitan Statistical Area". Y Metropolitan Statistical Area - A Core Based Statistical Area associated with at least one urbanized area that has a population of at least 50,000. The Metropolitan Statistical Area comprises the central county or counties containing the core, plus adjacent outlying counties having a high degree of social and economic integration with the central county as measured through commuting. N Micropolitan Statistical Area - A Core Based Statistical Area associated with at least one urban cluster that has a population of at least 10,000, but less than 50,000. The Micropolitan Statistical Area comprises the central county or counties containing the core, plus adjacent outlying counties having a high degree of social and economic integration with the central county as measured through commuting. |
| CBSAD.Code | 6 | Core Based Statistical Area Division (CBSAD) code. |
| CBSAD.Name | 73 | Core Based Statistical Area Division (CBSAD) name. |
| CSA.Code | 4 | Combined Statistical Area (CSA) code. |
| CSA.Name | 78 | Combined Statistical Area (CSA) name. |
| State.Abbreviation* | 3 | Two-character state abbreviation. |
| StateCode* | 3 | Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode. |

Latitude/Longitude

The table below lists the output fields that contain latitude and longitude data. Latitude/Longitude data contains the coordinates for the address and additional information about how the latitude and

longitude for the address was determined. To include latitude/longitude data in the output, set `Option.GeoTAXOutputRecordType = L`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include latitude/longitude data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| Latitude | 8 | Seven-digit number in degrees and calculated to four decimal places (in the format you specified). |
| Latitude.Directional | 2 | Latitude directional. N North S South |
| LatLong | 23 | Returned latitude/longitude, in the format you specified (up to 22 alphanumeric characters). |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|---|
| LatLong.MatchCode | 2 | Latitude/Longitude General Return Code. Denotes the level for which the geocode was determined. |
| | 2 | ZIP + 2 centroid |
| | 4 | ZIP + 4 Code centroid |
| | B | Block group centroid |
| | C | City centroid |
| | I | Intersection |
| | L | Match using the Landmark Auxiliary file |
| | O | Latitude/longitude was input |
| | R | Address-level based on street address |
| | S | State centroid |
| | T | Census tract centroid |
| | U | Address-level match using the GeoTAX Auxiliary file |
| | Z | ZIP Code centroid based on a five-digit ZIP code |
| | null | No latitude/longitude determined |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | If <code>AddressMatch.GenRC</code> is "P" (point match), then the following are possible values: |
| | 0 | Latitude/Longitude coordinates from User Dictionary. |
| | 2 | Latitude/Longitude coordinates from Parcel Centroid. |
| | 4 | Latitude/Longitude coordinates from Address Point. |
| | 5 | Latitude/Longitude coordinates from Structure Centroid. |
| | 7 | Latitude/Longitude coordinates from manually-placed Point. |
| | 8 | Latitude/Longitude coordinates from Front Door Point. |
| | 9 | Latitude/Longitude coordinates from Driveway Offset Point. |
| | A | Latitude/Longitude coordinates from Street Access Point. |
| | B | Latitude/Longitude coordinates from Base Parcel Point. |
| | C | Latitude/longitude coordinates from Backfill Address Point. |
| | D | Latitude/longitude coordinates from Virtual Address Point. |
| | E | Latitude/longitude coordinates from Interpolated Address Point. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|--|
| LatLong.MatchLevel* | 14 | A description of the value returned in the <code>LatLong.MatchCode</code> field. |
| | | ZIP+2 ZIP + 2 centroid |
| | | ZIP+4 ZIP + 4 centroid |
| | | Block Block group centroid |
| | | CityCentroid City centroid |
| | | Intersection Intersection match |
| | | LandmarkAux Match using the Landmark Auxiliary file |
| | | LatLonInput Input Latitude/Longitude coordinates was used |
| | | Rooftop Exact address match |
| | | StateCentroid State centroid |
| | | Tract Census tract centroid |
| | | Auxiliary Address-level match using the GeoTAX Auxiliary file |
| | | ZIP ZIP Code centroid |
| | | Point Point-level match. One of the following: <ul style="list-style-type: none"> • User Dictionary • Parcel Centroid • Address Point • Structure Centroid • Manually-placed Point • Front Door Point • Driveway Offset Point • Street Access Point • Base Parcel Point • Backfill Address Point • Virtual Address Point • Interpolated Address Point |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------|--|--|
| LatLong.StreetMatchCode* | 2 | Output street address return code. H House number not found on street L Latitude/longitude not determined on auxiliary match S Street not found in ZIP Code Z ZIP Code not found in street address database N Street-level matching option not selected null The street was successfully matched |
| LatLong.StreetMatchLevel* | 16 | Street level match used to determine the latitude/longitude FullMatch Successful match HouseNotFound House number not found on street LatLongNotFound Latitude/longitude not determined on auxiliary match StreetNotFound Street not found in ZIP Code ZipNotFound ZIP Code not found in street address database NotUsed Street-level matching option not selected |
| Longitude | 8 | Seven-digit number in degrees and calculated to four decimal places (in the format specified). |
| Longitude.Directional | 2 | Longitude directional. E East W West |

Input Address

AssignGeoTAXInfo always returns the input address as part of the output. Any changes to the address information resulting from the address cleansing process will be returned to these fields.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---------------------------|
| AddressLine1 | 101 | Input address line 1. |
| AddressLine2 | 101 | Input address line 2. |
| AddressLine3 | 101 | Input address line 3. |
| AddressLine4 | 101 | Input address line 4. |
| City | 51 | Input address city. |
| Country | 25 | Input address country. |
| FirmName | 101 | Input address firm name. |
| PostalCode | 10 | Input address postal code |
| StateProvince | 51 | Input address state. |

Payroll System Tax Code

The table below lists the output fields that contain Payroll System Tax Code (PTC) data. For more information about payroll tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set `Option.GeoTAXOutputRecordType = W`.

Note: `AssignGeoTAXInfo` returns up to six payroll tax codes per address.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------------|--|---|
| NumberPTCsFound | 2 | The number of payroll tax codes found for this address. |
| PTC <i>n</i> .MatchCode | 2 per PTC | Indicates the level of match obtained for the address. In order from most specific match to least, the possible match codes are: P The address was matched to a specific Payroll District ID. This is the most specific match. G The address was matched to a GNIS Code. F The address was matched to a county's FIPS code. S The address was matched to a state's FIPS code. This is the least specific match. |
| PTC <i>n</i> .PayrollCode | 16 per PTC | A code that represents a taxing authority in a payroll application. This is a user-defined code. The specific codes are determined by the payroll application that utilizes the data returned by AssignGeoTAXInfo. |
| PTC <i>n</i> .PayrollDescription | 41 per PTC | A description of the purpose of this payroll code. |
| PTC <i>n</i> .PayrollFlag | 7 per PTC | A user-defined flag from the PTC database. |
| StateCounty | 33 | The state abbreviation and county name. |

Tax Jurisdiction

Tax jurisdiction data contains information about the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state; or, an area recognized as significant because it is located in an incorporated municipality. Places are used to determine tax jurisdiction.

The table below lists the output fields that contain tax jurisdiction data. To include tax jurisdiction data in the output, set `Option.GeoTAXOutputRecordType = T`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| Confidence.SurfaceType | 3 | <p>Indicates the confidence surface type. Setting a non-zero buffer width enables confidence generation. To determine a confidence level, a confidence surface is first generated. The confidence surface provides the smallest possible area wherein an address is likely to be located.</p> <p>0 Undefined</p> <p>1 The search failed - the address was not found.</p> <p>2 Intersection confidence surface generated.</p> <p>3 Interpolated street segment.</p> <p>4 Point-level match.</p> <p>5 State confidence surface generated.</p> <p>6 County confidence surface generated.</p> <p>7 City confidence surface generated.</p> <p>8 Reserved</p> <p>9 A ZIP Code confidence surface generated.</p> <p>10 A ZIP+2 confidence surface generated.</p> <p>11 A ZIP+4 confidence surface generated.</p> <p>12 Reserved</p> <p>13 A street centroid confidence surface generated.</p> |
| GeoTAXKey | 10 | <p>The value in this field varies depending on the option you specified in the <code>Option.TaxKey</code> option:</p> <p>If you specified <code>T</code>, <code>GeoTAXKey</code> contains the proprietary codes used in Sovos tax compliance software. You can use this code in your Sovos application to find out the tax rate for the jurisdiction. The Sovos jurisdiction code formats are as follows:</p> <ul style="list-style-type: none"> • Sovos SUT - 2-digit SUT state code, 5-digit ZIP Code, 2-digit SUT geocode • Sovos TWE - variable-length TWE geocode <p>If you specified <code>V</code>, <code>GeoTAXKey</code> contains the proprietary Vertex[®] jurisdiction code (comprised of a two-digit Vertex[®] state code, three-digit FIPS county code, and four-digit Vertex[®] city code). You can use this code in your Vertex[®] application to find out the tax rate for the jurisdiction.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GeoTAXKey.MatchCode | 2 | <p>Return code denoting the level of match obtained against the Vertex or Sovos cross reference files.</p> <p>E Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name.</p> <p>P Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code.</p> <p>A Alternate match using two fields: ZIP Code, FIPS place name.</p> <p>N Record is default coded based on valid state code.</p> <p>null No matching record found.</p> |
| GeoTAXKey.MatchLevel | 12 | <p>A description of the value returned in the <code>GeoTAXKey.MatchCode</code> field.</p> <p>Exact Exact match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>Partial Partial match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>Alternate Alternate match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>DefaultCode Record is default coded. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>NoMatch No matching record found.</p> |
| GNISCode* | 10 | Unique nine-digit Geographic Names Information System (GNIS) code. |
| Place.ClassCode* | 3 | Place class code. Place class codes are used to determine the proper taxing jurisdictions |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------------|--|--|
| Place.Code * | 6 | Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.IncorporatedFlag * | 8 | Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. Inc Incorporated place code. Uninc Unincorporated place code. Unknown Incorporation status unknown. |
| Place.LastAnnexedDate * | 8 | Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information. |
| Place.LastUpdatedDate * | 8 | Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place. |
| Place.LastVerifiedDate * | 8 | Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. |
| Place.Name * | 41 | The name of the "place" where the address is located. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|--|
| Place.PointStatus* | 2 | <p>Returns the result for a comparison between the matched address point to the polygon defined by the Place.txb file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Place.txb is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.DistanceToBorder* | 10 | Returns the distance between the matched address point to the polygon defined by the Place.txb file. |
| Place.Confidence | 4 | Provides the percentage overlap of the geocode to the Place polygon layer. The returned percentage value describes the probability that the point falls in the specified Place. |

User-Defined Boundary File

The table below lists the output fields that contain data returned from user-defined boundary files. To include this data in the output, set `Option.GeoTAXOutputRecordType = U`.

Note: AssignGeoTAXInfo can return up to 10 user-defined areas for each input address.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|--|
| NumberUserBoundariesFound | 3 | The number of user-defined polygons found for the address. |
| UserBoundary <i>n</i> .BoundaryDescription | 51 per User Boundary | A description of the polygon. |

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| UserBoundary <i>n</i> .BoundaryID | 11 per User Boundary | The ID of the polygon as specified in the user-defined boundary file. |
| UserBoundary <i>n</i> .BufferRelation | 2 per User Boundary | <p>Indicates where in the polygon the address resides in relation to the edge of the area. Buffer width is specified by the option <code>Option.DefaultUserBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the polygon at a distance from the edge that is greater than the specified buffer width. I The address is inside the polygon but is close to the edge. B The address is outside the polygon but is close to the edge. <p>For more information, see Buffering on page 1008.</p> |
| UserBoundary <i>n</i> .DistanceToBorder | 10 per User Boundary | Indicates the distance from the address to the border of the polygon. The distance is in the units specified by the option <code>Option.DistanceUnits</code> . |
| UserBoundary <i>n</i> .SupplementalBoundaryID | 11 per User Boundary | A supplemental ID as specified in the user-defined boundary file. |
| UserBoundary <i>n</i> .BoundaryConfidence | 4 per User Boundary | Provides the percentage overlap of the geocode to the User-defined boundary polygon layer. The returned percentage value describes the probability that the point falls in the User-defined boundary area. |

Insurance Premium Tax Districts

The table below lists the output fields that contain Insurance Premium Tax Districts (IPD) data. To include IPD data in the output, set `Option.GeoTAXOutputRecordType = I`.

Note: `AssignGeoTAXInfo` returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberIPDsFound | 3 | The number of Insurance Premium Tax Districts found for the address |
| IPD <i>n</i> .BoundaryBuffer.BufferRelation | 2 per IPD | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| IPD <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per IPD | Indicates the distance from the address to the border of the district. |
| IPD <i>n</i> .BoundaryConfidence | 4 per IPD | Provides the percentage overlap of the geocode to the IPD boundary polygon layer. The returned percentage value describes the probability that the point falls in the IPD boundary area. |
| IPD <i>n</i> .DistrictID | 11 per IPD | IPD ID. |
| IPD <i>n</i> .DistrictName | 61 per IPD | IPD name. |
| IPD <i>n</i> .DistrictType | 7 per IPD | IPD district type. |
| IPD <i>n</i> .UpdateDate | 7 per IPD | IPD update date (MMYYYY). |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| IPDn.VersionDate | 7 per IPD | IPD compiled date (MMYYYY). |
| IPDn.Notes | 21 per IPD | Tax code descriptions. For example: 01, 33, A, B |
| IPDn.ChangeDate | 7 per IPD | IPD change date. |
| IPDn.EffectiveDate | 7 per IPD | MMDDYY - Identifies when district becomes active - State supplied For example: 010108 |
| IPDn.ExpirationDate | 7 per IPD | MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009 |
| IPDn.FireRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.FireFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semi colon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.CasualtyRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.CasualtyFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.VehicleRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.VehicleFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MarineRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MarineFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.HealthRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.HealthFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.LifeRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.LifeFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.OtherRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.OtherFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MinimumRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.MinimumFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

Parsed Elements

The parsed elements output fields contain standard street address line information as individual units, such as street suffixes (for example AVE, ST, or RD) and leading directionals (for example N and SE).

To include parsed elements in the output, assign the desired output fields to the `Option.OutputFields` parameter.

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|---|
| AddressMatch.UnitType | 5 | The type of unit, such as apartment, suite, or lot. |
| AddressMatch.UnitNumber | 12 | Apartment number. For example, 123 E Main St APT 3 |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|--|
| AddressMatch.HouseNumber | 12 | Building number for the address. |
| AddressMatch.PreDirectional | 3 | Leading directional. For example, 123 E Main St Apt 3 |
| AddressMatch.Street | 41 | The name of the street, not including any directionals or suffixes. For example, the word "Main" in this address: 123 E Main St Apt 3 |
| AddressMatch.StreetType | 5 | The street type of the matched location. For example, AVE for Avenue. |
| AddressMatch.PostDirectional | 3 | Street directional that follows the street name. For example, the "N" in this address: 456 Washington N . |

Payroll Tax Districts

The table below lists the output fields that contain Payroll Tax District (PAY) data. For more information on payroll tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set `Option.GeoTAXOutputRecordType = R`.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--------------------------|
| NumberPAYsFound | 3 | Number of PAYs returned. |

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| PAY <i>n</i> .BoundaryBuffer.BufferRelation | 2 per PAY | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| PAY <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per PAY | Indicates the distance from the address to the border of the district. The distance is in the units specified by the option <code>Option.DefaultBufferWidth</code> . |
| PAY <i>n</i> .BoundaryConfidence | 4 per PAY | Provides the percentage overlap of the geocode to the PAY boundary polygon layer. The returned percentage value describes the probability that the point falls in the PAY boundary area. |
| PAY <i>n</i> .DistrictID | 11 per PAY | PAY district ID. |
| PAY <i>n</i> .DistrictName | 61 per PAY | PAY district name. |
| PAY <i>n</i> .DistrictType | 7 per PAY | PAY district type. |
| PAY <i>n</i> .ID | 11 per PAY | PAY ID. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------------------|--|---|
| PAY <i>n</i> .MunicipalEMSTax | 2 per PAY | <p>PAY municipality emergency municipal services tax.</p> <p>The values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .MunicipalIncomeTax | 2 per PAY | <p>PAY municipality income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X None</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictEMSTax | 2 per PAY | <p>PAY school district emergency municipal services tax.</p> <p>The Values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictIncomeTax | 2 per PAY | <p>PAY school district income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X N</p> <p>The values for Ohio are:</p> <p>R Resident</p> <p>X None</p> <p>All other states are null.</p> |

Special Purpose Tax Districts

The table below lists the output fields that contain Special Purpose Tax Districts (SPD) data. For more information on special purpose tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set `Option.GeoTAXOutputRecordType = S`.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberSPDsFound | 3 | Number of SPDs returned. |
| SPD <i>n</i> .BoundaryBuffer.BufferRelation | 2 per SPD | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| SPD <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per SPD | Indicates the distance from the address to the border of the district. The distance is in the units specified by the option <code>Option.DefaultBufferWidth</code> . |
| SPD <i>n</i> .BoundaryConfidence | 4 per SPD | Provides the percentage overlap of the geocode to the SPD boundary polygon layer. The returned percentage value describes the probability that the point falls in the SPD boundary area. |
| SPD <i>n</i> .CompiledDate | 7 per SPD | SPD compiled date. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|-----------------------------|
| SPD <i>n</i> .DistrictCode | 4 per SPD | 3-digit district type code. |
| SPD <i>n</i> .DistrictName | 61 per SPD | SPD name. |
| SPD <i>n</i> .DistrictNumber | 6 per SPD | SPD district number. |
| SPD <i>n</i> .EffectiveDate | 7 per SPD | SPD effective date. |
| SPD <i>n</i> .UpdateDate | 7 per SPD | SPD update date. |
| SPD <i>n</i> .VersionDate | 7 per SPD | SPD version date. |

Sales and Use Tax Rates

The table below lists the output fields that contain the sales and use tax rate data.

To include tax rate data in the output, set `Option.GeoTAXOutputRecordType = B`.

To select the tax rate type, set `Option.TaxRate` to one of the following:

- N** Do not return sales and use tax rates. (default)
- G** Return the General sales and use tax rates.
- A** Return the Automotive sales and use tax rates.
- C** Return the Construction sales and use tax rates.
- M** Return the Medical sales and use tax rates.

Note: You must be a licensed user of the Precisely Sales and Use Tax Rate file to use this feature.

The following table describes the Sales and Use Tax Rate output fields.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------|--|--|
| TaxRate.RC | 2 | Tax Rate return code denoting the level of match obtained against the Precisely Sales and Use Tax Rate file: E Exact match, using all 5 fields P Partial match, using 4 fields A Alternate match, using 3 fields N Record is default-coded based on valid state code. Blank No matching PB Software Sales and Use Tax Rate record found. |
| Municipal.SalesTaxRate | 11 | Municipality sales tax rate for the selected tax rate type. |
| County.SalesTaxRate | 11 | County sales tax rate for the selected tax rate type. |
| State.SalesTaxRate | 11 | State sales tax rate for the selected tax rate type. |
| SPD <i>n</i> .SalesTaxRate | 11 per SPD | Sales tax rate for up to 10 Special Purpose Districts (SPD). |
| TaxRate.SalesTotal | 11 | The sum of the individual Municipal, County, State and SPD sales tax rates. |
| Municipal.UseTaxRate | 11 | Municipality use tax rate for the selected tax rate type. |
| County.UseTaxRate | 11 | County use tax rate for the selected tax rate type. |
| State.UseTaxRate | 11 | State use tax rate for the selected tax rate type. |
| SPD <i>n</i> .UseTaxRate | 11 per SPD | Use tax rate for up to 10 Special Purpose Districts (SPD). |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| TaxRate.UseTotal | 11 | The sum of the individual Municipal, County, State and SPD use tax rates. |

Error Reporting

The table below defines the error reporting output fields.

Note: Fields denoted by an asterisk "*" are always included in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|--|---|--------------|-----------------------------|----------|-----------------------------|-----------|--------------------------|-----------|---------------------------|----------|---------------|----------|------------------------------------|----------|---------------------------------|----------|-------------------|----------|--------------------|----------|-------------------------------|----------|--|----------|----------------------|----------|----------------------------------|----------|------------------|----------|---------------|
| GTX.ErrorCode* | 3 | <p>This field contains a return code if the GeoTAX engine experiences an abnormal termination.</p> <p>Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <p>The first character indicates the file (or set of files affected).</p> <table border="0"> <tr> <td>Blank</td> <td>Matcher terminated normally</td> </tr> <tr> <td>A</td> <td>User Auxiliary file problem</td> </tr> <tr> <td>CE</td> <td>coubsub.txb file problem</td> </tr> <tr> <td>CI</td> <td>Confidence engine problem</td> </tr> <tr> <td>D</td> <td>Boundary file</td> </tr> <tr> <td>F</td> <td>User-defined boundary file problem</td> </tr> <tr> <td>G</td> <td>Address Matching engine problem</td> </tr> <tr> <td>L</td> <td>Licensing problem</td> </tr> <tr> <td>S</td> <td>State file problem</td> </tr> <tr> <td>U</td> <td>GeoTAX Auxiliary file problem</td> </tr> <tr> <td>X</td> <td>Combination of Street and state file problem</td> </tr> <tr> <td>Z</td> <td>zip.gsb file problem</td> </tr> </table> <p>The second position is one of the following:</p> <table border="0"> <tr> <td>E</td> <td>Fatal issue, program terminating</td> </tr> <tr> <td>F</td> <td>Expired database</td> </tr> <tr> <td>I</td> <td>Informational</td> </tr> </table> | Blank | Matcher terminated normally | A | User Auxiliary file problem | CE | coubsub.txb file problem | CI | Confidence engine problem | D | Boundary file | F | User-defined boundary file problem | G | Address Matching engine problem | L | Licensing problem | S | State file problem | U | GeoTAX Auxiliary file problem | X | Combination of Street and state file problem | Z | zip.gsb file problem | E | Fatal issue, program terminating | F | Expired database | I | Informational |
| Blank | Matcher terminated normally | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | User Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CE | coubsub.txb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CI | Confidence engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | Boundary file | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | User-defined boundary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| G | Address Matching engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| L | Licensing problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| S | State file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| U | GeoTAX Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| X | Combination of Street and state file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Z | zip.gsb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| E | Fatal issue, program terminating | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | Expired database | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| I | Informational | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------|--|--|
| GTX.ErrorDescription* | 81 | <p>If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <hr/> <p>SI-"TS158 FILES NOT FOUND" SI-"TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" SI-"STATE FILES NOT FOUND" SE-"STATE AND TS158 FILES NOT FOUND" SE-"STATE NOT FOUND AND TS158 VINTAGE ERROR" SI-"STATE FILES VINTAGE OR INCOMPLETE DB ERROR" SE-"STATE VINTAGE ERROR AND TS158 NOT FOUND" SE-"STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR"</p> <hr/> <p>GI-"STREET FILES NOT FOUND" XI-"STREET AND TS158 FILES NOT FOUND" XI-"STREET NOT FOUND AND TS158 FILES VINTAGE ERROR" XI-"STREET AND STATE FILES NOT FOUND" XE-"STREET STATE AND TS158 FILES NOT FOUND" XE-"STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR" XI-"STREET NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR"</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | GI-"STREET FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND TS158 NOT FOUND" XI-"STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND STATE NOT FOUND" XE-"STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND" XE-"STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND" XI-"STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR" XE-"STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND" XE-"STREET STATE AND TS158 VINTAGE ERROR" |
| | | LF-"INVALID FUNCTION PASSED TO GTDBLIO : "AI-"GENIO ERROR:FILE = G1GTAUX , FUNC = ,ST = " UI-"GENIO ERROR: FILE =G1GTAX2 , FUNC = , ST = " XF-"The (DB Vintage) database has expired!" XF-"The (SPD file Vintage) SPD File has expired!" |
| | | DI- "UNABLE TO VALIDATE BOUNDARY LICENSE" DI- "UNABLE TO OPEN BOUNDARY FILE" DI- "BOUNDARY FILE NOT FOUND" FI- "UNABLE TO VALIDATE USER BOUNDARY LICENSE" FI- "UNABLE TO OPEN USER BND FILE" FI- "USER BND FILE NOT FOUND" |
| GTX.WarnCode* | 3 | This field contains warning codes returned by the GeoTAX engine. It is blank if no warnings were issued. A value of <code>WN</code> indicates a database will expire next month. <p>Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|---|
| GTX.WarnDescription* | 81 | <p>A text description of any warnings returned by the GeoTAX engine.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> |
| Status | 2 | <p>Reports the success or failure of the match attempt.</p> <p>null Success</p> <p>F Failure. Some examples of failures are your license expired or you did not select any output record types and fields for AssignGeoTAXInfo to return.</p> |
| Status.Code | 12 | <p>If AssignGeoTAXInfo could not process the address, this field will show the reason. Currently there is one possible value for this field: Invalid Address.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Status.Description | 64 | <p>If AssignGeoTAXInfo could not process the address, this field will show a description of the failure. One of the following:</p> <hr/> <p>TS158 FILES NOT FOUND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR STATE FILES NOT FOUND STATE AND TS158 FILES NOT FOUND STATE NOT FOUND AND TS158 VINTAGE ERROR STATE FILES VINTAGE OR INCOMPLETE DB ERROR STATE VINTAGE ERROR AND TS158 NOT FOUND STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR</p> <hr/> <p>STREET FILES NOT FOUND STREET AND TS158 FILES NOT FOUND STREET NOT FOUND AND TS158 FILES VINTAGE ERROR STREET AND STATE FILES NOT FOUND STREET STATE AND TS158 FILES NOT FOUND STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR STREET NOT FOUND AND STATE VINTAGE ERROR STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR STREET FILES VINTAGE OR INCOMPLETE DB ERROR STREET VINTAGE ERROR AND TS158 NOT FOUND</p> <hr/> <p>STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR STREET VINTAGE ERROR AND STATE NOT FOUND STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND STREET STATE AND TS158 VINTAGE ERROR</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| | | INVALID FUNCTION PASSED TO GTDBLIO : GENIO ERROR: FILE = G1GTAUX , FUNC = , ST = GENIO ERROR: FILE = G1GTAX2 , FUNC = , ST = The (DB Vintage) database has expired! The (SPD file Vintage) SPD File has expired! UNABLE TO VALIDATE BOUNDARY LICENSE UNABLE TO OPEN BOUNDARY FILE BOUNDARY FILE NOT FOUND UNABLE TO VALIDATE USER BOUNDARY LICENSE UNABLE TO OPEN USER BND FILE USER BND FILE NOT FOUND |

CalculateDistance

CalculateDistance takes two sets of latitude/longitude coordinates as input, calculates the distance between the coordinates, and returns the distance between the two points.

CalculateDistance is part of Spectrum Enterprise Tax.

Resource URL

JSON endpoint:

```
http://server:port/rest/CalculateDistance/results.json
```

XML endpoint:

```
http://server:port/rest/CalculateDistance/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/CalculateDistance/results.json?
Data.SecondLatitude=41.881833&Option.LatLongFormat=Decimal&
Data.SecondLongitude=-87.785587&Data.FirstLatitude=41.857333&
Data.FirstLongitude=-88.325183
```

The JSON returned by this request would be:

```
{ "output_port": [ {
  "Distance": "27.799",
  "user_fields": [ ]
} ] }
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/CalculateDistance/results.xml?
Data.SecondLatitude=41.881833&Option.LatLongFormat=Decimal&
Data.SecondLongitude=-87.785587&Data.FirstLatitude=41.857333&
Data.FirstLongitude=-88.325183
```

The XML returned by this request would be:

```
<ns2:xml.CalculateDistanceResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/CalculateDistance">
  <ns2:output_port>
    <ns2:Result>
      <ns2:Distance>27.799</ns2:Distance>
      <ns2:user_fields/>
    </ns2:Result>
  </ns2:output_port>
</ns2:xml.CalculateDistanceResponse>
```

Request

Parameters for Input Data

CalculateDistance takes latitude and longitude information as input.

The table below defines the CalculateDistance input data.

| Parameter | Description | | | | |
|--------------------------------|--|----------|-------|----------|-------|
| Data.FirstLatitude | Latitude of the first point for which you want distance returned. | | | | |
| Data.FirstLatitude.Directional | First latitude directional. <table border="0" style="margin-left: 20px;"> <tr> <td>N</td> <td>North</td> </tr> <tr> <td>S</td> <td>South</td> </tr> </table> | N | North | S | South |
| N | North | | | | |
| S | South | | | | |

| Parameter | Description |
|----------------------------------|---|
| Data.FirstLongitude | Longitude of the first point for which you want distance returned. |
| Data.FirstLongitude.Directional | First longitude directional. E East W West |
| Data.SecondLatitude | Latitude of the second point for which you want distance returned. |
| Data.SecondLatitude.Directional | Second latitude directional. N North S South |
| Data.SecondLongitude | Longitude of the second point for which you want distance returned. |
| Data.SecondLongitude.Directional | Second longitude directional. E East W West |

Parameters for Options

The table below defines the output data and format options.

| Parameter | Description |
|----------------------|--|
| Option.LatLongFormat | <p>Indicates the format of the input latitude/longitude. The options are:</p> <p>DegMinSec For example 90 00 00N180 00 00W.</p> <p>PreZero Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W. (default)</p> <p>PreZeroDecimal Decimal degrees using directional indicator. For example, 090.000000N180.000000W.</p> <p>Decimal Decimal degrees using signed latitude/longitude. For example, 90.000000-180.000000.</p> <p>DecimalAssumed Decimal degrees using signed latitude/longitude (no decimal point). For example, 90000000-180000000.</p> |
| Option.ReturnUnits | <p>Indicates the measurement units returned for distance calculation:</p> <ul style="list-style-type: none"> • Feet • Km • Meters • Miles (default) |

Response

CalculateDistance always returns the Confidence field to indicate the confidence in the output provided.

If CalculateDistance fails to process the data, it returns the fields Status, Status.Code, and Status.Descriptions. These fields provide information on why CalculateDistance failed to process the data. Some examples of failures are your license expired or you did not select any output record types and fields for CalculateDistance to return. The following table provides the record-level qualifiers and data outputs for CalculateDistance.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| Distance | 9 | Distance between the two input coordinates in the units of measurement that you specified. |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|---|
| Status | 2 | Reports the success or failure of the match attempt: null Success F Failure |
| Status.Code | 2 | Reason for failure or error. If Status = F, Status.Code = Failure. |
| Status.Description | 64 | Description of the problem. If Status = F, Status.Description = Unable to compute distance. |

ReverseGeoTAXInfoLookup

ReverseGeoTAXInfoLookup allows latitude/longitude coordinates to be supplied as input and identifies the tax districts that apply to the given coordinate. Specifically, ReverseGeoTAXInfoLookup can return the following information about a location:

- FIPS state codes and county codes
- State and county names
- MCD codes and names
- Place codes and names
- Boundary file districts
- Cross-reference tax keys
- The relationship of the input coordinates to user-defined polygons
- Sales and use tax rates, if licensed for the Precisely Sales and Use Tax Rate File

ReverseGeoTAXInfoLookup optionally includes enhanced tax jurisdiction information for a location, including:

- **Insurance premium districts**—Areas designated for the collection of taxes imposed on insurance policy premiums, based on the policy holder's address. Insurance premium districts are created by state governments.
- **Payroll tax districts**—Areas designated for the collection of taxes imposed on employers to support state or local government facilities and services, based on the employee's and/or employer's address. Examples include taxes collected for districts to pay for schools, police, or other services. Payroll tax districts are created by state or local governments.

- **Payroll system tax codes**—Codes that represent specific jurisdictions that collect payroll tax. Using payroll system tax codes has advantages over using the payroll tax district information returned by ReverseGeoTAXInfoLookup:
 - ReverseGeoTAXInfoLookup uses an additional database to determine payroll tax codes, resulting in more accurate payroll tax determination.
 - Many payroll systems use specific codes to determine withholding amounts. Since you can customize the payroll tax codes returned by ReverseGeoTAXInfoLookup, you can set up a process where ReverseGeoTAXInfoLookup returns the exact payroll tax codes required by your payroll system, instead of returning jurisdictional IDs that must then be translated into the codes used by your system.
- **Special purpose tax districts**—Areas designated for the collection of taxes imposed on residents to support specialized services for residents of the district, based on the resident's address. Examples include services such as sewer service, transit service, or water resources. Special purpose tax districts are created by legislative action, court action, or public referendums. This optional information requires the use of boundary files which require an additional license. Contact your Precisely sales representative for more information.

Using the optional Precisely Sales and Use Tax Rate file, ReverseGeoTAXInfoLookup includes tax rate data for a location, including:

Tax rate type:

- General
- Automotive
- Medical
- Construction

Sales and/or use tax rates for:

- State
- County
- Municipality
- Up to 10 SPDs
- Total Rate - the sum of the individual state, county, municipality and SPD rates.

Required input format

The required format for the input coordinates is as follows:

| Response Element | Format |
|--------------------|---|
| Data.InputLatitude | 00.000000 or without the decimal point 00000000 |

| Response Element | Format |
|---------------------|--|
| Data.InputLongitude | 000.000000 or without the decimal point 000000000, or 00.000000 or without the decimal point 00000000 |

ReverseGeoTAXInfoLookup is part of Spectrum Enterprise Tax.

Resource URL

JSON endpoint:

```
http://server:port/rest/ReverseGeoTAXInfoLookup/results.json
```

XML endpoint:

```
http://server:port/rest/ReverseGeoTAXInfoLookup/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/ReverseGeoTAXInfoLookup/results.json
?Data.InputLatitude=40.018998&Data.InputLongitude=-105.239580
```

The JSON returned by this request would be:

```
{
  "output_port": [
    {
      "Confidence": "100.0",
      "ProcessedBy": "GTX",
      "County.Code": "013",
      "County.Name": "Boulder",
      "StateCode": "08",
      "InputLatitude": "40.018998",
      "InputLongitude": "-105.239580",
      "State.Abbreviation": "CO",
      "Place.ClassCode": "C1",
      "Place.Code": "07850",
      "Place.IncorporatedFlag": "Inc",
      "Place.Name": "Boulder",
      "Place.LastAnnexedDate": "10/2011",
      "Place.LastUpdatedDate": "04/2013",
      "Place.LastVerifiedDate": "01/2013",
      "Place.DistanceToBorder": "000000387",
      "Place.PointStatus": "P",
      "GNISCode": "002409883",
      "GTX.ErrorCode": ""
    }
  ]
}
```

```
"GTX.ErrorDescription": "",
"GTX.WarnCode": "",
"GTX.WarnDescription": ""
}}}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/ReverseGeoTAXInfoLookup/results.xml
?Data.InputLatitude=40.018998&Data.InputLongitude=-105.239580
```

The XML returned by this request would be:

```
ns2:xml.ReverseGeoTAXInfoLookup
xmlns:ns2="http://www.precisely.com/spectrum/services/ReverseGeoTAXInfoLookup">

  <ns2:output_port>
    <ns2:Address>
      <ns2:Confidence>100.0</ns2:Confidence>
      <ns2:ProcessedBy>GTX</ns2:ProcessedBy>
      <ns2:County.Code>013</ns2:County.Code>
      <ns2:County.Name>Boulder</ns2:County.Name>
      <ns2:StateCode>08</ns2:StateCode>
      <ns2:InputLatitude>40.018998</ns2:InputLatitude>
      <ns2:InputLongitude>-105.239580</ns2:InputLongitude>
      <ns2:State.Abbreviation>CO</ns2:State.Abbreviation>
      <ns2:Place.ClassCode>C1</ns2:Place.ClassCode>
      <ns2:Place.Code>07850</ns2:Place.Code>
      <ns2:Place.IncorporatedFlag>Inc</ns2:Place.IncorporatedFlag>
      <ns2:Place.Name>Boulder</ns2:Place.Name>
      <ns2:Place.LastAnnexedDate>10/2011</ns2:Place.LastAnnexedDate>

      <ns2:Place.LastUpdatedDate>04/2013</ns2:Place.LastUpdatedDate>

      <ns2:Place.LastVerifiedDate>01/2013</ns2:Place.LastVerifiedDate>

    <ns2:Place.DistanceToBorder>000000387</ns2:Place.DistanceToBorder>
      <ns2:Place.PointStatus>P</ns2:Place.PointStatus>
      <ns2:GNISCode>002409883</ns2:GNISCode>
      <ns2:GTX.ErrorCode>""</ns2:GTX.ErrorCode>
      <ns2:GTX.ErrorDescription>""</ns2:GTX.ErrorDescription>
      <ns2:GTX.WarnCode>""</ns2:GTX.WarnCode>
      <ns2:GTX.WarnDescription>""</ns2:GTX.WarnDescription>
    </ns2:Address>
  </ns2:output_port>
</ns2:xml.ReverseGeoTAXInfoLookup>
```

Request

Geocoding Options

Reverse geocoding information lookup is the process of taking an input latitude/longitude coordinate and returning jurisdictional tax information. The geocoding options specify the distance units and buffer distance to use when matching to a boundary file.

| Parameter | Description |
|---------------------|--|
| Option.Database.GTX | Select the database resource to use in the reverse geocoding lookup process. |

Boundary matching: The following options can be set when matching to a boundary file such as SPD, IPD, PAY, Place and MCD, or user-defined.

| | | | | | |
|----------------------|---|-------------|---|---------------|-----------------------------------|
| Option.DistanceUnits | Specifies the units in which to measure distance. One of the following: <table border="0"> <tr> <td>Feet</td> <td>Distances are measured in feet. (default)</td> </tr> <tr> <td>Meters</td> <td>Distances are measured in meters.</td> </tr> </table> | Feet | Distances are measured in feet. (default) | Meters | Distances are measured in meters. |
| Feet | Distances are measured in feet. (default) | | | | |
| Meters | Distances are measured in meters. | | | | |

Default buffer widths

| | |
|-------------------------------|--|
| Option.DefaultBufferWidth | <p>Specifies the buffer width to use for tax district boundary files. The tax district boundary files are the Special Purpose Districts (SPD) file, the Insurance Premium Districts (IPD) file, the Payroll Tax Districts (PAY) file, and Place and MCD files.</p> <p>The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>For more information about buffers, see Buffering on page 1008.</p> |
| Option.DefaultUserBufferWidth | <p>Specifies the buffer width to use for user-defined boundary files. The distance is in the units of measurement specified in the Distance units option. For information about buffers, see Buffering on page 1008. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>Note: To use buffers, the user-defined boundary file must support buffers.</p> |

Output Data Options

Data options control the data returned by ReverseGeoTAXInfoLookup.

| Parameter | Description |
|-------------------------------|---|
| Option.GeoTAXOutputRecordType | <p>Select one or more of the following to obtain the type of data you want returned. ReverseGeoTAXInfo Lookup groups the output fields into record types. If you do not want all of the fields in a record type returned, do not select the check box, and list only those fields you want returned in <code>Option.OutputFields</code>.</p> <ul style="list-style-type: none"> • C—Census • T—Tax Jurisdiction • U—User-defined boundary file • W—Payroll System Tax Codes • X—Auxiliary File • B—PB Software Sales and Use Tax Rate file <p>You can also specify one, and only one, of the following:</p> <p>I Insurance Premium Tax District (IPD)</p> <p>R Payroll Tax District (PAY)</p> <p>S Special Purpose Tax District (SPD)</p> <p>For a description of the fields in each output group, see Response on page 661.</p> <p>Note: If you specify W, also specify R to obtain the best payroll system tax code match possible.</p> |
| Option.TaxKey | <p>If you integrate ReverseGeoTAXInfo Lookup with third-party tax compliance software from Vertex or Sovos, select which vendor you use. This controls the value returned in the <code>GeoTAXKey</code> output field. One of the following:</p> <p>N Do not return either the Sovos or Vertex jurisdiction codes (default).</p> <p>T Return the Sovos jurisdiction code for the address.</p> <p>V Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Precisely.</p> |
| Option.TaxRate | <p>Select the desired Sales tax rate type or none:</p> <p>N Do not return sales tax rates. (default)</p> <p>G Return the General sales tax rates.</p> <p>A Return the Automotive sales tax rates.</p> <p>C Return the Construction sales tax rates.</p> <p>Return the Medical sales tax rates.</p> |

| Parameter | Description |
|---------------------|--|
| Option.OutputFields | <p>Indicates the individual output fields you want returned. You can use this field instead of the Output Record Type to limit the output to those fields that are important to your current data needs.</p> <p>For a list of the fields included in each data type, see Response on page 661.</p> |

Output Format

Output format options control how ReverseGeoTAXInfo Lookup formats output data.

| Parameter | Description |
|---------------------|--|
| Option.OutputCasing | <p>Specifies the casing of these output fields: <code>County.Name</code>, <code>MCD.Name</code>, <code>Place.Name</code>, <code>IPDn.DistrictName</code>, <code>PAYn.DistrictName</code>, <code>SPDn.DistrictName</code>, and <code>PTCn.PayrollDescription</code>.</p> <p>One of the following:</p> <p>M Returns the output in mixed case (default). For example: Boulder.</p> <p>U Returns the output in upper case. For example: BOULDER.</p> |

Response

Auxiliary File

The table below lists the output fields that contain Auxiliary file data. To include Auxiliary file data in the output, set `Option.GeoTAXOutputRecordType = X`. The following table lists the output fields that contain tax jurisdiction data.

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AuxiliaryData.AuxiliaryFile | 301 | Data retrieved as a result of an auxiliary match from the user-defined area of the auxiliary file. |
| AuxiliaryData.StateFile | 201 | Data retrieved as a result of a state match. Data content and format vary depending on the state file used. |

Census

The census output fields contains census information from the U.S. Census, including Minor Civil Divisions (MCDs) and Census County Division (CCD) names and codes. MCDs are the primary political or administrative divisions of a county, representing many kinds of legal entities with a variety of governmental and administrative functions. CCDs are established in states where there are no legally established MCDs. The Census Bureau recognizes MCDs in 28 states and has established CCDs in 21 states. The District of Columbia has no primary divisions, and the city of Washington, DC is considered equivalent to an MCD for data presentation purposes.

Census data also contains the Federal Information Processing Standards (FIPS) codes for each state and county. The FIPS State Code and the FIPS County Code are both used by the Census Bureau to identify these geographic units.

The following table lists the output fields that contain census data. To include census data in the output, set `Option.GeoTAXOutputRecordType = C`.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| County.Code | 4 | Three-digit Federal Information Processing Standards (FIPS) county code extracted from the Census.BlockCode. Note: This field is always included in the output regardless of whether or not you choose to include census data in the output. |
| County.Name | 26 | Name of the county. Note: This field is always included in the output regardless of whether or not you choose to include census data in the output. |
| MCD.Code | 6 | Minor Civil Division/Census County Division (MCD/CCD) Code. |
| MCD.Name | 41 | Minor Civil Division/Census County Division (MCD/CCD) name. |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|---|
| MCD.PointStatus | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched geocode location to the polygon defined by the Cousub.txt file.</p> <p>For more information about buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| MCD.DistanceToBorder | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txt file. |
| StateCode | 3 | <p>Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include census data in the output.</p> |

Input Latitude/Longitude

ReverseGeoTAXInfoLookup always returns the input coordinates as part of the output. The input latitude/longitude fields are returned as input from the data. ReverseGeoTAXInfoLookup does not change these input values.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|-----------------|
| InputLatitude | 12 | Input latitude. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|------------------|
| InputLongitude | 12 | Input longitude. |

Payroll System Tax Code

The following table lists the output fields that contain Payroll System Tax Code (PTC) data. For more information on payroll tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `Option.GeoTAXOutputRecordType = W`.

Note: ReverseGeoTAXInfoLookup returns up to six payroll tax codes per input location.

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|--|
| NumberPTCsFound | 2 | The number of payroll system tax codes found for this location. |
| PTCn.MatchCode | 2 per PTC | <p>Indicates the level of match obtained for the location. In order from most specific match to least, the possible match codes are:</p> <ul style="list-style-type: none"> P The address was matched to a specific Payroll District ID. This is the most specific match. G The address was matched to a GNIS Code. F The address was matched to a county's FIPS code. S The address was matched to a state's FIPS code. This is the least specific match. |
| PTCn.PayrollCode | 16 per PTC | A code that represents a taxing authority in a payroll application. This is a user-defined code. The specific codes are determined by the payroll application that utilizes the data returned by ReverseGeoTAXInfo Lookup. |
| PTCn.PayrollDescription | 41 per PTC | A description of the purpose of this payroll code. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| PTCn.PayrollFlag | 7 per PTC | A user-defined flag from the PTC database. |
| StateCounty | 33 | The state abbreviation and county name. |

Tax Jurisdiction

Tax jurisdiction data contains information about the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state; or, an area recognized as significant because it is located in an incorporated municipality. Places are used to determine tax jurisdiction.

The following table lists the output fields that contain tax jurisdiction data. To include tax jurisdiction data in the output, set `Option.GeoTAXOutputRecordType = T`.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| GeoTAXKey | 10 | <p>The value in this field varies depending on the option you specified in the <code>Option.TaxKey</code> option:</p> <p>If you specified <code>T</code>, <code>GeoTAXKey</code> contains the proprietary codes used in Sovos tax compliance software. You can use this code in your Sovos application to find out the tax rate for the jurisdiction. The Sovos jurisdiction code formats are as follows:</p> <ul style="list-style-type: none"> • Sovos SUT - 2-digit SUT state code, 5-digit ZIP Code, 2-digit SUT geocode • Sovos TWE - variable-length TWE geocode <p>If you specified <code>V</code>, <code>GeoTAXKey</code> contains the proprietary Vertex[®] jurisdiction code (comprised of a two-digit Vertex[®] state code, three-digit FIPS county code, and four-digit Vertex[®] city code). You can use this code in your Vertex[®] application to find out the tax rate for the jurisdiction.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GeoTAXKey.MatchCode | 2 | <p>Return code denoting the level of match obtained against the Vertex or Sovos cross reference files.</p> <p>E Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name.</p> <p>P Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code.</p> <p>A Alternate match using two fields: ZIP Code, FIPS place name.</p> <p>N Record is default coded based on valid state code.</p> <p>null No matching record found.</p> |
| GeoTAXKey.MatchLevel | 12 | <p>A description of the value returned in the GeoTAXKey.MatchCode field.</p> <p>Exact Exact match. See description in GeoTAXKey.MatchCode.</p> <p>Partial Partial match. See description in GeoTAXKey.MatchCode.</p> <p>Alternate Alternate match. See description in GeoTAXKey.MatchCode.</p> <p>DefaultCode Record is default coded. See description in GeoTAXKey.MatchCode.</p> <p>NoMatch No matching record found.</p> |
| GNISCode | 10 | <p>Unique nine-digit Geographic Names Information System (GNIS) code.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include census data in the output.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|---|
| Place.ClassCode | 3 | <p>Place class code. Place class codes are used to determine the proper taxing jurisdictions</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.Code | 6 | <p>Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.IncorporatedFlag | 8 | <p>Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> <p>Inc Incorporated place code.</p> <p>Uninc Unincorporated place code.</p> <p>Unknown Incorporation status unknown.</p> |
| Place.LastAnnexedDate | 8 | <p>Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| Place.LastUpdatedDate | 8 | <p>Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.LastVerifiedDate | 8 | <p>Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.Name | 41 | <p>The name of the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.PointStatus | 2 | <p>Returns the result for a comparison between the matched geocode location to the polygon defined by the Place.txb file. For more information on buffers, see Buffering on page 1008.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.DistanceToBorder | 10 | <p>Returns the distance in feet between the matched address point to the polygon defined by the Place.txb file.</p> |

User-Defined Boundary File

The following table lists the output fields that contain data returned from user-defined boundary files. To include this data in the output, set `Option.GeoTAXOutputRecordType = U`.

Note: ReverseGeoTAXInfoLookup can return up to 10 user-defined areas for each input location.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|--|
| NumberUserBoundariesFound | 3 | The number of user-defined polygons found for the address. |
| UserBoundary <i>n</i> .BoundaryDescription | 51 per User Boundary | A description of the polygon. |
| UserBoundary <i>n</i> .BoundaryID | 11 per User Boundary | The ID of the polygon as specified in the user-defined boundary file. |
| UserBoundary <i>n</i> .BufferRelation | 2 per User Boundary | <p>Indicates where in the polygon the location resides in relation to the edge of the area. Buffer width is specified by the option <code>Option.DefaultUserBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The geocode is inside the polygon at a distance from the edge that is greater than the specified buffer width. I The geocode is inside the polygon but is close to the edge. This indicates that the address is in the buffer area. B The geocode is outside the polygon but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| UserBoundary <i>n</i> .DistanceToBorder | 10 per User Boundary | Indicates the distance in feet from the input location to the border of the polygon. |

| Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| UserBoundary n .SupplementalBoundaryID | 11 per User Boundary | A supplemental ID as specified in the user-defined boundary file. |

Insurance Premium Tax Districts

The following table lists the output fields that contain Insurance Premium Tax Districts (IPD) data. For more information on insurance premium tax districts, see [ReverseGeoTAXInfoLookup](#) on page 656. To include IPD data in the output, set `Option.GeoTAXOutputRecordType = I`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| NumberIPDsFound | 3 | The number of Insurance Premium Tax Districts found for the location. |
| IPD n .BoundaryBuffer.BufferRelation | 2 per IPD | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| IPD n .BoundaryBuffer.DistanceToBorder | 10 per IPD | Indicates the distance in feet from the location to the border of the district. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| IPDn.DistrictID | 11 per IPD | IPD ID. |
| IPDn.DistrictName | 61 per IPD | IPD name. |
| IPDn.DistrictType | 7 per IPD | IPD district type. |
| IPDn.UpdateDate | 7 per IPD | IPD update date (MMYYYY). |
| IPDn.VersionDate | 7 per IPD | IPD compiled date (MMYYYY). |
| IPDn.Notes | 21 per IPD | Tax code descriptions. For example: 01, 33, A, B |
| IPDn.ChangeDate | 7 per IPD | IPD change date. |
| IPDn.EffectiveDate | 7 per IPD | MMDDYY - Identifies when district becomes active - State supplied For example: 010108 |
| IPDn.ExpirationDate | 7 per IPD | MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009 |
| IPDn.FireRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.FireFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semi colon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|--|
| IPDn.CasualtyRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.CasualtyFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.VehicleRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.VehicleFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MarineRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MarineFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.HealthRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.HealthFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.LifeRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.LifeFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.OtherRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.OtherFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MinimumRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MinimumFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

Payroll Tax Districts

The following table lists the output fields that contain Payroll Tax District (PAY) data. For more information on payroll tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `Option.GeoTAXOutputRecordType = R`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberPAYsFound | 3 | Number of payroll tax districts found for the location. |
| PAY <i>n</i> .BoundaryBuffer.BufferRelation | 2 per PAY | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| PAY <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per PAY | Indicates the distance in feet from the location to the border of the district. |
| PAY <i>n</i> .DistrictID | 11 per PAY | PAY district ID. |
| PAY <i>n</i> .DistrictName | 61 per PAY | PAY district name. |
| PAY <i>n</i> .DistrictType | 7 per PAY | PAY district type. |
| PAY <i>n</i> .ID | 11 per PAY | PAY ID. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------------------|--|---|
| PAY <i>n</i> .MunicipalEMSTax | 2 per PAY | <p>PAY municipality emergency municipal services tax.</p> <p>The values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .MunicipalIncomeTax | 2 per PAY | <p>PAY municipality income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X None</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictEMSTax | 2 per PAY | <p>PAY school district emergency municipal services tax.</p> <p>The Values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictIncomeTax | 2 per PAY | <p>PAY school district income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X N</p> <p>The values for Ohio are:</p> <p>R Resident</p> <p>X None</p> <p>All other states are null.</p> |

Special Purpose Tax Districts

The following table lists the output fields that contain Special Purpose Tax Districts (SPD) data. For more information on special purpose tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `Option.GeoTAXOutputRecordType = S`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberSPDsFound | 3 | Number of Special Purpose Tax Districts found for the location. |
| SPD <i>n</i> .BoundaryBuffer.BufferRelation | 2 per SPD | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>Option.DefaultBufferWidth</code> or by the input field <code>Data.BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| SPD <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per SPD | Indicates the distance in feet from the address to the border of the district. |
| SPD <i>n</i> .CompiledDate | 7 per SPD | SPD compiled date. |
| SPD <i>n</i> .DistrictCode | 4 per SPD | 3-digit district type code. |
| SPD <i>n</i> .DistrictName | 61 per SPD | SPD name. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|----------------------|
| SPD <i>n</i> .DistrictNumber | 6 per SPD | SPD district number. |
| SPD <i>n</i> .EffectiveDate | 7 per SPD | SPD effective date. |
| SPD <i>n</i> .UpdateDate | 7 per SPD | SPD update date. |
| SPD <i>n</i> .VersionDate | 7 per SPD | SPD version date. |

Sales and Use Tax Rates

The table below lists the output fields that contain the sales and use tax rate data.

To include tax rate data in the output, set `Option.GeoTAXOutputRecordType = B`.

To select the tax rate type, set `Option.TaxRate` to one of the following:

- N** Do not return sales and use tax rates. (default)
- G** Return the General sales and use tax rates.
- A** Return the Automotive sales and use tax rates.
- C** Return the Construction sales and use tax rates.
- M** Return the Medical sales and use tax rates.

Note: You must be a licensed user of the Precisely Sales and Use Tax Rate file to use this feature.

The following table describes the Sales and Use Tax Rate output fields.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|---|
| TaxRate.RC | 2 | <p>Tax Rate return code denoting the level of match obtained against the Precisely Sales and Use Tax Rate file:</p> <p>E Exact match, using all 5 fields</p> <p>P Partial match, using 4 fields</p> <p>A Alternate match, using 3 fields</p> <p>N Record is default-coded based on valid state code.</p> <p>Blank No matching PB Software Sales and Use Tax Rate record found.</p> |
| Municipal.SalesTaxRate | 11 | Municipality sales tax rate for the selected tax rate type. |
| County.SalesTaxRate | 11 | County sales tax rate for the selected tax rate type. |
| State.SalesTaxRate | 11 | State sales tax rate for the selected tax rate type. |
| SPD n .SalesTaxRate | 11 per SPD | Sales tax rate for up to 10 Special Purpose Districts (SPD). |
| TaxRate.SalesTotal | 11 | The sum of the individual Municipal, County, State and SPD sales tax rates. |
| Municipal.UseTaxRate | 11 | Municipality use tax rate for the selected tax rate type. |
| County.UseTaxRate | 11 | County use tax rate for the selected tax rate type. |
| State.UseTaxRate | 11 | State use tax rate for the selected tax rate type. |
| SPD n .UseTaxRate | 11 per SPD | Use tax rate for up to 10 Special Purpose Districts (SPD). |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| TaxRate.UseTotal | 11 | The sum of the individual Municipal, County, State and SPD use tax rates. |

Error Reporting

The table below defines the error reporting output fields.

| Response Element | Max. Field Length with null terminator | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|--|---|--------------|-----------------------------|----------|-----------------------------|-----------|--------------------------|-----------|---------------------------|----------|---------------|----------|------------------------------------|----------|---------------------------------|----------|-------------------|----------|--------------------|----------|-------------------------------|----------|--|----------|----------------------|----------|----------------------------------|----------|------------------|----------|---------------|
| GTX.ErrorCode | 3 | <p>This field contains a return code if the GeoTAX engine experiences an abnormal termination.</p> <p>Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <p>The first character indicates the file (or set of files affected).</p> <table border="0"> <tr> <td>Blank</td> <td>Matcher terminated normally</td> </tr> <tr> <td>A</td> <td>User Auxiliary file problem</td> </tr> <tr> <td>CE</td> <td>coubsub.txb file problem</td> </tr> <tr> <td>CI</td> <td>Confidence engine problem</td> </tr> <tr> <td>D</td> <td>Boundary file</td> </tr> <tr> <td>F</td> <td>User-defined boundary file problem</td> </tr> <tr> <td>G</td> <td>Address Matching engine problem</td> </tr> <tr> <td>L</td> <td>Licensing problem</td> </tr> <tr> <td>S</td> <td>State file problem</td> </tr> <tr> <td>U</td> <td>GeoTAX Auxiliary file problem</td> </tr> <tr> <td>X</td> <td>Combination of Street and state file problem</td> </tr> <tr> <td>Z</td> <td>zip.gsb file problem</td> </tr> </table> <p>The second position is one of the following:</p> <table border="0"> <tr> <td>E</td> <td>Fatal issue, program terminating</td> </tr> <tr> <td>F</td> <td>Expired database</td> </tr> <tr> <td>I</td> <td>Informational</td> </tr> </table> | Blank | Matcher terminated normally | A | User Auxiliary file problem | CE | coubsub.txb file problem | CI | Confidence engine problem | D | Boundary file | F | User-defined boundary file problem | G | Address Matching engine problem | L | Licensing problem | S | State file problem | U | GeoTAX Auxiliary file problem | X | Combination of Street and state file problem | Z | zip.gsb file problem | E | Fatal issue, program terminating | F | Expired database | I | Informational |
| Blank | Matcher terminated normally | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | User Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CE | coubsub.txb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CI | Confidence engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | Boundary file | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | User-defined boundary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| G | Address Matching engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| L | Licensing problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| S | State file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| U | GeoTAX Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| X | Combination of Street and state file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Z | zip.gsb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| E | Fatal issue, program terminating | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | Expired database | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| I | Informational | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GTX.ErrorDescription | 81 | <p>If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally. The maximum length is 80.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <hr/> <p>SI-"TS158 FILES NOT FOUND" SI-"TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" SI-"STATE FILES NOT FOUND" SE-"STATE AND TS158 FILES NOT FOUND" SE-"STATE NOT FOUND AND TS158 VINTAGE ERROR" SI-"STATE FILES VINTAGE OR INCOMPLETE DB ERROR" SE-"STATE VINTAGE ERROR AND TS158 NOT FOUND" SE-"STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR"</p> <hr/> <p>GI-"STREET FILES NOT FOUND" XI-"STREET AND TS158 FILES NOT FOUND" XI-"STREET NOT FOUND AND TS158 FILES VINTAGE ERROR" XI-"STREET AND STATE FILES NOT FOUND" XE-"STREET STATE AND TS158 FILES NOT FOUND" XE-"STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR" XI-"STREET NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR"</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | GI-"STREET FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND TS158 NOT FOUND" XI-"STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND STATE NOT FOUND" XE-"STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND" XE-"STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND" XI-"STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR" XE-"STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND" XE-"STREET STATE AND TS158 VINTAGE ERROR" |
| | | LF-"INVALID FUNCTION PASSED TO GTDBLIO : " AI-"GENIO ERROR: FILE = G1GTAUX , FUNC = , ST = " UI-"GENIO ERROR: FILE = G1GTAX2 , FUNC = , ST = " XF-"The (DB Vintage) database has expired!" XF-"The (SPD file Vintage) SPD File has expired!" |
| | | DI- "UNABLE TO VALIDATE BOUNDARY LICENSE" DI- "UNABLE TO OPEN BOUNDARY FILE" DI- "BOUNDARY FILE NOT FOUND" FI- "UNABLE TO VALIDATE USER BOUNDARY LICENSE" FI- "UNABLE TO OPEN USER BND FILE" FI- "USER BND FILE NOT FOUND" |
| GTX.WarnCode | 3 | This field contains warning codes returned by the GeoTAX engine. It is blank if no warnings were issued. A value of <code>WN</code> indicates a database will expire next month. Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|--|
| GTX.WarnDescription | 81 | A text description of any warnings returned by the GeoTAX engine. Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform. |

Match and Location Codes

Match Codes

Match Codes indicate the portions of the address that matched or did not match to the reference file. If a match could not be made, the Match Code begins with "E" and the remaining digits indicate why the address did not match (see [Match Codes for No Match - Definitions for "Ennn" return codes](#) on page 157). The digits do not specifically refer to which address elements did not match, but rather why the address did not match. These fields are always included in the output from AssignGeoTAXInfo.

Match Code Definitions

| Response Element | Description |
|------------------|---|
| Ahhh | Same as Shhh, but indicates match to an alias name record or an alternate record. |
| Chh | Street address did not match, but 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 | House number was changed. |
| 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. |

| Response Element | Description |
|------------------|---|
| 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. |
| 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. |
| Z | No address given, but verified the provided ZIP Code. |

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: 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 154.

| 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. |

| Code | In first hex position means: | In second and third hex position means: |
|------|------------------------------------|---|
| 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. |
| 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. |

| Code | In first hex position means: | In second and third hex position means: |
|------|--|--|
| 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 additional information is returned 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, Q, R, S, T or U (see [Match Codes](#) on page 151).

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 152
- Extended Match Codes, see the table below.

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

- The output address has extra information (for example, a mailstop) in the address line.
- The output address has a second address line (`AddressLine2`).
- 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. |
| 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 moved to <code>AddressLine2</code> . |

| Input Addressline | Output Addressline | Extended Code | Description |
|--|------------------------|---------------|--|
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |

| 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 moved to <code>AddressLine2</code> . |
| 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 moved to <code>AddressLine2</code> . |

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

Note: For Landmark Auxiliary file matches, the 3rd hex digit is always "0".

| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 2 | Matched on Unit Number. Unit Type changed. |
| 3 | Matched on Unit Number. Unit Type changed. Extra information on address line ignored. Extra information on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 4 | Unit Number changed or ignored. |

| Code | In 3rd hex position means: |
|------|---|
| 5 | Unit Number changed or ignored. Extra information on address line ignored. Extra information on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 8 | Matched on Unit Number and Unit Type if included. House Number changed or ignored. |
| 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 moved to <code>AddressLine2</code> . |
| 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 moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |

Match Codes for No Match - Definitions for "Ennn" return codes

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

| Code | "nnn" values | Description |
|------|--------------|--|
| Ennn | | 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 = 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 Group 1 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" values | 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. |

Location Codes

The Location Codes indicate the methodology used to compute the geocode and may also provide some information about the quality of the geocode.

A Location Code of ""E" indicates a location code is not available. This usually occurs when you have requested ZIP Code centroids of a high quality, and one is not available for that match. It can occur infrequently when Spectrum Enterprise Tax does not have a 5-digit centroid location. An "E" location code type may also be returned when the input address cannot be standardized and there is no input ZIP Code. In this case, do not assume the ZIP Code returned with the nonstandardized address is the correct ZIP Code because Spectrum Enterprise Tax did not standardize the address; therefore, Spectrum Enterprise Tax does not return geocoding or Census Block information.

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.

| | |
|---|---|
| 1 st character | Always an "A" indicating an address location. |
| 2 nd 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 |
| 3 rd and 4 th character | Digit indicating other qualities about the location. |

Location Codes

| Code | Description |
|---------|---|
| AGn | Indicates a geocode match to a GeoTAX Auxiliary or Landmark Auxiliary file where n is one of the following values: |
| n = 0 | The geocode represents the center of a parcel, building or landmark. |
| 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 = 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 | <p>Structure point</p> <p>Indicates a location within a building footprint polygon that is associated with the matched address.</p> <p>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.</p> <p>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.</p> <p>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.</p> |
| nn = 07 | <p>Manually placed</p> <p>Address points are manually placed to coincide with the midpoint of a parcel's street frontage at a distance from the center line.</p> |
| nn = 08 | <p>Front door point</p> <p>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.</p> |
| nn = 09 | <p>Driveway offset point</p> <p>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.</p> |

| Code | Description |
|---------|--|
| nn = 10 | <p>Street access point</p> <p>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.</p> |
| nn = 21 | <p>Base parcel point</p> <p>The Centrus point data includes individual parcels that may be "stacked". These stacked parcels are individually identified by their unit or suite number, and Spectrum Enterprise Tax is able to match to this unit number and return the correct tax jurisdictions. If an input address is for a building or complex without a unit number, the "base" parcel information returns and will not standardize to a unit number or return additional information such as tax jurisdictions.</p> |
| nn = 22 | <p>Backfill address point</p> <p>The precise parcel centroid is unknown. The address location assigned is based on two known parcel centroids.</p> |
| nn = 23 | <p>Virtual address point</p> <p>The precise parcel centroid is unknown. The address location assigned is relative to a known parcel centroid and a street segment end point.</p> |
| nn = 24 | <p>Interpolated address point</p> <p>The precise parcel centroid is unknown. The address location assigned is based on street segment end points.</p> |
| AIn | <p>The correct segment is inferred from the candidate records at match time.</p> |
| ASn | <p>House range address geocode. This is the most accurate street interpolated geocode available.</p> |

| Code | Description |
|--|--|
| AIn, ASn and ACn share the same values for the 3 rd character "n" as follows: | |
| n = 0 | Best location. |
| 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 | <p>Indicates one or both of the following:</p> <ul style="list-style-type: none"> • 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. <p>Note: Only the second case is valid for non-TIGER data because segment range interpolation is only completed for TIGER data.</p> |
| 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. |
| 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. |

| Code | Description |
|-------|--|
| 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. |
| 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

Street centroid location codes indicate the Census ID accuracy and the position of the geocode on the returned street segment. A street centroid location code has the following characters.

| | |
|---------------------------|---|
| 1 st character | Always "C" indicating a location derived from a street segment. |
| 2 nd 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 table below contains the values and descriptions for the 2nd - 3rd characters in the street centroid location codes.

| Character position | Code | Description |
|---------------------------|------|--|
| 2 nd 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. |
| 3 rd Character | | |
| | C | Segment Centroid. |
| | L | Segment low-range end point. |
| | H | Segment high-range end point. |

ZIP + 4 Location Codes

Location codes that begin with a "Z" are ZIP + 4 centroid location codes. 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.

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

The table below contains the values and descriptions for the 2nd-4th characters in the ZIP + 4 location codes.

| Character Position | Code | Description |
|---------------------------|------|---|
| 2 nd Character | | |
| | B | Block Group accuracy (most accurate). |
| | T | Census Tract accuracy. |
| | C | Unclassified Census accuracy. Normally accurate to at least the County level. |
| 3 rd Character | | |

| Character Position | Code | Description |
|---------------------------|------|--|
| | 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 4 th character for a precise indication of locational accuracy. |
| | 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 4 th character for a precise indication of locational accuracy. |
| 4 th 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | 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 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. |
| | 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | M | Created ZIP+2 cluster centroid as defined by methodology 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 methodology 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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 geographic centroid fallback option is enabled and an address-level geocode could not be determined. Geographic centroid location codes indicate the quality of a city, county, or state centroid.

| | | | | | | | |
|---------------------------|---|----------|------------------------------------|----------|--------|----------|-------|
| 1 st character | Always "G" indicating a location derived from a geographic centroid. | | | | | | |
| 2 nd character | Geographic area type. One of the following: <table border="0"> <tr> <td>M</td> <td>Municipality (for example, a city)</td> </tr> <tr> <td>C</td> <td>County</td> </tr> <tr> <td>S</td> <td>State</td> </tr> </table> | M | Municipality (for example, a city) | C | County | S | State |
| M | Municipality (for example, a city) | | | | | | |
| C | County | | | | | | |
| S | State | | | | | | |

Type Codes

The returned type code is referenced from an installed tax district file and indicates the type of tax district or tax jurisdiction for the address location.

This appendix provides the definitions for the following tax district files' type codes:

- **Special Purpose Districts (SPD)**
- **Insurance Premium Districts (IPD)**
- **Payroll Tax Districts (PAY)**

Special Purpose Districts (SPD)

| Type | Descriptions |
|------|-----------------------------------|
| AMB | AMBULANCE DISTRICT |
| ASC | SALES AND USE TAX |
| ATA | ADVANCED TRANSPORTATION AUTHORITY |
| ATD | AIRPORT TAX DISTRICT |
| BSD | BASEBALL STADIUM DISTRICT |
| CAD | COUNTY ASSISTANCE DISTRICT |
| CCD | CRIME CONTROL DISTRICT |
| CFA | COUNTY FINANCE AUTHORITY |
| CMB | COMBINED DISTRICT |
| CTY | CITY TRANSACTIONS |
| DVD | DEVELOPMENT DISTRICT |
| EDD | ECONOMIC DEVELOPMENT DISTRICT |
| EDZ | ECONOMIC DEVELOPMENT ZONE |
| ESD | EMERGENCY SERVICES DISTRICT |
| FCD | FIRE CONTROL DISTRICT |
| FPA | FLOOD PROTECTION AUTHORITY |
| FPD | FIRE PROTECTION DISTRICT |
| FSD | FOOTBALL STADIUM DISTRICT |

| Type | Descriptions |
|------|-----------------------------------|
| HBZ | HOSPITAL BENEFIT ZONE |
| HSA | HOUSING AUTHORITY |
| HSD | HEALTHCARE SERVICES DISTRICT |
| HSP | HOSPITAL DISTRICT |
| IMP | IMPROVEMENT DISTRICT |
| IRD | INDIAN RESERVATION |
| LFW | LFW/CDC |
| LIB | LIBRARY DISTRICT |
| MSD | MUSEUM DISTRICT |
| MTA | METRO TRANSPORTATION AUTHORITY |
| OSA | OPEN SPACE AUTHORITY |
| PFD | PUBLIC FACILITY DISTRICT |
| POL | POLICE DISTRICT |
| PRD | PARK AND RECREATION DISTRICT |
| PSI | PUBLIC SAFETY IMPROVEMENT |
| RCT | RACE TRACK |
| RDA | REVENUE DEVELOPMENT AREA |
| RMA | ROAD MAINTENANCE AUTHORITY |
| RTA | REGIONAL TRANSPORTATION AUTHORITY |
| RTD | RESTAURANT TAX DISTRICT |

| Type | Descriptions |
|------|--|
| SAD | SPORTS DISTRICT |
| SCD | SCIENCE AND CULTURAL DISTRICT |
| SUT | SALES AND USE TAX |
| TDD | TRANSPORTATION DEVELOPMENT DISTRICT |
| TED | TOURISM COMMUNITY ENHANCEMENT DISTRICT |
| UNI | SCHOOL DISTRICT |
| URA | URBAN RENEWAL AUTHORITY |
| WCD | WATER COMMISSION DISTRICT |
| ZOO | ZOO DISTRICT |

Insurance Premium Districts (IPD)

| State | Type | Descriptions |
|-------|--------|-------------------------|
| AL | FIRE | Fire District |
| AL | NT-MUN | Non-Taxing Municipality |
| AL | PREM | Premium Tax District |
| AZ | PRIV | Private Fire District |
| AZ | PUB | Public Fire District |
| DE | FIRE | Fire District |
| FL | FIRE | Fire District |
| FL | POLICE | Police District |

| State | Type | Descriptions |
|-------|--------|--------------------------------------|
| GA | PREM | Premium Tax District |
| IL | FIRE | Fire District |
| KY | COUNTY | County |
| KY | MUNI | Municipality |
| KY | USD | Urban Services District |
| LA | PREM | Premium Tax District |
| MN | FIRE | Fire District |
| ND | FIRE | Fire District |
| NJ | FIRE | Fire District |
| NY | FIRE | Fire District |
| SC | FIRE | Fire District |
| SC | NT-MUN | Non-Taxing Municipality |
| SC | PREM | Premium Tax District |
| TX | PROP | Windstorm Surcharge on Property Line |

Payroll Tax Districts (PAY)

| Type | Descriptions |
|------|-------------------------------------|
| JED | Joint Economic Development District |
| MTA | Mass Transit Authority |
| MUN | Municipality |

| Type | Descriptions |
|------|-----------------|
| UNI | School District |

Class Codes

This appendix lists definitions for the FIPS Class Codes.

Class C—Incorporated Places

| Class Code | Description |
|------------|--|
| C1 | <p>Identifies an active incorporated place that is not also recognized as an Alaska Native Village Statistical area, and does not also serve as a primary county division; that is, it is included in and is part of a primary county division.</p> <p>For example, the city of Hammond, Indiana is within and part of North township; the city of Austin, Texas is within and part of several census county divisions in several counties; Hammond and Austin are coded C1.</p> |
| C2 | <p>Identifies an incorporated place that also serves as a primary county division because, although the place is coextensive with a minor civil division (MCD), the Census Bureau, in agreement with State officials, does not recognize the MCD for presenting census data because the MCD is a nonfunctioning entity; applies to Iowa and Ohio only.</p> <p>For example, the city of Dubuque, Iowa is coextensive with Julien township, which does not function as a governmental unit and may not be well-known even to local residents; the city is assigned code C2, and the township, Z8. This subclass is new for FIPS 55-3. Also see subclass C5.</p> |
| C3 | <p>Identifies a consolidated city; that is, an incorporated place that has consolidated its governmental functions with a county or MCD, but continues to include other incorporated places that are legally part of the consolidated government.</p> <p>For example, the city of Columbus, Georgia is consolidated with Muscogee County, which continues to exist as a nonfunctioning legal entity in the State; however, the town of Bibb City continues to exist as a separate active incorporated place within the consolidated government and, therefore, Columbus is treated as a consolidated city. At the time of publication, there are seven consolidated cities in the United States: Athens-Clarke County, Georgia; Butte-Silver Bow, Montana; Columbus, Georgia; Indianapolis, Indiana; Jacksonville, Florida; Milford, Connecticut; and Nashville-Davidson, Tennessee. This subclass is new for FIPS 55-3.</p> |

| Class Code | Description |
|------------|---|
| C4 | <p>Identifies an alternate authoritative common name of any member of the other subclasses of Class C. The entity code of the legal name is referenced in the "Other Name Code" of the record, and in the entry for the legal name, the Other Name Code references the alternate.</p> <p>For example, the entity in California whose legal name is San Buenaventura (subclass C1) is commonly known as Ventura, which is coded C4.</p> |
| C5 | <p>Identifies an incorporated place that also serves as a primary county division; that is, it is not included in any adjacent primary county division of class T or Z. For example, Boston, MA, is legally a primary division of the county and recognized as an incorporated place and, therefore, is coded C5. Also see subclass C2.</p> |
| C6 | <p>Identifies an incorporated place that is coincident with or approximates an Alaska Native Village statistical area. The Other Name Code references the Alaska Native Village statistical area; see code E6.</p> |
| C7 | <p>Identifies an independent city. At the time of publication, independent cities exist in only four States: Maryland (Baltimore City), Nevada (Carson City), Missouri (St. Louis City), and Virginia (41 cities). These cities also serve as county equivalents, and all but Carson City also serve as primary county divisions.</p> |
| C8 | <p>Identifies the portion of a consolidated city that is not within another incorporated place; see subclass C3. The Census Bureau identifies these nonfunctioning entities by taking the name of the consolidated city and appending in parentheses the word remainder. For example, Columbus (remainder) identifies the portion of the Columbus, Georgia consolidated city that is not also in Bibb City. This code is new for FIPS 55-3.</p> |
| C9 | <p>Identifies an inactive or nonfunctioning incorporated place.</p> |

Class U—Unincorporated Places (Except Those Associated with Facilities)

| Type | Descriptions |
|------|--|
| U1 | <p>Identifies a census designated place (CDP) with a name identical to the authoritative common name that describes essentially the same population. Also see code M2.</p> |
| U2 | <p>Identifies a CDP with a name not identical to an authoritative common name of essentially the same area. If there is an alternate authoritative common name, it is referenced in the Other Name Code field. For example, Suitland-Silver Hill, Maryland is the name of a locally delineated CDP recognized by the Census Bureau which is a combination of two communities Suitland and Silver Hill and, therefore, because it is not the authoritative name of the area, is coded U2; Sierra Vista Southeast, Arizona is a CDP that includes the built-up area adjoining the city of Sierra Vista on the southeast, but is not an authoritative name for that area and, therefore, is coded U2. Also see code M2.</p> |

| Type | Descriptions |
|------|---|
| U3 | Identifies (a) an alternate, authoritative common name of a population essentially described by a specific CDP with a different name (the Other Name Code references the CDP), or (b) a community wholly or substantially within the boundaries of a CDP with a different name (the Part of Code references the CDP). For example, Silver Hill and Suitland are coded U3 and cross-referenced to the CDP of Suitland-Silver Hill (see code U2). |
| U4 | Identifies a populated place wholly or substantially within the boundaries of an incorporated place with a different name; the Part of Code identifies the incorporated place. For example, Harlem and Greenwich Village, which are part of New York city, and Hollywood, which is part of Los Angeles, California, are coded U4. |
| U5 | Dropped. Only one place the CDP of Arlington, Virginia was in this subclass in FIPS PUB 95-2; it has been recoded as U1 as a place and as Z3 as a subclass in FIPS 55-3 as a county subdivision. |
| U6 | Identifies a populated place located wholly or substantially outside the boundaries of any incorporated place or CDP with an authoritative common name recognized by the U.S. Geological Survey. |
| U8 | Identifies a populated place located wholly or substantially outside the boundaries of an incorporated place or CDP but whose name has not been verified as authoritative by the U.S. Geological Survey. |
| U9 | Identifies a CDP that is coincident with or approximates the area of an Alaska Native Village statistical area. The Other Name Code references the Alaska Native Village statistical area; see code E2. This code is new for FIPS 55-3. |

Spectrum GeoConfidence

GeoConfidenceSurface

GeoConfidenceSurface returns geoconfidence polygons (also called surfaces) based on the quality of the geocode information generated by Spectrum Enterprise Geocoding. With the geoconfidence polygons generated, you can then overlap this polygon with other spatial data to determine a risk or probability.

This service is used by the Spectrum GeoConfidence's FloodZoneAnalysis dataflow template.

Note: GeoConfidence uses services provided by the Spectrum Enterprise Geocoding and Spatial modules.

Resource URL

JSON endpoint:

```
http://server:port/rest/GeoConfidenceSurface/results.json
```

XML endpoint:

```
http://server:port/rest/GeoConfidenceSurface/results.xml
```

Request

The input fields for GeoConfidenceSurface are the output fields returned by the GeoConfidence output category of the Enterprise Geocoding Module. These fields are described below.

| columnName Field Name Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| GeoConfidenceCode | 13 | <p>The value returned in this field indicates which geoconfidence surface type has been returned.</p> <p>The possible values are:</p> <p>INTERSECTION A geocode point for the intersection of two streets.</p> <p>ADDRESS An array of street segment points representing the street segment where the address is located.</p> <p>POINT If the geocoder was able to match the address using point data, the point geometry where the address is located.</p> <p>POSTAL1 A geocode point for the ZIP centroid.</p> <p>POSTAL2 An array of points for all street segments in the ZIP + 2 in which the address is located.</p> <p>POSTAL3 An array of points for street segments in the ZIP + 4 in which the address is located.</p> <p>ERROR An error has occurred.</p> |
| StreetSegmentPoints | 1024 | <p>An array of latitude/longitude values that represent the street segment points.</p> <p>Note: This field contains values only if the GeoConfidenceCode field returns a value of ADDRESS, POSTAL2, or POSTAL3.</p> |

| columnName Field Name Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| GeoConfidenceCentroidLatitude | 11 | The latitude of the centroid of the geoconfidence polygon. |
| GeoConfidenceCentroidLongitude | 12 | The longitude of the centroid of the geoconfidence polygon. |

Response

The GeoConfidenceSurface output field contains the geoconfidence polygon.

| Response Element | Description |
|------------------|--|
| Geometry | A geoconfidence polygon that represents the returned geometry. |

Spectrum Global Address Validation

Spectrum Global Address Validation

Spectrum Global Address Validation provides enhanced address standardization and validation. Global Address Validation combines data from multiple data sources into one database to provide the most extensive and accurate international addressing data possible.

Spectrum Global Address Validation analyzes and compares each input address to the Global Addressing database for the appropriate country. If needed, Global Address Validation corrects and formats the address in accordance with the postal standards for that country.

Standard address output consists of address lines which correspond to how the address would appear on an address label. City, state or province, postal code, and other data are also included in the standard address output.

Global AddressValidation is part of the Global Addressing Module.

Resource URL

JSON endpoint:

```
http://server:port/rest/GlobalAddressValidation/result.json
```

XML endpoint:

```
http://server:port/rest/GlobalAddressValidation/result.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://server:8080/rest/GlobalAddressValidation/result.json?
Data.AddressLine1=103-113 STANLEY ST VICTORIA WEST MELBOURNE 3003
&Data.Country=AUS
```

The JSON returned by this request would be:

```
{ "output_port": [ {
  "AddressLine1": "103-113 Stanley Street",
  "AddressBlock1": "103-113 Stanley Street",
  "AddressBlock2": "West Melbourne VIC 3003",
  "City": "West Melbourne",
  "StateProvince": "VIC",
  "PostalCode": "3003",
  "Country": "Australia",
  "PrecisionCode": "S8HPNTSCZG",
  "ProcessedBy": "GAM",
  "MultimatchCount": "1",
  "HouseNumber": "103-113",
  "StreetName": "Stanley",
  "StreetType": "Street",
  "Confidence": "92",
  "Principality": "VIC",
  "MatchOnAllStreetFields": "true",
  "MatchOnStreetDirectional": "true",
  "City.Matched": "true",
  "CitySubdivision.Matched": "true",
  "StateProvince.Matched": "false",
  "StateProvinceSubdivision.Matched": "true",
  "StreetName.Matched": "true",
  "StreetType.Matched": "true",
  "Firmname.Matched": "true",
  "Housenumber.Matched": "true",
  "Postalcode.Matched": "true",
  "user_fields": []
} ] }
```

Example with XML Response

The following example requests an XML response:

```
http://server:8080/rest/GlobalAddressValidation/result.xml?Data.AddressLine1=103-113
STANLEY ST VICTORIA WEST MELBOURNE 3003&Data.Country=AUS
```

The XML returned by this request would be:

```
<xml.GlobalAddressValidationResponse
xmlns="http://www.precisely.com/spectrum/services/GlobalAddressValidation">
<output_port>
  <Row>
    <AddressLine1>103-113 Stanley Street</AddressLine1>
    <AddressBlock1>103-113 Stanley Street</AddressBlock1>
    <AddressBlock2>West Melbourne VIC 3003</AddressBlock2>
    <City>West Melbourne</City>
    <StateProvince>VIC</StateProvince>
    <PostalCode>3003</PostalCode>
    <Country>Australia</Country>
    <PrecisionCode>S8HPNTSCZG</PrecisionCode>
    <ProcessedBy>GAM</ProcessedBy>
    <MultimatchCount>1</MultimatchCount>
    <HouseNumber>103-113</HouseNumber>
    <StreetName>Stanley</StreetName>
    <StreetType>Street</StreetType>
    <Confidence>92</Confidence>
    <Principality>VIC</Principality>
    <MatchOnAllStreetFields>true</MatchOnAllStreetFields>
    <MatchOnStreetDirectional>true</MatchOnStreetDirectional>
    <City.Matched>true</City.Matched>
    <CitySubdivision.Matched>true</CitySubdivision.Matched>
    <StateProvince.Matched>false</StateProvince.Matched>
    <StateProvinceSubdivision.Matched>true
      </StateProvinceSubdivision.Matched>
    <StreetName.Matched>true</StreetName.Matched>
    <StreetType.Matched>true</StreetType.Matched>
    <Firmname.Matched>true</Firmname.Matched>
    <Housenumber.Matched>true</Housenumber.Matched>
    <Postalcode.Matched>true</Postalcode.Matched>
    <user_fields/>
  </Row>
</output_port>
</xml.GlobalAddressValidationResponse>
```

JSON Example

```
http://server:8080/rest/GlobalAddressValidation/result.json
```

JSON POST request example:

```
{
  "options":
  {
    "Database_GAV": "JP_AU"
  },
  "input_port" : {
    "Input" : [
      {
        "AddressLine1": "103-113 STANLEY ST VICTORIA WEST MELBOURNE 3003",
        "Country": "AUS"
      },
      {
        "AddressLine1": "103-114 STANLEY ST VICTORIA WEST MELBOURNE 3004",
        "Country": "AUS"
      }
    ]
  }
}
```

JSON POST response example:

```
{
  "output_port": [
    {
      "AddressLine1": "103-113 Stanley Street",
      "AddressBlock1": "103-113 Stanley Street",
      "AddressBlock2": "West Melbourne VIC 3003",
      "City": "West Melbourne",
      "StateProvince": "VIC",
      "PostalCode": "3003",
      "Country": "Australia",
      "PrecisionCode": "S8HPNTSCZG",
      "ProcessedBy": "GAM",
      "MultimatchCount": "1",
      "HouseNumber": "103-113",
      "StreetName": "Stanley",
      "StreetType": "Street",
      "Confidence": "92",
      "Principality": "VIC",
      "MatchOnAllStreetFields": "true",
      "MatchOnStreetDirectional": "true",
      "City.Matched": "true",
      "CitySubdivision.Matched": "true",
      "StateProvince.Matched": "false",
      "StateProvinceSubdivision.Matched": "true",
      "StreetName.Matched": "true",
      "StreetType.Matched": "true",
      "Firmname.Matched": "true",
    }
  ]
}
```

```

        "Housenumber.Matched": "true",
        "Postalcode.Matched": "true",
        "user_fields": []
    },
    {
        "AddressLine1": "103-114 Stanley Street",
        "AddressBlock1": "103-114 Stanley Street",
        "AddressBlock2": "West Melbourne VIC 3003",
        "City": "West Melbourne",
        "StateProvince": "VIC",
        "StateProvinceSubdivision": "Melbourne",
        "PostalCode": "3003",
        "Country": "Australia",
        "PrecisionCode": "S5HPNTSC-A",
        "ProcessedBy": "GAM",
        "MultimatchCount": "1",
        "HouseNumber": "103-114",
        "StreetName": "Stanley",
        "StreetType": "Street",
        "Confidence": "80",
        "Principality": "VIC",
        "MatchOnAllStreetFields": "true",
        "MatchOnStreetDirectional": "true",
        "City.Matched": "true",
        "CitySubdivision.Matched": "true",
        "StateProvince.Matched": "false",
        "StateProvinceSubdivision.Matched": "true",
        "StreetName.Matched": "true",
        "StreetType.Matched": "true",
        "Firmname.Matched": "true",
        "Housenumber.Matched": "true",
        "Postalcode.Matched": "false",
        "user_fields": []
    }
]
}

```

XML Example

```
http://server:8080/rest/GlobalAddressValidation/result.xml
```

XML POST request example:

```

<GlobalAddressValidationRequest
xmlns:gav="http://www.precisely.com/spectrum/services/GlobalAddressValidation">
<options>
  <Database_GAV>JP_AU</Database_GAV>
</options>
<gav:input_port>
  <gav:Input>
    <gav:AddressLine1>103-113 STANLEY ST VICTORIA WEST MELBOURNE

```

```

3003
  </gav:AddressLine1>
  <gav:Country>AUS</gav:Country>
  <gav:user_fields>
    <gav:user_field>
      <gav:name>id</gav:name>
      <gav:value>1</gav:value>
    </gav:user_field>
  </gav:user_fields>

</gav:Input>
<gav:Input>
  <gav:AddressLine1>103-113 STANLEY ST VICTORIA WEST MELBOURNE
3003
  </gav:AddressLine1>
  <gav:Country>AUS</gav:Country>
  <gav:user_fields>
    <gav:user_field>
      <gav:name>id</gav:name>
      <gav:value>2</gav:value>
    </gav:user_field>
  </gav:user_fields>
</gav:Input>
</gav:input_port>
</GlobalAddressValidationRequest>

```

XML POST response example:

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xml.GlobalAddressValidationResponse
xmlns="http://www.precisely.com/spectrum/services/GlobalAddressValidation">

  <output_port>
    <Row>
      <AddressLine1>103-113 Stanley Street</AddressLine1>
      <AddressBlock1>103-113 Stanley Street</AddressBlock1>
      <AddressBlock2>West Melbourne VIC 3003</AddressBlock2>
      <City>West Melbourne</City>
      <StateProvince>VIC</StateProvince>
      <PostalCode>3003</PostalCode>
      <Country>Australia</Country>
      <PrecisionCode>S8HPNTSCZG</PrecisionCode>
      <ProcessedBy>GAM</ProcessedBy>
      <MultimatchCount>1</MultimatchCount>
      <HouseNumber>103-113</HouseNumber>
      <StreetName>Stanley</StreetName>
      <StreetType>Street</StreetType>
      <Confidence>92</Confidence>
      <Principality>VIC</Principality>
      <MatchOnAllStreetFields>true</MatchOnAllStreetFields>
      <MatchOnStreetDirectional>true</MatchOnStreetDirectional>
      <City.Matched>true</City.Matched>
    </Row>
  </output_port>
</xml.GlobalAddressValidationResponse>

```

```

<CitySubdivision.Matched>true</CitySubdivision.Matched>
<StateProvince.Matched>false</StateProvince.Matched>
<StateProvinceSubdivision.Matched>true
</StateProvinceSubdivision.Matched>
<StreetName.Matched>true</StreetName.Matched>
<StreetType.Matched>true</StreetType.Matched>
<Firmname.Matched>true</Firmname.Matched>
<Housenumber.Matched>true</Housenumber.Matched>
<Postalcode.Matched>true</Postalcode.Matched>
<user_fields>
  <user_field>
    <name>id</name>
    <value>1</value>
  </user_field>
</user_fields>
</Row>
<Row>
  <AddressLine1>103-113 Stanley Street</AddressLine1>
  <AddressBlock1>103-113 Stanley Street</AddressBlock1>
  <AddressBlock2>West Melbourne VIC 3003</AddressBlock2>
  <City>West Melbourne</City>
  <StateProvince>VIC</StateProvince>
  <PostalCode>3003</PostalCode>
  <Country>Australia</Country>
  <PrecisionCode>S8HPNTSCZG</PrecisionCode>
  <ProcessedBy>GAM</ProcessedBy>
  <MultimatchCount>1</MultimatchCount>
  <HouseNumber>103-113</HouseNumber>
  <StreetName>Stanley</StreetName>
  <StreetType>Street</StreetType>
  <Confidence>92</Confidence>
  <Principality>VIC</Principality>
  <MatchOnAllStreetFields>true</MatchOnAllStreetFields>
  <MatchOnStreetDirectional>true</MatchOnStreetDirectional>
  <City.Matched>true</City.Matched>
  <CitySubdivision.Matched>true</CitySubdivision.Matched>
  <StateProvince.Matched>false</StateProvince.Matched>
  <StateProvinceSubdivision.Matched>true
  </StateProvinceSubdivision.Matched>
  <StreetName.Matched>true</StreetName.Matched>
  <StreetType.Matched>true</StreetType.Matched>
  <Firmname.Matched>true</Firmname.Matched>
  <Housenumber.Matched>true</Housenumber.Matched>
  <Postalcode.Matched>true</Postalcode.Matched>
  <user_fields>
    <user_field>
      <name>id</name>
      <value>2</value>
    </user_field>
  </user_fields>
</Row>
</output_port>
</xml.GlobalAddressValidationResponse>

```

Request

Input

Spectrum Global Address Validation uses an address as input. All addresses use this format regardless of the address's country. To obtain the best performance and address match, your input address lists should be as complete as possible, free of misspellings and incomplete addresses, and as close to postal authority standards as possible. Most postal authorities have websites that contain information about address standards for their particular country.

Note: The country name or two- or three- character country ISO code is optional. If you omit the country, Spectrum Global Address Validation returns the best available candidates for the **Default Country** selected on the **Default Options** tab. For a list of ISO codes, see [ISO Country Codes and Coder Support](#).

Table 3: Spectrum Global Address Validation Input

| Field Name | Format | Description |
|--------------|--------|---|
| FirmName | String | Company, firm name, or place name. For example, PRECISELY. |
| AddressLine1 | String | The first address line. For example, 34 GLENVIEW ROAD MOUNT KURNING-GAI NSW 2080 . AddressLine1 can also contain a dual address (contains more than one mailable address). For example, the dual address PO BOX 3220 STN C 181 QUEEN STREET OTTAWA ON K1Y1E4 contains both a PO Box and a street address. |
| AddressLine2 | String | The second address line (USA only). |
| AddressLine3 | String | The third address line (USA only). |
| AddressLine4 | String | The fourth address line (USA only). |
| AddressLine5 | String | The fifth address line (USA only). |
| AddressLine6 | String | The sixth address line (USA only). |
| LastLine | String | The last line of the address. For example, 34 GLENVIEW ROAD MOUNT KURNING-GAI NSW 2080 . Note: Spectrum Global Address Validation only considers the LastLine information when individual components such as City and PostalCode are not provided. |

| Field Name | Format | Description |
|--------------------------|--------|--|
| City | String | The city or town name. To produce the best match results, your input address should use the official city name. |
| CitySubdivision | String | The name of one of the following depending on the country: <ul style="list-style-type: none"> • Not used—AUS, AUT, BEL, CHE, DEU, DNK, FIN, FRA, IRL, MYS, NLD, NOR, POL, SWE • Dissemination Area and Enumeration Area (DA and EA)—CAN • Locality—BRA, GBR, GRC, ITA, ESP • Suburb—NZL • Urbanization name (Puerto Rico)—USA |
| StateProvince | String | The name of the state or province depending on the country: <ul style="list-style-type: none"> • Not used—BEL, CHE, DNK, IRL, NLD, NOR • Bundesland—DEU • Province—CAN • Province (voivodship)—POL • Region—AUT, ESP, FRA, GBR, GRC, NZL • Region (län)—FIN • Region (lan)—SWE • State—AUS, BRA, USA • State (negeri)—MYS |
| StateProvinceSubdivision | String | The name of the state or province subdivision depending on the country: <ul style="list-style-type: none"> • Not used—AUT, BRA, CAN, FIN, GBR, MYS • Department—FRA • District—GRC • District (fylke/counties)—NOR • District (powiat)—POL • Kommun—SWE • Kreis—DEU • Local Government Authority (LGA)—AUS • Province—BEL, CHE, DNK, ESP, IRL, ITA, NLD • Region—NZL |
| PostalCode | String | The postal code in the appropriate format for the country. |
| Country | String | The country name of the two- or three-character ISO country code. This field is optional. If you omit the country, Spectrum Global Address Validation returns the best available candidates for the Default Country selected on the Default Options tab. For a list of ISO codes, see ISO Country Codes and Coder Support . |

Options

Global Address Validation uses the default options settings to define address validation processing.

Table 4: Global Addressing Options

| Option Name | Country Support | Description |
|---------------------------|-----------------|---|
| Global Addressing Options | All except USA | The options specific to global address processing. |
| City fallback | All | When a street level match cannot be made, use the input city to determine match candidates. |
| Custom match fields | All | <p>These options set custom match criteria for determining match candidates. To enable these options, you must set the Match Mode to <i>Custom</i>. By default, these options are disabled.</p> <p>Address number A match must be made to the input address number.</p> <p>Street A match must be made to the input street name, type, and directional fields.</p> <p>City A match must be made to the input address city.</p> <p>City subdivision A match must be made to the input address city subdivision.</p> <p>State/province A match must be made to the input address state or province.</p> <p>State/province subdivision A match must be made to the input address state or province subdivision.</p> <p>Postal code A match must be made to the input address postal code.</p> |
| Database | All | The database to use for address processing. Only databases that have been defined in the Database Resources panel in the Management Console are available. |

| Option Name | Country Support | Description |
|---------------------------|-------------------|--|
| Default country | All | <p>The default country for address processing.</p> <p>To improve coding performance when your input addresses do not contain country information, set up an additional instance of the Global Address Validation stage as a preliminary stage to process and retrieve the country code for the input addresses.</p> <ol style="list-style-type: none"> Set up an additional instance of the Global Address Validation stage as a preliminary (first) stage in your dataflow. <ul style="list-style-type: none"> Give the preliminary stage a unique label. For example, "Identify Country". Specify "World" as the default country for the preliminary stage. The preliminary stage uses the available input address elements with additional data sources (available when you select "World" as the default country) to determine the country code. The "country-coded" output from the preliminary stage becomes the input for the next step in the dataflow. As the next step in the dataflow, the addresses are sent through a second Global Address Validation stage with the proper country code (retrieved in the preliminary stage) to validate the address to the street/house/premise level. |
| Match mode | All | <p>Match modes determine the leniency used to make a match between the input address and the reference data. Select one of the following match modes based on the quality of your input and your desired output.</p> <p>Exact A very tight match. This restrictive mode generates the fewest match candidates. When using this mode, ensure that your input is very clean; free of misspellings and incomplete addresses.</p> <p>Relaxed A loose match. This mode generates the most match candidates and results in more multiple matches. Use this mode if you are not confident that your input is clean and free of misspellings and incomplete addresses.</p> <p>Custom A custom match. Allows you to define the matching criteria by selecting Custom Match Fields.</p> |
| Postal fallback | All | <p>When a street level match cannot be made, use the input postal code to determine match candidates.</p> |
| Prefer PO Box over street | CAN FRA GBR | <p>Prefer candidates matching the input PO Box over matches to the input street. The default is disabled.</p> |

| Option Name | Country Support | Description |
|------------------------------|-----------------|---|
| Prefer postal code over city | AUS | Prefer candidates matching the input postal code over matches to the input city. The default is disabled. |

Table 5: US Addressing Options

| Option Name | Description |
|--------------------------------------|--|
| US Addressing Options | The options specific to U.S. address processing. |
| Address | Mailer's address. This information displays in section D box 3 on the USPS Form 3553 (CASS Summary Report). |
| Address2 | Additional address line for Mailer's address displays in section D box 3 on the USPS Form 3553 (CASS Summary Report). |
| Address3 | Additional address line for Mailer's address displays in section D box 3 on the USPS Form 3553 (CASS Summary Report). |
| Address4 | Additional address line for Mailer's address displays in section D box 3 on the USPS Form 3553 (CASS Summary Report). |
| All street matching | Perform All Street Matching (ASM) processing. ASM applies additional matching logic to correct errors in street names to find a match. For example, when the first letter of a street is misspelled or missing on input, ASM searches all street names in a locality to find an input address. ASM provides the best address validation but may reduce performance. ASM processing is available for U.S. addresses only. |
| Assign abbreviated city | Return abbreviated city names in the label lines. |
| Assign enhanced Line of Travel codes | Assign enhanced Line of Travel (eLOT) codes. The default is disabled. |
| CASS flag | Process in a United States Postal Service (USPS) CASS-certified mode. Note: If you are performing Global Address Validation US Addressing processing outside of the physical United States, you must disable the Delivery Point Validation (DPV), LACSLink, SuiteLink, and Residential Delivery Indicator (RDI) options. You will not be able to run in a USPS CASS-certified mode. |

| Option Name | Description |
|---|---|
| CASS Mailer Information | The Mailer's name and address that displays on the USPS Form 3553 (CASS Summary Report). This information is required if you are running DPV and optional if you are not running DPV. |
| City line | The Mailer's city, state, and ZIP Code information displays in section D box 3 on the USPS Form 3553 (CASS Summary Report). |
| Commercial Mail Receiving Agency (CMRA) | Perform Commercial Mail Receiving Agency (CMRA) processing. A private company offering mailbox rental services to individuals and businesses is a Commercial Mail Receiving Agency (CMRA). The default is disabled. |
| Convert secondary to PMB | <p>Convert secondary information to Private Mail Box (PMB) under the following conditions:</p> <ul style="list-style-type: none"> • A secondary number is present in the returned ZIP + 4 address. • The secondary number does not DPV confirm. • The primary number (and/or other secondary number) confirms as a Commercial Mail Receiving Agency (CMRA). • The unconfirmed unit designator is not a pound sign (#). <p>Note: This processing only applies if the primary address codes to a CMRA.</p> |
| Delivery Point Validation | <p>Perform Delivery Point Validation (DPV) processing. USPS CASS regulations require DPV processing. If you do not perform DPV processing, Global Address Validation does not generate a USPS Form 3553 (CASS Summary Report). The default is disabled.</p> <p>Note: When DPV is disabled, all DPV options are also disabled and display in a grayed out mode.</p> <p>Note: If you are performing Global Address Validation US Addressing processing outside of the physical United States, you must disable the Delivery Point Validation (DPV) option. You will not be able to run in a USPS CASS-certified mode.</p> |
| Do not concatenate dangling lines | Check Do not concatenate dangling lines if you do not want to concatenate one-word lines to address lines. |
| Do not concatenate lines beginning with # | Check Do not concatenate lines beginning with # if you do not want to concatenate lines beginning with a # to existing address lines. |
| Do not identify CitySubdivision | Check Do not identify CitySubdivision if you do not want to perform Urbanization name (Puerto Rico) processing. |

| Option Name | Description |
|--|--|
| Do not identify FirmName | Check Do not identify FirmName if you do not want to perform firm name processing. |
| Do not identify LastLine | Check Do not identify LastLine if you do not want to perform last line (City, State, or ZIP Code) processing. |
| Do not merge secondary or PMB | Check Do not merge secondary or PMB if you do not want to merge a separated second unit and PMB information. |
| Do not recognize periods (.) as valid characters | Check Do not recognize periods (.) as valid characters if you do not want to recognize periods as valid characters (periods should be removed before scanning address lines). |
| Door Not Accessible | Perform DPV Door Not Accessible (DNA) processing. Use the DNA Table to identify delivery addresses where carriers cannot knock on the door for mail delivery or where carriers cannot physically access a residence/building such as rural/highway contract route (HCR), long driveway, or gated community. Return the proper DNA code to the output. The default is disabled. |

| Option Name | Description |
|----------------------|--|
| Dual address | <p>If the input file contains dual addresses (one a conventional address, a second containing a PO Box address), this field determines the order to use to process and match the addresses. If the selected address is valid, processing stops. If the selected address does not validate, processing attempts to code the secondary address.</p> <ul style="list-style-type: none"> • Above city and ZIP Code. The address line closest to the last line in the address is given the highest priority in the match process. Any address line above the last line is not used for matching. Default. • Line 1 is given preference. The first line in the dual address is given the highest priority in the match process. • Line 2 is given preference. The second line in the dual address is given the highest priority in the match process. • Conventional is given preference. The conventional address is given the highest priority in the match process. • PO Box is given preference. The PO Box is given the highest priority in the match process. • The first valid address is given preference. The first valid address (in the order Address line 1 and then Address line 2) is given the highest priority in the match process. <p>Note: When dual addresses are contained on a single line and the CASS flag is enabled, the USPS address type priority is used in the following order:</p> <ol style="list-style-type: none"> 1. PO Box 2. Firm 3. Highrise 4. Street 5. Rural Route 6. General Delivery |
| Early Warning System | <p>Perform Early Warning System (EWS) processing. Perform Early Warning System (EWS) processing. New address information that is in use, but not yet available on the ZIP + 4 File, can be found as part of the USPS Early Warning System (EWS). The USPS requires all CASS-certified software to verify addresses that are not found in the current ZIP + 4 File against the USPS EWS File. If an address is found in the EWS File, the address is not matched to any similar addresses in the current ZIP + 4 File. Instead, the input address fails and is not coded until the ZIP + 4 File is updated with the correct address from the USPS EWS File. The default is disabled.</p> |

| Option Name | Description |
|----------------------------------|---|
| LACSLink | <p>Perform LACSLink (Locatable Address Conversion System) processing. USPS CASS regulations require LACSLink processing. If you do not perform LACSLink processing, Global Address Validation does not generate a USPS Form 3553 (CASS Summary Report). The default is disabled.</p> <p>Note: If you are performing Global Address Validation US Addressing processing outside of the physical United States, you must disable the LACSLink option. You will not be able to run in a USPS CASS-certified mode.</p> |
| Log level | <p>The level of the messages to log.</p> <ul style="list-style-type: none"> • No messages. • Critical messages • Error messages • Warning messages • Info messages • Debug messages |
| Multiple Address Line Processing | <p>The options specific to multiple address line processing. To specify the format of your returned lines address, select three options (one for each returned line and a special option).</p> |
| Name | <p>The Mailer's name displays in section D box 3 on the USPS Form 3553 (CASS Summary Report).</p> |
| No-Stat | <p>Perform DPV No-Stat processing. Use the No-Stat table to identify deliveries that are not valid for Computerized Delivery Sequence (CDS) pre-processing. Return the proper No-Stat code to the output. The default is disabled.</p> |
| No Secure Location | <p>Perform DPV No Secure Location (NSL) processing. Use the NSL table to identify delivery locations that are not secure. For example, a carrier can access a door but cannot leave a package due to security concerns. The NSL designation alerts mailers to locations where businesses are closed on certain days and locations without mail receptacles (for example, a storefront). Return the NSL result to the output. The default is disabled.</p> |
| PO Box as street address | <p>Perform DPV P.O. Box Street Address (PBSA) processing. A PBSA address is a street address that represents a USPS P.O. Box. Use the PBSA table to identify PBSA addresses. Return the PBSA result to the output. The default is disabled.</p> |

| Option Name | Description |
|--------------------------------|--|
| Processing Mode | <p>Defines the intended processing mode.</p> <ul style="list-style-type: none"> Specify a value of I for Interactive mode. This implies random, single record input. No performance caching will be turned on. Specify the default value of B for Batch mode. An internal cache will be turned on allowing improved performance. Best performance in batch mode is achieved when the input is in PostalCode (ZIP Code) sequence. <p>This option is only available during with the Web Services API.</p> |
| R777 deliverable | <p>Addresses with Carrier Route code R777 are phantom routes and are not eligible for street delivery. However, since these addresses are assigned a ZIP + 4 code by the USPS, these addresses are marked as deliverable. If you do not want the addresses with Carrier Route code R777 marked as deliverable, disable this option and the following actions are performed for the address:</p> <ul style="list-style-type: none"> ZIP + 4 code is not assigned. The address is not counted on the USPS Form 3553 (CASS Summary Report) DPV Footnote Code R7 is returned. |
| Remove noise characters | <p>Remove noise characters (for example, unnecessary punctuation and blanks).</p> |
| Residential Delivery Indicator | <p>Perform Residential Delivery Indicator (RDI) processing. The default is disabled.</p> <p>Note: If you are performing Global Address Validation US Addressing processing outside of the physical United States, you must disable the Residential Delivery Indicator (RDI) option. You will not be able to run in a USPS CASS-certified mode.</p> |

| Option Name | Description |
|---------------------|--|
| Return alias street | <p>Return alias street names in the label line. An alias street name is an alternate name for a street, maintained at the ranged ZIP + 4 Code level.</p> <ul style="list-style-type: none"> • Return input alias or base. If the input address matches to an alias, return the alias. If the input address matches to a base address, but a preferred alias exists, return the preferred alias. If the input address matches to a base address and no Preferred alias exists, return the base address • Return preferred alias or base. If a preferred alias exists, return the preferred alias. Otherwise, return the base street. • Return preferred, abbreviated, input alias or base. Return the preferred alias. If no preferred alias exists, return the abbreviated alias. If no abbreviated or preferred alias exists, but some other alias type was input, return the input alias. If none of these scenarios apply, return the base street name. • Return preferred, abbreviated alias, or base. If a preferred alias exists, return the preferred alias. If no preferred alias exists, return the Abbreviated alias. If neither exists, return the base street name. • Return abbreviated, preferred, input alias or base. Return the abbreviated alias. If no abbreviated alias exists, return the preferred alias. If no abbreviated or preferred alias exists, but some other alias type was input, return the input alias. If no alias exists, return the base street name. • Return abbreviated, preferred, alias or base. Return the abbreviated alias. If no abbreviated alias exists, return the Preferred alias. If neither exists, return only the base street name |
| Return input firm | Return the input firm. |
| Return Line 1 | <p>Specify the format for returned line 1.</p> <ul style="list-style-type: none"> • Return the first valid line from the top. • Return the first firm line from top. • Return the first valid line above the city line. • Return the street address line above the city line. • Return the PO box or RR/HC line above the city line. • Return the best street or PO box line from top. If not found, the first firm or rural route line from top. When selecting this option for return line 1, processing selects from the top down in this order: <ol style="list-style-type: none"> 1. First PO box line or complete address. 2. First street line with a range but no suffix. 3. First Street line with a suffix but no range. 4. First rural route line. 5. First firm type. |

| Option Name | Description |
|----------------------------|--|
| Return Line 2 | <p>Specify the format for returned line 2.</p> <ul style="list-style-type: none"> • Return a blank line. • Return the first valid line from the top. • Return the second valid line from the top. • Return the first firm line from top. • Return the second half of the combined address line if the first half is returned in line 1. • Return the first valid line above the city line. • Return the second valid line above city line. • Return the street address line above the city line. • Return the PO box or RR/HC line above the city line. • Return the first valid line above city line. If that criteria is not met resulting in a blank second line, return the topmost line not used in return line 2. |
| Return Line Order | <p>Specify the order for returning the standardized lines.</p> <ul style="list-style-type: none"> • 0Firm line. • 1PO Box address line. • 2Address line with both a range and a suffix word. • 3Address line with a range but no suffix, Address line with a suffix but no range. • 4Rural route address line. • 5Personal name, Firm name (w/o firm words), Unidentified. • 6Apartment type line. • 7Possible city line. • 8City line. • 9Ignore this line. • BBox address line following a rural route address line. • MMilitary address line. • NBest address line 1, best address line 2, city state, firm. URB and ZIP Code are returned in separate fields. The default is N. • RRRural route address line preceding a box line. |
| Return SuiteLink secondary | <p>Indicate how to return secondary information when SuiteLink secondary information is available.</p> <ul style="list-style-type: none"> • Both SuiteLink and input. Return both SuiteLink and input secondary information. Default. • SuiteLink only. Return SuiteLink secondary only. Do not return input secondary. • Input only. Return input secondary only. Do not return SuiteLink secondary. • None. Do not return SuiteLink secondary or input secondary. |

| Option Name | Description |
|-----------------------------|---|
| Save PMB in separate field | Do not merge separate second unit and PMB information. Save PMB in separate field. |
| Save unit in separate field | Do not merge separate second unit and PMB information. Save unit in separate field. |
| Special Options | <p>Specify special options for returned lines.</p> <ul style="list-style-type: none"> • Use all special options for returned lines. • Do not use any special options for returned lines. • Add a range of "0" to street lines without a range. For example, Main St becomes 0 Main St. • Separate combined address lines. |
| SuiteLink | <p>Perform SuiteLink processing. USPS CASS regulations require SuiteLink processing. If you do not perform SuiteLink processing, Global Address Validation does not generate a USPS Form 3553 (CASS Summary Report). The default is disabled.</p> <p>Note: If you are performing Global Address Validation US Addressing processing outside of the physical United States, you must disable the SuiteLink option. You will not be able to run in a USPS CASS-certified mode.</p> |
| Throwback | Perform DPV P.O. Box Throwback processing. Use the P.O. Box Throwback Table to identify a delivery point that is a street address where mail is not delivered. Instead, delivery is made to the customer's P.O. Box address. Return the P.O. Box Throwback result to the output. The default is disabled. |
| Tie Break | <p>The USPS allows DPV processing to be used as a tie breaker for matching inexact street records. If only one of the records in a tie is delivery point validated, a match is allowed to the inexact record. When processing results in an inexact match due to the input address directional, DPV processing can be used as a tie breaker if only one of the records is found to be delivery point validated and the delivery point validated record does not violate the cardinal direction rule. The default is enabled.</p> <p>USPS CASS regulations require DPV Tie Break processing to generate the USPS Form 3553 (USPS CASS Summary Report).</p> |
| Vacant | Perform DPV Vacant Table processing. Use the Vacant Table to identify delivery addresses that have been active in the past but, according to USPS data, have not been occupied within the last 90 days. Return the proper Vacant code to the output. The default is disabled. |

Table 6: Output Options

| Option Name | Description |
|---------------------------|--|
| Output Options | The elements to be returned by Global Address Validation processing. |
| Casing | The format for the returned address: <ul style="list-style-type: none"> Mixed The output data is returned in a mixed case format. For example, 100 Main Street. Lower The output data is returned in an all lower case format. For example, 100 main street. Upper The output data is returned in an all upper case format. For example, 100 MAIN STREET. The default is Upper. |
| Country specific fields | Return country-specific output information. |
| Global Addressing Options | The output options specific to global address processing. |
| Input address | Return the original input address. |
| Maximum records to return | The number of match candidates to return. Using a looser Match Mode setting such as "Relaxed" can result in the matching output including multiple match candidates. The specified number of match candidates are presented to the user to select the desired match candidate. If an exact match is found, the single match candidate is returned. |
| Parsed address | The parsed address elements (for example, Address Line 1, postal codes, and country). The meaning of some of these fields may vary by country. Do not select Parsed address when returning G/Z level matches. |
| Precision | Return a code describing the precision of the address match. |

Spectrum Global Type Ahead

Spectrum Global Type Ahead automatically suggests addresses as you type and immediately returns candidates based on your input. You can then select your candidate from the presented candidate list.

Spectrum Global Type Ahead is part of Spectrum Global Addressing Management.

Resource URL

JSON endpoint:

```
http://server:port/rest/GlobalTypeAheadValidation/result.json
```

Example with JSON Response

The following example requests a JSON response:

```
http://server:8080/rest/GlobalTypeAhead/result.json?Data.AddressLine1=Victori
```

The JSON returned by this request would be:

```
{
  "output_port": [
    {
      "FirmName": "The Victoria",
      "AddressLine1": "West Street",
      "LastLine": "DUNSTABLE, LU6 1ST",
      "FormattedAddress": "The Victoria, 69, West Street, DUNSTABLE,
        LU6 1ST",
      "PostalCode": "LU6 1ST",
      "City": "DUNSTABLE",
      "Country": "UNITED KINGDOM",
      "Type": "2",
      "Ranges": [],
      "user_fields": []
    },
    {
      "FirmName": "The Victoria",
      "AddressLine1": "Wilmslow Road",
      "LastLine": "MANCHESTER, M20 3BW",
      "FormattedAddress": "The Victoria, 438, Wilmslow Road, MANCHESTER,
        M20 3BW",
      "PostalCode": "M20 3BW",
      "City": "MANCHESTER",
      "Country": "UNITED KINGDOM",
      "Type": "2",
      "Ranges": [],
      "user_fields": []
    },
    {
      "FirmName": "The Victoria, 42-43",
      "AddressLine1": "Promenade",
      "LastLine": "SOUTHPORT, PR9 0DS",
      "FormattedAddress": "The Victoria, 42-43, Promenade, SOUTHPORT,
        PR9 0DS",
      "PostalCode": "PR9 0DS",
      "City": "SOUTHPORT",

```

```

    "Country": "UNITED KINGDOM",
    "Type": "2",
    "Ranges": [],
    "user_fields": []
  },
  {
    "FirmName": "The Victoria, 2a",
    "AddressLine1": "Hough Side Road",
    "LastLine": "PUDSEY, LS28 9BR",
    "FormattedAddress": "The Victoria, 2a, Hough Side Road, PUDSEY,
    LS28 9BR",
    "PostalCode": "LS28 9BR",
    "City": "PUDSEY",
    "Country": "UNITED KINGDOM",
    "Type": "2",
    "Ranges": [],
    "user_fields": []
  },
  {
    "FirmName": "The Victoria",
    "AddressLine1": "Ainsworth Road",
    "LastLine": "Radcliffe, MANCHESTER, M26 4FD",
    "FormattedAddress": "The Victoria, 119, Ainsworth Road, Radcliffe,
    MANCHESTER,
    M26 4FD",
    "Locality": "Radcliffe",
    "PostalCode": "M26 4FD",
    "City": "MANCHESTER",
    "Country": "UNITED KINGDOM",
    "Type": "2",
    "Ranges": [],
    "user_fields": []
  }
]
}

```

Spectrum Global Geocoding

Introduction to Spectrum Global Geocoding Services

The Spectrum 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

Making Requests using HTTP

WADL URL

The WADL for the 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 |
|---------------------------|--|-----------------------|
| <i>service_name</i> .json | No special header information. | json |
| <i>service_name</i> .json | Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8 | xml |
| <i>service_name</i> .json | Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8 | json |
| <i>service_name</i> | Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8 | json |
| <i>service_name</i> | Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8 | xml |
| <i>service_name</i> | No special header information. | json |
| <i>service_name</i> .xml | Content-Type: application/json; charset=utf-8 Accept: application/json; charset=utf-8 | json |
| <i>service_name</i> .xml | Content-Type: application/xml; charset=utf-8 Accept: application/xml; charset=utf-8 | xml |
| <i>service_name</i> .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**) The The The The 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 <code>GET</code> or a <code>POST</code> 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 <code>Accept</code> header is not supported. The supported media types include <code>application/JSON</code> and <code>application/xml</code> . |
| 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>/<context>/rest/GeocodeService/<name_of_service>
```

See [Example: JSON POST Request & Response](#) on page 229.

```
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>/<context>/rest/GeocodeService/<name_of_service>
```

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

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

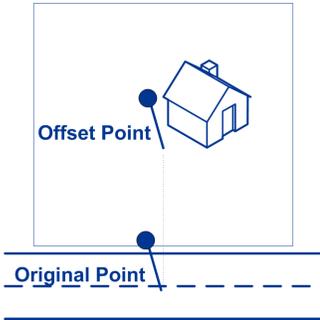
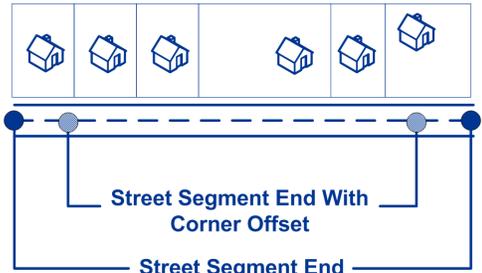
Query Parameters

The following table defines the `GET` query parameters for the `Geocode` service. For information on the response, see [GeocodeServiceResponse Object](#) on page 218.

| Parameter | Type | Description |
|---|--------|--|
| POST: mainAddressLine GET: mainAddress | String | <p>Single Line input—If no other field is populated, then the <code>mainAddress</code> 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:</p> <p>4750 Walnut St., Boulder CO, 80301</p> <p>Multiline Input If the address fields (<code>placeName</code>, <code>lastLine</code>, <code>postalCode</code>, 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.</p> <p>Street Intersection Input—To enter an intersection, specify the two street names separated by a double ampersand (&&).</p> |
| country | String | ISO 3166-1 alpha-3 country code. Required. For country codes, see ISO 3166-1 Country Codes . |
| 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. |

| Parameter | Type | Description |
|--|---------|---|
| placeName | String | Building name, place name, Point of Interest (POI), company or firm name associated with the input address. Optional. For example: Precisely 4750 Walnut St. Boulder, CO 80301 |
| 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 <code>MustMatch</code> fields; however, you can only set the <code>MustMatch</code> fields using a <code>POST</code> request. For a <code>GET</code> request, the <code>MustMatch</code> 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. |

| Parameter | Type | Description |
|----------------|---------|--|
| fallbackPostal | Boolean | <p>Specifies whether to attempt to determine a post code centroid when an address-level geocode cannot be determined. Optional.</p> <p>true Return a post code centroid when an address-level centroid cannot be determined. Default.</p> <p>false Do not return a post code centroid when an address-level centroid cannot be determined.</p> |
| maxCands | Integer | <p>The maximum number of candidates to return. Optional. Must be an integer value. Default = 1.</p> |
| maxRanges | Integer | <p>A range is a series of addresses along a street segment. For example, 5400-5499 Main St. represents 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.</p> <p>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.</p> <p>Must be an integer value. Default = 0.</p> |
| maxRangeUnits | Integer | <p>This option specifies the maximum number of units (for example, apartments or suites) to return for each range.</p> <p>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.</p> <p>Must be an integer value. Default = 0.</p> |

| Parameter | Type | Description |
|-------------------|--------|--|
| streetOffset | Double | <p>The offset distance from the street segments. The distance is in the units you specify in the <code>streetOffsetUnits</code> preference. Default value = 7 meters.</p> <p>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.</p> <p>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.</p>  |
| streetOffsetUnits | String | <p>Unit of measurement for the street offset. One of the following: Feet, Meters (default).</p> |
| cornerOffset | Double | <p>Distance to offset the street end points in street-level matching. The distance is in the units you specify in the <code>cornerOffsetUnits</code> 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.</p> <p>The following diagram compares the end points of a street to offset end points.</p>  |

| Parameter | Type | Description |
|-------------------|--------|---|
| 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>/<context>/rest/GeocodeService/<name_of_service>
```

See [Example: JSON POST Request & Response](#) on page 229.

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

Request Parameters

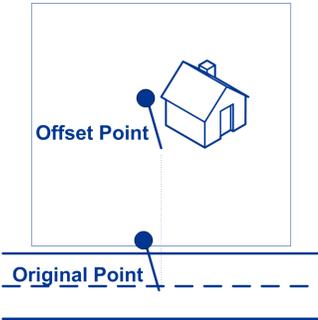
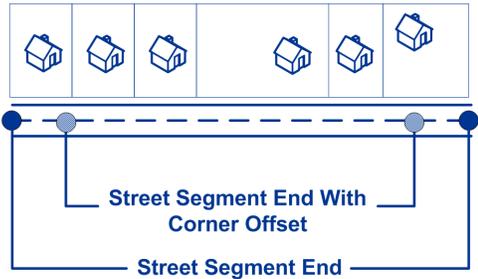
The `POST` request comprises the following input parameters:

- `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 | String | Indicates the geocode type to be performed. Optional. |
| GET: geocodeType | | <p>ADDRESS Geocode to a street address. Default.</p> <p>GEOGRAPHIC Geocode to the geographic centroid of a city or state.</p> <p>POSTAL Geocode to a postal code.</p> |

| Parameter | Type | Description |
|--|---------|---|
| 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. |
| 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. |

| Parameter | Type | Description |
|-------------------|--------|--|
| streetOffset | Double | <p>The offset distance from the street segments. The distance is in the units you specify in the <code>streetOffsetUnits</code> preference. Default value = 7 meters.</p> <p>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.</p> <p>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.</p>  |
| streetOffsetUnits | String | <p>Unit of measurement for the street offset. One of the following: Feet, Meters (default).</p> |
| cornerOffset | Double | <p>Distance to offset the street end points in street-level matching. The distance is in the units you specify in the <code>cornerOffsetUnits</code> 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.</p> <p>The following diagram compares the end points of a street to offset end points.</p>  |

| Parameter | Type | Description |
|-------------------|---------|--|
| cornerOffsetUnits | String | Unit of measurement for the street offset. One of the following: Feet , Meters (default). |
| matchMode | String | <p>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:</p> <p>EXACT Requires a very tight match. This restrictive mode generates the fewest match candidates, which decreases the processing time.</p> <p>STANDARD Requires a close match and generates a moderate number of match candidates. Default.</p> <p>RELAXED Allows a loose match and generates the most match candidates, which increases the processing time and results in more multiple matches.</p> <p>CUSTOM Provides the capability for you to define the matching criteria by setting <code>MustMatch</code> fields; however, you can only set the <code>MustMatch</code> fields using a <code>POST</code> request. For a <code>GET</code> request, the <code>MustMatch</code> default values are used.</p> |
| maxRanges | Integer | <p>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.</p> <p>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.</p> <p>Must be an integer value. Default = 0.</p> |
| maxRangeUnits | Integer | <p>This option specifies the maximum number of units (for example, apartments or suites) to return for each range.</p> <p>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.</p> <p>Must be an integer value. Default = 0.</p> |

| Parameter | Type | Description | |
|-----------------------------------|---------|---|--|
| POST: clientCoordSysName | String | <p>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 = <code>EPSG:4326</code>.</p> <p>Specify the coordinate reference system in the format <code>codespace:code</code>.</p> | |
| 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 <code>PostCode1</code> field. |
| | | false | A match does not need to be made to the input <code>PostCode1</code> field. Default. |
| POST: matchOnPostCode2 | Boolean | true | A match must be made to the input <code>PostCode2</code> field. |
| | | false | A match does not need to be made to the input <code>PostCode2</code> field. Default. |
| POST: matchOnAreaName1 | Boolean | true | A match must be made to the input <code>AreaName1</code> field. |
| | | false | A match does not need to be made to the input <code>AreaName1</code> field. Default. |
| POST: matchOnAreaName2 | Boolean | true | A match must be made to the input <code>AreaName2</code> field. |
| | | false | A match does not need to be made to the input <code>AreaName2</code> field. Default. |

Note: This option is not supported by USA.

| Parameter | Type | Description | |
|--------------------------------------|--------------|---|--|
| POST: matchOnAreaName3 | Boolean | true | A match must be made to the input <code>AreaName3</code> field. |
| | | false | A match does not need to be made to the input <code>AreaName3</code> field. Default. |
| POST: matchOnAreaName4 | Boolean | true | A match must be made to the input <code>AreaName4</code> field. |
| | | false | A match does not need to be made to the input <code>AreaName4</code> 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. |
| 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 <code>customFields</code> 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. |

| Parameter | Type | Description | |
|--|---------|--------------|--|
| POST: returnStreetAddressFields | Boolean | true | Return all of the individual street fields that make up the <code>formattedStreetAddress</code> field separately, as follows: <ul style="list-style-type: none"> • <code>MAIN_ADDRESS</code> • <code>THOROUGHFARE_TYPE</code> • <code>ADDRESS_ID</code> • <code>PRE_ADDRESS</code> • <code>POST_ADDRESS</code> • <code>PRE_DIRECTIONAL</code> • <code>POST_DIRECTIONAL</code> |
| | | false | Do not return the individual street fields separately; return these values in the <code>formattedStreetAddress</code> field. Default. |
| POST: returnUnitInformation | Boolean | true | Where available, return unit type and unit value information separately in the <code>unitType</code> and <code>unitValue</code> fields, as well as in the <code>formattedStreetAddress</code> field. |
| | | false | Where available, return unit type and unit value information only in the <code>formattedStreetAddress</code> 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 |
|--------------------------------------|---------|--|
| <code>totalPossibleCandidates</code> | Integer | Indicates the total number of possible candidates. |
| <code>totalMatches</code> | Integer | Indicates the total number of matches. |

| Name | Type | Description |
|--|---------|--|
| <p><code>candidates</code> object of type <code>Candidate</code>, consisting of an array with one or more match candidates and associated address, matching and location information. Contains the following elements:</p> | | |
| <code>precisionLevel</code> | Integer | <p>A code describing the precision of the geocode. One of the following:</p> <ul style="list-style-type: none"> 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. <p style="text-align: center;">Note: This field is not returned for USA. For geocode precision information for USA, see Location Codes.</p> |
| <code>formattedStreetAddress</code> | String | The formatted main address line. |
| <code>formattedLocationAddress</code> | String | The formatted last address line. |
| <code>identifier</code> | String | For street- or point-level candidates, this is usually the segment ID. |

| Name | Type | Description |
|---------------|--------|-------------|
| precisionCode | String | |

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

| Name | Type | Description |
|--|---------|--|
| | | <p>U Custom user dataset.</p> <p>Note: For more detailed information including country-specific meanings and values, see Global Result Codes.</p> |
| 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". |
| <p>matching object. Indicates what parts of the input matched; consisting of the following elements:</p> | | |
| matchOnAddressNumber | Boolean | <p>Indicates if the input address number matched the candidate's address number.</p> <p>True The input address number matched the candidate's address number.</p> <p>False The input address number did not match the candidate's address number.</p> |
| matchOnPostCode1 | Boolean | <p>Indicates if the input <code>postCode1</code> field matched the candidate's <code>postCode1</code> field.</p> <p>True The input <code>postCode1</code> matched the candidate's <code>postCode1</code>.</p> <p>False The input <code>postCode1</code> did not match the candidate's <code>postCode1</code>.</p> |
| matchOnPostCode2 | Boolean | <p>Indicates if the input <code>postCode2</code> field (post code extension) matched the candidate's <code>postCode2</code> field.</p> <p>True The input <code>postCode2</code> matched the candidate's <code>postCode2</code>.</p> <p>False The input <code>postCode2</code> did not match candidate's <code>postCode2</code></p> |
| matchOnAreaName1 | Boolean | <p>Indicates if the input <code>areaName1</code> field matched the candidate's <code>areaName1</code> field.</p> <p>True The input <code>areaName1</code> matched the candidate's <code>areaName1</code>.</p> <p>False The input <code>areaName1</code> did not match the candidate's <code>areaName1</code>.</p> |

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

| 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. |
| <i>geometry</i> object. Returned geocode consisting of the following elements: | | |
| coordinates | Double | The candidate's geocode, specified as x (longitude) and y (latitude) 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 <i>Point</i> . |
| <i>address</i> 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. |
| <p><code>ranges</code>: <code>CandidateRange</code> object. Contains information about a candidate's ranges, consisting of the following elements:</p> | | |
| 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 | <p>Provides information on the side of street that the candidate's range is located.</p> <p>LEFT The range is on the left side of the street.</p> <p>RIGHT The range is on the right side of the street.</p> <p>BOTH The range is on both the left and right side of the street.</p> <p>UNKNOWN No information is available on the side of the street this range is located.</p> |

| 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/Geocode/rest/GeocodeService/geocode.json?
mainAddress=SANTA ANA&country=Mex&areaName1=DISTRITO FEDERAL
&postalCode=44910 HTTP/1.1
```

```
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/Geocode/rest/GeocodeService/geocode.xml?
mainAddress=18 Merivales St&country=AUS&areaName1=QLD&postalCode=4101
HTTP/1.1

```

```

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>
  </candidates>
</GeocodeServiceResponse>

```

```

<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/Geocode/rest/GeocodeService/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
    }
]
}

```

```

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",

```



```

    <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>

```

```

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>
      <identififier>65277882</identififier>
      <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>
    </candidates>
  </responses>
</GeocodeServiceResponseList>

```

```
</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 [Geocoding Requests](#) on page 207.

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>/<contextpath>/rest/GeocodeService/reverseGeocode
[.content type]?[query parameters]
```

```
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 244.

| 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 . |
| 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 <code>codespace:code</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 <code>distanceUnits</code> . 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: <ul style="list-style-type: none"> • 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>/<contextpath>/rest/GeocodeService/reverseGeocode
[.content type]
```

```
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.

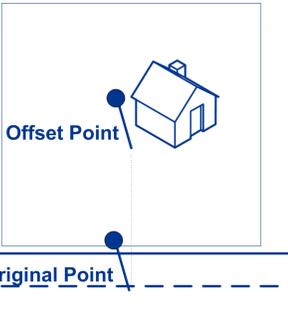
| Name | Type | Description |
|--------------------------|--------|---|
| <code>points</code> | | an array object containing both a geometry object and a country code string: |
| <code>country</code> | String | Indicates the country to search for the reverse geocode result, specified using a 3-letter ISO country code. Optional. For country codes, see ISO 3166-1 Country Codes . |
| <code>geometry</code> | | object, consisting of the following elements: |
| <code>coordinates</code> | Double | Specifies the x, y input coordinates, where x=longitude and y=latitude. For example: [-105.25175, 40.024494] |
| <code>type</code> | String | Indicates the type of geographic entity the input coordinates represent. point The input coordinates represent a point location. |
| <code>crs</code> | 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 = <code>EPSG:4326</code> . Specify the coordinate reference system in the format <code>codespace:code</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`.

| Name | Type | Description |
|---------------------------------|--------|---|
| 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 <code>distanceUnits</code> . 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: <ul style="list-style-type: none"> • 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 . |
| 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 = <code>EPSG:4326</code> . Specify the coordinate reference system in the format <code>codespace:code</code> . |

| Name | Type | Description |
|-------------------|--------|--|
| streetOffset | Double | <p>The offset distance from the street segments. The distance is in the units you specify in the <code>streetOffsetUnits</code> preference. Default value = 7 meters.</p> <p>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.</p> <p>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.</p>  |
| streetOffsetUnits | String | <p>Unit of measurement for the street offset. One of the following: Feet, Meters (default).</p> |
| cornerOffset | Double | <p>Distance to offset the street end points in street-level matching. The distance is in the units you specify in the <code>cornerOffsetUnits</code> 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.</p> <p>The following diagram compares the end points of a street to offset end points.</p>  |

| Name | Type | Description |
|-------------------|-------------------------------|--|
| cornerOffsetUnits | String | Unit of measurement for the street offset. One of the following: Feet , Meters (default). |
| customPreferences | Map<String key, String value> | <p>Specifies the country-specific input preferences. This object can be used to specify:</p> <ul style="list-style-type: none"> • A country override to a default value of one or more elements in the <code>preferences</code> or <code>returnFieldsDescriptor</code> objects. • A custom country input option. <p>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:</p> <pre><customPreferences> <entry> <key>CAN.distance</key> <value>300</value> </entry> </customPreferences></pre> <p>Custom country input options are available for the following countries:</p> <ul style="list-style-type: none"> • Australia (AUS) • Canada (CAN) • France (FRA) • Germany (DEU) • Great Britain (GBR) • New Zealand (NZL) • Portugal (PRT) • Singapore (SGP) • Sweden (SWE) • United States (USA) <p>In addition, for countries that support both custom user dictionaries and standard geocoding datasets, you can set a custom preference with the key <code>KEY_CUSTOM_DICTIONARY_USAGE</code> 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.</p> |

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 7: GeocodeServiceResponse Elements Definitions

| Name | Type | Description |
|--|---------|--|
| <code>totalPossibleCandidates</code> | Integer | Indicates the total number of possible candidates. |
| <code>totalMatches</code> | Integer | Indicates the total number of matches. |
| <code>candidates</code> object of type <code>Candidate</code> , 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 | <p>A code describing the precision of the geocode. One of the following:</p> <ul style="list-style-type: none"> 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. <p>Note: This field is not returned for USA. For geocode precision information for USA, see Location Codes.</p> |
| 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 ; for all other countries, see Reverse Geocoding 'R' Result Codes . |
| 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". |

| Name | Type | Description |
|---|--------|--|
| <code>geometry</code> object. Returned geocode consisting of the following elements: | | |
| <code>coordinates</code> | Double | The candidate's geocode, specified as x (longitude) and y (latitude) coordinates separated by a comma. |
| <code>crs</code> | String | The coordinate reference system used for the candidate's geocode. |
| <code>type</code> | String | Geometry type. The return value is always <code>Point</code> . |
| <code>address</code> object. Returned candidate address which may contain some of the following elements: | | |
| <code>mainAddressLine</code> | String | Candidate address line. |
| <code>addressLastLine</code> | String | Candidate last address line. |
| <code>placeName</code> | String | Firm, company, organization, business or building name. |
| <code>areaName1</code> | String | State, province or region. |
| <code>areaName2</code> | String | County or district. |
| <code>areaName3</code> | String | City, town or suburb. |
| <code>areaName4</code> | String | Locality |
| <code>postCode1</code> | String | Main postal code. |
| <code>postCode2</code> | String | Secondary postal code, where one exists. |
| <code>country</code> | String | Country |
| <code>addressNumber</code> | String | House or building number. |
| <code>streetName</code> | String | Street name. |
| <code>unitType</code> | String | The type of unit, such as Apt., Ste. and Bldg. |
| <code>unitValue</code> | String | The unit value/number, such as "3B". |

| Name | Type | Description |
|---|--------|---|
| customFields | Object | The fields and corresponding values returned are country-specific. |
| <i>ranges</i> : 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¶meter2=value`.

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

```
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": "0.9420000000000001"
        }
    },
    "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/Geocode/rest/GeocodeService/reverseGeocode.xml?
distanceUnits=METER&distance=100&coordSysName=EPSG:4326&y=51.543396
&x=13.419194 HTTP/1.1
```

```
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>
  </candidates>
</GeocodeServiceResponse>

```

```

<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/Geocode/rest/GeocodeService/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,
      "matchOnPostCode2": false,
      "matchOnAreaName1": false,
      "matchOnAreaName2": false,
      "matchOnAreaName3": false,
      "matchOnAreaName4": false,
      "matchOnAllStreetFields": 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"
      }
    }
  }
}
]
}

```

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",
  "identifiant": 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/Geocode/rest/GeocodeService/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>
    </mustMatchFields>
  </preferences>
</reverseGeocodeRequest>

```

```

    <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>

```

```

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>
    </mustMatchFields>
  </preferences>
</reverseGeocodeRequest>

```

```

    <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>
      </geometry>
    </candidates>
  </responses>
</GeocodeServiceResponseList>

```

```

    <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>
    </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 [Geocoding Requests](#) on page 207

Interactive Geocode Service Request

Global Interactive Geocode GET Request

A GET request to the 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/GeocodeService/geocode[.content
type]?[query parameters]
```

```
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 service. For information on the response, see [InteractiveGeocodeServiceResponse Object](#) on page 264.

| 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 |

POST Request

A `POST` request to the 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>/<contextpath>/rest/GeocodeService/geocode[.content
type]
```

```
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 USE_STANDARD_DICTIONARIES_ONLY |
| matchMode | string | default: STANDARD, possible values: RELAXED STANDARD, CLOSE |

| Parameter | Type | Description |
|------------------|-------------|--|
| originXY | List Double | <pre>{ "preferences" : { "originXY" : [-73.70252500000001, 42.68323] }, "address" : { "mainAddressLine" : "350 Jordan Rd" } }</pre> |
| restrictedSearch | Bounds | <pre>{ "preferences": { "restrictedSearch": {"northEastXY": [-73.70252500000001, 42.68323], "southWestXY": [-73.70252500000001, 42.68323] } }, "address": { "mainAddressLine": "350 Jordan Rd" } }</pre> |

Global Interactive Service Response

InteractiveGeocodeServiceResponse Object

For a list of response elements from the Interactive Geocode service, see [GeocodeServiceResponse Object](#) on page 218.

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,
        "matchOnPostCode": false,
        ...
        "matchOnStreetType": false,
        "matchOnStreetDirectional": false,
        "matchOnPlaceName": false,
        "matchOnInputFields": false
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
48.844045
          2.275675,
          ],
        "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 [Geocoding Requests](#) on page 207.

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>/Geocode/rest/GeocodeService/geocode[.content type]?[query parameters]
```

```
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 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 <code>PB_KEY</code> and <code>GNAF-PID</code> |
| country | string | 3-letter ISO code that represents the country for which the lookup is being performed. Currently <code>AUS</code> and <code>USA</code> 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>/<contextpath>/rest/GeocodeService/geocode[.content
type]
```

```
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 elements from the Key Lookup service, see [GeocodeServiceResponse Object](#) on page 218.

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",
"LRANGE": "350",
"STREET_SIDE": "L",
"DATATYPE": "12",
"SEG_LRANGE": "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>/<contextpath>/rest/GeocodeService/capabilities.[content
type]?[query parameters]
```

```
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. |

| Name | Description |
|-----------|--|
| operation | Type of geocoding service operation. One of the following: <ul style="list-style-type: none"> • 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. |
| supportedOperations | 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 | <p>Lists the required input fields for the operation. Includes the following elements:</p> <ul style="list-style-type: none">• name (String)• description (String)• type (String)• defaultValue (String)• lowBoundary (String)• highBoundary (String)• allowedValuesWithDescriptions (Map) |
| optionalInputs | InputParameter | <p>Lists the optional input fields for the operation. Includes the following elements:</p> <ul style="list-style-type: none">• name (String)• description (String)• type (String)• defaultValue (String)• lowBoundary (String)• highBoundary (String)• allowedValuesWithDescriptions (Map) |
| outputs | OutputParameter | <p>Lists the operation's output fields. Includes the following elements:</p> <ul style="list-style-type: none">• name (String)• description (String)• type (String) |

| Name | Type | Description |
|---------------|--------------|--|
| supportLevels | SupportLevel | <p>Lists the support levels for the operation. Includes the following elements:</p> <ul style="list-style-type: none"> supportedDataLevel (Integer) <ul style="list-style-type: none"> 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. <p>The data level will contain the sum of all available data keys. For example,</p> <p>Value — Type of data</p> <ul style="list-style-type: none"> 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 <ul style="list-style-type: none"> 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) 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: <ul style="list-style-type: none"> • 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/Geocode/rest/GeocodeService/capabilities.json?
country=GBR HTTP/1.1
```

```
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>/<contextpath>/rest/GeocodeService/dictionaries.[content type]?[query parameters]
```

```
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: <ul style="list-style-type: none"> • 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/Geocode/rest/GeocodeService/dictionaries.json?
country=FRA HTTP/1.1
```

```
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"
    ]
  }
]
}
```

Precisely Geocoding Connector

Introduction

The Precisely Geocoding Connector allows customers to integrate Spectrum Geocoding within third-party systems such as ArcGIS™ Online or ArcGIS™ Pro Desktop.

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

- Spectrum Global Geocoding Software Developer Kit (SDK)
- Spectrum Global Geocoding
- Location Intelligence GeoCode API (available at developer.precisely.com)

To download:

1. Go to <https://developer.precisely.com/apis/geocode/gis> and scroll down to **Quick Setup**.
2. In the ArcGIS Pro section, click **Download**.

Geocoding Operations

The Precisely 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 Precisely Geocoding Connector.

Parameters

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

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

| Parameter | Details |
|-----------------|----------------|
| 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 Precisely 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. |
| addresses | Required: Addresses for batch geocoding. SingleLine and multiline addresses are supported. |
| outFields | Optional: The list of fields to be returned. |

| Parameter | Details |
|--------------------|---|
| 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. |
| 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. |

| Parameter | Details |
|--------------------|----------------|
| 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. |
| 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. |

| Parameter | Details |
|-----------------|----------------|
| matchOutofRange | Not supported. |
| locationType | Not supported. |

Spectrum Global Sentry

GlobalSentry

The GlobalSentry service matches transactions against government-provided watch lists that contain data from various countries. These lists include:

- Denied Persons List (United States)
- Unverified List (BIS Red Flag) (United States)
- Consolidated Financial Sanction Targets (Individuals and Entities) (United Kingdom or European Union)
- Consolidated lists of persons, groups, and entities subject to EU financial sanctions (European Union)
- DFAT Consolidated List (Australia)
- OSFI Consolidated List (Individuals and Entities) (Canada)
- Specially Designated Nationals, Terrorists, Narcotic Traffickers and other Blocked Persons List (United States)
- Statutorily Debarred Parties List (United States)
- Politically Exposed Persons (PEP) list
- The consolidated Sanctions List including all individuals and entities who have been subjected to sanctions by the United Nations Security Council.

Matches are performed against Sanctioned Countries, Name, Address, ID Number and other information such as DOB to provide an "Overall Risk Level Score" that allows your organization to make the right choice before making a decision to block a particular transaction and avoid false positive results.

These steps describe how GlobalSentry processes data:

1. The service first scans all required data in the transaction to identify countries that have been sanctioned. If a sanction country match has been identified, the transaction bypasses all other matching criteria and is assigned the highest possible risk score.
2. If a sanctioned country match has not been identified, the service then attempts to match the transaction against the GlobalSentry database using the GlobalSentry Name Check, GlobalSentry Address Check or GlobalSentry ID Number Check subflows.

3. The GlobalSentry Name Check attempts to match individuals, entities and vessels. If a name match is identified a Name Score is returned from the service.
4. The GlobalSentry Address Check attempts to match addresses within a country. If an Address match is identified an Address Score is returned from the service.
5. The GlobalSentry ID Number Check attempts to match identification numbers, such as Passport, National ID, SSN, and Fiscal Code. If an ID Number match is identified an ID Number Score is returned from the service.
6. If a transaction is not identified as a Name, Address or ID Number match, the transaction record is written to the output and given an overall risk level score of zero.
7. If a transaction has been identified as a Name, Address or Identification Number match, the service attempts to match those transactions against the GlobalSentry database using the GlobalSentry Other Data Check subflow.
8. The GlobalSentry Other Data Check attempts to match the Place of Birth, Date of Birth, Nationality or Citizenship. If a match is identified a Place of Birth Score, Date of Birth Score, Nationality Score or Citizenship Score is returned by the service.
9. GlobalSentry assigns an Overall Risk Level score to each transaction. The score is a value between 0 and 16 and is returned in the OverallRiskLevel field. In calculating the risk level, GlobalSentry takes into account what data was provided in the input record and which inputs, if any, matched entries in the GlobalSentry database. Generally, a higher value indicates a higher risk associated with the transaction.

Resource URL

JSON endpoint:

```
http://server:port/rest/GlobalSentry/results.json
```

XML endpoint:

```
http://server:port/rest/GlobalSentry/results.xml
```

Example with JSON Response

This example requests a JSON response:

```
http://myserver:8080/rest/GlobalSentry/results.json?
Data.FirstName=Miguel&Data.LastName=Batista
```

The JSON returned by this request would be:

Note: Empty response elements have been removed from this example. Only the first response record shown.

```
{"Output": [
  {
```

```

    "OverallRiskLevel": "10",
    "SanctionedCountryIdentified": "No",
    "Status": "S",
    "FirstName": "Miguel",
    "LastName": "Batista",
    "PlaceOfBirth": "San Sebastian (Guipuzcoa) Spain",
    "EntryID": "315",
    "InputFilteredFirstName": "Miguel",
    "InputFilteredLastName": "Batista",
    "InputFirstName": "Miguel",
    "InputLastName": "Batista",
    "ListType": "DFAT Consolidated List",
    "MatchKey1": "MGL",
    "MatchKey2": "BTST",
    "NameMatchIdentified": "Yes",
    "NameProvided": "Yes",
    "AddressProvided": "No",
    "IDNumberProvided": "No",
    "AddressMatchIdentified": "No",
    "IDNumberMatchIdentified": "No",
    "CitizenshipScore": "0",
    "CitizenshipMatchIdentified": "No",
    "CitizenshipUID": "",
    "DOBScore": "0",
    "DOBMatchIdentified": "No",
    "NationalityScore": "0",
    "NationalityMatchIdentified": "No",
    "PlaceOfBirthScore": "0",
    "PlaceOfBirthMatchIdentified": "No",
    "CitizenshipProvided": "No",
    "DOBProvided": "No",
    "NationalityProvided": "No",
    "PlaceOfBirthProvided": "No",
    "WatchListFirstName": "Miguel",
    "WatchListLastName": "ALBISU IRIARTE",
    "NameScore": "100",
    "user_fields": []
  }
}

```

Example with XML Response

This example requests an XML response:

```

http://myserver:8080/rest/GlobalSentry/results.xml?
Data.FirstName=Miguel&Data.LastName=Batista

```

The XML response would be:

Note: Empty response elements have been removed from this example. Only the first response record shown.

```
<xml.GlobalSentryResponse
xmlns="http://www.precisely.com/spectrum/services/GlobalSentry">
  <Output>
    <Row>
      <OverallRiskLevel>10</OverallRiskLevel>
      <SanctionedCountryIdentified>No</SanctionedCountryIdentified>
      <Status>S</Status>
      <FirstName>Miguel</FirstName>
      <LastName>Batista</LastName>
      <PlaceOfBirth>San Sebastian (Guipuzcoa) Spain</PlaceOfBirth>
      <EntryID>315</EntryID>
      <InputFilteredFirstName>Miguel</InputFilteredFirstName>
      <InputFilteredLastName>Batista</InputFilteredLastName>
      <InputFirstName>Miguel</InputFirstName>
      <InputLastName>Batista</InputLastName>
      <ListType>DFAT Consolidated List</ListType>
      <MatchKey1>MGL</MatchKey1>
      <MatchKey2>BTST</MatchKey2>
      <NameMatchIdentified>Yes</NameMatchIdentified>
      <NameProvided>Yes</NameProvided>
      <AddressProvided>No</AddressProvided>
      <IDNumberProvided>No</IDNumberProvided>
      <AddressMatchIdentified>No</AddressMatchIdentified>
      <IDNumberMatchIdentified>No</IDNumberMatchIdentified>
      <CitizenshipScore>0</CitizenshipScore>
      <CitizenshipMatchIdentified>No</CitizenshipMatchIdentified>
      <DOBScore>0</DOBScore>
      <DOBMatchIdentified>No</DOBMatchIdentified>
      <NationalityScore>0</NationalityScore>
      <NationalityMatchIdentified>No</NationalityMatchIdentified>
      <PlaceOfBirthScore>0</PlaceOfBirthScore>
      <PlaceOfBirthMatchIdentified>No</PlaceOfBirthMatchIdentified>
      <CitizenshipProvided>No</CitizenshipProvided>
      <DOBProvided>No</DOBProvided>
      <NationalityProvided>No</NationalityProvided>
      <PlaceOfBirthProvided>No</PlaceOfBirthProvided>
      <WatchListFirstName>Miguel</WatchListFirstName>
      <WatchListLastName>ALBISU IRIARTE</WatchListLastName>
      <NameScore>100</NameScore>
      <user_fields/>
    </Row>
  </Output>
</xml.GlobalSentryResponse>
```

Request

Parameters for Input Data

Table 8: Global Sentry Input Fields

| Parameter | Description |
|-------------------|--|
| Data.Name | Full name. Required if FirstName and LastName is not used. |
| Data.FirstName | First name or all name elements other than last name. Required if Name is not used. |
| Data.LastName | Last name only. Required if Name is not used. |
| Data.AddressLine1 | The first address line. Recommended if provided. |
| Data.AddressLine2 | The second address line. Recommended if provided. |
| Data.AddressLine3 | The third address line. Recommended if provided. |
| Data.Country | Full country name. Required if address lines are used. |
| Data.IDNumber | Identification Number, such as SSN, Passport, and Visa. Recommended if provided. |

| Parameter | Description |
|-------------------|---|
| Data.PlaceOfBirth | Any place of birth data. Recommended if provided. |
| Data.DOB | Date Of Birth, in the format of Year, Month, Day. Recommended if provided. |
| Data.Citizenship | Full country name. Recommended if provided. |
| Data.Nationality | Full country name. Recommended if provided. |

Response

Table 9: Global Sentry Service Output

| Response Element | Description |
|--------------------|---|
| Status | Reports the success or failure of the match attempt. null : Success F : Failure |
| Status.Code | Reason for failure. |
| Status.Description | Description of the problem that caused the failure. |
| Name | |
| InputName | Input Name from the original data source. |

| Response Element | Description |
|------------------------|--|
| InputFilteredName | Input Name with titles, suffixes and special characters removed from the original data source. |
| Name | Name returned from database. |
| InputFirstName | Input First Name from the original data source. |
| InputFilteredFirstName | Input First Name with titles, suffixes and special characters removed from the original data source. |
| FirstName | First Name returned from database. |
| InputLastName | Input Last Name from the original data source. |
| InputFilteredLastName | Input Last Name with titles, suffixes and special characters removed from the original data source. |
| LastName | Last Name returned from database. |
| NameScore | Name match score. 0 - 100. |
| NameMatchIdentified | Identifies if the Name is a match. Values are Yes or No. |
| NameProvided | Identifies if the Name is provided in the input data . Values are Yes or No. |
| Address | |
| InputAddressLine1 | Input Address line from the original data source. |
| AddressLine1 | Address line returned from database. |

| Response Element | Description |
|-------------------------|--|
| InputAddressLine2 | Input Address line from the original data source. |
| AddressLine2 | Address line returned from database. |
| InputAddressLine3 | Input Address line from the original data source. |
| AddressLine3 | Address line returned from database. |
| AddressScore | Address match score. 0 - 100. |
| AddressMatchIdentified | Identifies if the Address is a match. Values are Yes or No. |
| AddressProvided | Identifies if the Address is provided in the input data. Values are Yes or No. |
| InputCountry | Input Country from the original data source. |
| Country | Country returned from database. |
| ID Number | |
| InputIDNumber | Input ID Number from the original data source. |
| IDNumber | ID Number returned from database. |
| IDNumberScore | ID Number match score. 0-100. |
| IDNumberMatchIdentified | Identifies if the ID Number is a match. Yes or No. |

| Response Element | Description |
|-----------------------------|---|
| IDNumberProvided | Identifies if the ID Number is provided in the input data. Values are Yes or No. |
| Place of Birth | |
| InputPlaceOfBirth | Input Place of Birth from the original data source. |
| PlaceOfBirth | Place of Birth returned from database. |
| PlaceOfBirthScore | Place of Birth match score. 0-100. |
| PlaceOfBirthMatchIdentified | Identifies if the Place of Birth is a match. Values are Yes or No. |
| PlaceOfBirthProvided | Identifies if the Place of Birth is provided in the input data. Values are Yes or No. |
| Date of Birth | |
| InputDOB | Input Date of Birth from the original data source. |
| DOB | Date of Birth returned from database. |
| DOBScore | Date of Birth match score. 0-100. |
| DOBMatchIdentified | Identifies if the Date of Birth is a match. Values are Yes or No. |
| DOBProvided | Identifies if the Date of Birth is provided in the input data. Values are Yes or No. |
| Citizenship | |

| Response Element | Description |
|------------------------------------|--|
| InputCitizenship | Input Citizenship from the original data source. |
| Citizenship | Citizenship returned from database. |
| CitizenshipScore | Citizenship match score. 0 to 100. |
| CitizenshipMatchIdentified | Identifies if Citizenship is a match. Values are <i>Yes</i> or <i>No</i> . |
| CitizenshipProvided | Identifies if the Citizenship is provided in the input data. Values are <i>Yes</i> or <i>No</i> . |
| Nationality | |
| InputNationality | Input Nationality from the original data source. |
| Nationality | Nationality returned from database. |
| NationalityScore | Nationality match score. 0-100. |
| NationalityMatchIdentified | Identifies Nationality was a match. Values are <i>Yes</i> or <i>No</i> . |
| NationalityProvided | Identifies if the Nationality is provided in the input data. Values are <i>Yes</i> or <i>No</i> . |
| Government List Information | |
| EntryID | Entry ID that identifies a name, entity, vessel, address, id number, place of birth, date of birth, citizenship or nationality. This is provided by each govt. agency. |
| ListType | Name of list provided by the government agencies. SDN, EU, Bank Of England, Financial Institutions of Canada. |

| Response Element | Description |
|-----------------------------|--|
| Risk Analysis | |
| OverAllRiskLevel | Risk score per match. 0-16. For more information, see Understanding the Risk Analysis Score on page 722. |
| SanctionedCountryIdentified | Indicates if the sanctioned country is identified as a match. Values are Yes or No. |

Understanding the Risk Analysis Score

Risk analysis processing assigns a point value to each of these inputs depending on whether the input was provided and whether it matched a record in the Global Sentry database. The risk analysis score is the sum of these point values. Points are assigned as shown in this table.

Table 10: Risk Analysis Scoring Method

| Input | No Data Provided | Matched | Did Not Match |
|----------------|------------------|---------|---------------|
| Name | 0 | 4 | 0 |
| Address | 1 | 2 | 0 |
| ID | 1 | 2 | 0 |
| Date of Birth | 1 | 2 | 0 |
| Place of Birth | 1 | 2 | 0 |
| Citizenship | 1 | 2 | 0 |
| Nationality | 1 | 2 | 0 |

Generally, each input that matches the database is assigned 2 points; Name is the exception. A name match scores 4 points. Name score is weighted higher following guidance from sources including OFAC, who indicate that a name match is more significant than other types of matches.

If an input is provided and does not match an entry on the database, it is assigned 0 points and has no effect on the overall risk level. This is consistent with guidance stating that a name match, coupled with a significant amount of additional data which does not match that entry in the database, should not be considered a "hit" against a particular list.

If an input is not provided, it is assigned a score of 1. This has the effect of identifying as higher risk those transactions where one or more inputs match the database, but there are some inputs which are not available for matching. For these types of transactions, the true risk level cannot be accurately calculated because of the missing data. Guidance from agencies such as OFAC suggests that in these cases you should attempt to obtain as much of the missing data as possible in order to return a more accurate assessment of the risk involved in the transaction.

Although higher scores indicate a higher risk transactions, the risk level alone is not always sufficient to determine the appropriate action. This is because different combinations of matched, not-matched, and not-provided inputs can result in the same score. To provide additional information to determine whether an interdiction is appropriate, the Global Sentry service also returns two indicators for each of the seven inputs that are used in matching. These indicate whether the input was provided and whether the input matched the database. This allows you to perform additional analysis on transactions that are in the middle of the risk spectrum to understand whether it is appropriate to report the transaction to the watch list authority, to flag the transaction as needing additional input data for an accurate risk assessment, to approve the transaction, or to take some other action.

Customizing the Global Sentry Service

Global Sentry deploys five dataflow templates that you can modify in Enterprise Designer. Each dataflow consists of various components that were installed from Spectrum Technology Platform Universal Name, Data Normalization, and Advanced Matching.

The names of the dataflows are:

- Global Sentry
- Global Sentry Name Check
- Global Sentry Address Check
- Global Sentry ID Number Check
- Global Sentry Other Data Check
- Global Sentry Batch
- Global Sentry Name Check Batch
- Global Sentry Address Check Batch
- Global Sentry ID Number Check Batch
- Global Sentry Other Data Check Batch

Spectrum Information Extractor

InformationExtractor

InformationExtractor extracts entities such as names and addresses from strings of unstructured data (also known as plain text).

It is possible that not all entities for any selected type will be returned because accuracy varies depending on the type of input. Because Information Extractor uses natural-language processing, a string containing a grammatically correct sentence from a news article or blog would likely have a more accurate return of names than a simple list of names and dates.

Resource URL

JSON endpoint:

```
http://server:port/rest/InformationExtractor/result.json
```

XML endpoint:

```
http://server:port/rest/InformationExtractor/result.xml
```

Example with JSON Response

This example requests a JSON response:

```
http://myserver:8080/rest/InformationExtractor/result.json?  
Data.PlainText=My+name+is+Arthur+Pitney&Option.EntityList=Person
```

The JSON returned by this request would be:

```
{"output_port": [{  
  "Entity": [ {  
    "Text": "Aurthur Pitney",  
    "Type": "Person"  
  } ],  
  "user_fields": []  
}]}
```

Example with XML Response

This example requests an XML response:

```
http://myserver:8080/rest/InformationExtractor/result.xml?
Data.PlainText=My+name+is+Arthur+Pitney&Option.EntityList=Person
```

The XML returned by this request would be:

```
<xml.InformationExtractorResponse
xmlns="http://www.precisely.com/spectrum/services/InformationExtractor">

  <output_port>
    <Result>
      <Entity>
        <Entity>
          <Text>Aurthur Pitney</Text>
          <Type>Person</Type>
        </Entity>
      </Entity>
      <user_fields/>
    </Result>
  </output_port>
</xml.InformationExtractorResponse>
```

Request

Parameters for Input Data

InformationExtractor takes as input unstructured strings of data.

Table 11: Input Format

| Parameter | Description |
|----------------|---|
| Data.PlainText | The unstructured string of data from which you want to extract information. |

Options

The InformationExtractor stage enables you to select entities for output data. It auto-assigns attributes for the entity types that were brought in to this stage. However, you can use the Quick Add function and select any or all of the 15 attributes:

| Parameter | Description |
|--------------------------|---|
| Option.CategorizerName | Specifies which model to use for text categorization. |
| Option.CategoryCount | Specifies how many matching levels of the category should be output (closest match, closest plus second closest, etc.). |
| Option.EntityList | <p>Specifies the type of data you want to extract from the unstructured string. Specify one or more of these. Separate each entity type with a comma.</p> <ul style="list-style-type: none"> Address CreditCard Date Email HashTag ISBN Location Mention Organization Person Phone ProperNouns SSN WebAddress ZipCode |
| Option.OutputEntityCount | <p>Specifies whether to return a count of how many times a particular entity occurred in the output.</p> <ul style="list-style-type: none"> true Return a count of the entities found in the unstructured string. false Do not return a count of the entities found in the unstructured string. |

Response

The output from InformationExtractor is a list of the entities found in the input string. For example, if you selected an entity type of "Person," the output would be a list of the names found in the input string. Likewise, if you selected an entity type of "Date," the output would be a list of the dates found

in the input string. Each entity (whether it be a name, address, date, and so on) is returned only once even if the entity appears multiple times in the input string.

| Response Element | Description |
|------------------|--|
| Text | The text extracted from the string. |
| Type | The entity type of the extracted text. One of the following: Address CreditCard Date Email HashTag ISBN Location Mention Organization Person Phone ProperNouns SSN WebAddress ZipCode |
| Count | If the option to return a count is enabled, this field contains the number of times that particular entity appeared in the input. For example, if you choose to return Name entities and the input text contains five instances of the name "John," the name "John" will be included in the output just one time, with "Name" as the entity type, and "5" as the output count. |
| Category | If you used a categorizer, the predicted category for each record in the input file. |
| Rank | If you used a categorizer, the rank of categories from highest count to lowest count. |

Spectrum Spatial

Where to Find Documentation?

Spectrum Spatial provides spatial services that allows you to determine relationships between locations, areas, or points of interest and other business data, and visually show these relationships on a map. These services include:

- Geometry
- Feature
- Mapping
- MapTiling
- Named Resource
- Web Feature Service
- Web Map Service

Spectrum Spatial also includes routing services, which include:

- DescribeDatasets
- GetCapabilities
- GetRoute
- GetRouteCostMatrix
- GetSegmentData
- GetTravelBoundary
- PersistentUpdate

To learn about Spectrum Spatial services, see the *Spectrum Spatial Guide* on support.precisely.com.

What is the Routing Demo Page?

The Routing Demo page is an interactive user interface for demonstrating basic routing capabilities using the Spectrum Spatial routing REST services. It sends a REST request to the service endpoint, displays the request used, and shows the response on the map.

Currently, there are two REST services featured in the Routing Demo Page: `GetTravelBoundary` and `GetRoute`.

GetTravelBoundary

`GetTravelBoundary` determines a drive or walk time or distance boundary from a location. This feature obtains polygons corresponding to an isochrone or isodistance calculation. An isochrone is a polygon or set of points representing an area that can be traversed in a network from a starting point in a given amount of time. An isodistance is a polygon or set of points representing the area that is a

certain distance from the starting point. The `GetTravelBoundary` operation (also known as an iso definition) takes a starting point, a unit (linear or time), one or more costs as input and returns the resulting travel boundary. Costs refer to the amount of time or distance to use in calculating an iso. Multiple costs can also be given as input. In case of multiple costs, costs can also be provided as a comma delimited string. For detailed descriptions of all the `GetTravelBoundary` REST input parameters, see [GetTravelBoundary](#) on page 348.

GetRoute

`GetRoute` returns routing information for a set of two distinct points or multiple points. It takes a starting location and an ending location with optional intermediate points as input, and returns the route that is either the fastest or the shortest (time or distance). For detailed descriptions of all the `GetRoute` REST input parameters, see [GetRoute](#) on page 308.

Launching The Routing Demo Page

This procedure launches the Routing demo page from your web browser.

1. From the Spectrum Spatial section of the Welcome Page, navigate to the Sample Applications tab then click **Open Routing Demo Page** under **Routing Demo Page**. Alternatively, enter the URL for the Routing demo page web application: `http://<servername>:<port>/routingdemopage/` in the address bar of your browser
2. Sign in with your user name and password. These are the same credentials you use for the Spectrum Technology Platform.
3. Click **Next**.

Note: If you enter incorrect credentials, you are returned to the sign-in page with an error message indicating the user name or password was invalid.

The Routing demo page appears.

Using The Routing Demo Page

While using the interactive interface for demonstrating routing capabilities, follow these instructions:

- **Database** - Change the routing database resource used in the demo. To change the database resource, select the database resource from the drop down list. Only resources configured with Spectrum will show in this list.
- **Service Tab** - Select either Travel Boundary or Route for the appropriate service to demo.
- **Points** - Both services use points in the operations (Base, Start, or End Point). Either enter the point information manually, or right click a position on the map, and select the type of point.
- **Advanced Options** - Add additional routing options to the request. To add additional routing options, click on **Advanced Options** and provide the options. For more information on the service REST parameters and options, see [GetTravelBoundary](#) on page 348 or [GetRoute](#) on page 308.
- **Apply** - Initiate the REST request, returning the map image and displaying the REST request used to generate the map. To make a request, provide all the required parameters (*), and click **Apply**.

- **Clear** - Remove the map image, or the field data, or both. To clear the routing information on the map or the field data, click the **Clear** drop down and select either All, Data, or Map.
- **Change User** - Change the user which you are currently logged in. To change the user, click the **Change User** button, enter the User Name and Password, click **Next**.

GetRoute

Description

The `GetRoute` service returns routing information for a set of two distinct points or multiple points. It takes a starting location and an ending location with optional intermediate points as input, and returns the route that is either the fastest or the shortest.

Note: The response from the REST service will be in JSON format and the geometry returned will be in GeoJSON format. When a request contains invalid query parameters in the GET URL or an invalid payload for POST, a cumulative error response will be returned in a JSON array. The `value` node in the response JSON is deprecated. For error checking, only the `errors` node should be utilized.

HTTP GET URL Format

The following format is used for HTTP GET requests. HTTP GET is used for simple routes where no additional JSON payload is required. Intermediate points can also be added to the HTTP GET request.

```
HTTP GET
/rest/Spatial/erm/databases/dbsource.json?q=route&query_parameters
```

Where *dbsource* is the name of the database that contains the data to use for the route. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

HTTP POST URL Format

The following format is used for HTTP POST requests:

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=route&query_parameters
POST BODY: Content-Type:application/json {Route Data}
```

`Route Data` is the POST json body (Content-Type: application/json) for the additional route information to be used in the calculation containing intermediate points, transient updates, or priority for road types. For information and examples on these options, see [GetRoute HTTP POST Options](#) on page 317.

Query Parameters

This operation takes the following query parameters.

| Parameter | Type | Required | Description |
|---------------------------------------|---------|----------|--|
| <code>startPoint</code> | String | Yes | The start location of the route in the format: <code>x,y,coordSys</code> . For example: <code>-74.2,40.8,epsg:4326</code> |
| <code>endPoint</code> | String | Yes | The end location of the route in the format: <code>x,y,coordSys</code> . For example: <code>-74.2,40.8,epsg:4326</code> |
| <code>intermediatePoints</code> | String | No | A list of intermediate points to include along the route. To include in the HTTP GET request, use the format: <code>Long,Lat,Long,Lat,...,coordsys</code> . For example: <code>-74.2,40.8,-73,42,epsg:4326</code> . To include a set of intermediate points in a HTTP POST request, add the MultiPoint JSON payload indicating the points that the route will include. If intermediate points are specified both in the URL and in the json payload, the json payload is given preference, and the intermediate points in URL are ignored. |
| <code>oip</code> | Boolean | No | A processing parameter that indicates if the intermediate points should be optimized. The default is false. By default the intermediate points will be used in the calculation in the order specified. If set to true, the specified points will be re-ordered in an optimal manner during route computation. |
| <code>returnIntermediatePoints</code> | Boolean | No | Whether to return the intermediate points with the route response. The default is false. For any value other than true or false, it will default to false. This option will return the intermediate points in order as specified in the POST body. If the value of option <code>oip</code> is set to true, then this option will return intermediate points in optimized order. |
| <code>destinationSrs</code> | String | No | The coordinate system to return the route and resulting geometries. The default is the coordinate system of the data used. |
| <code>optimizeBy</code> | String | No | The type of optimizing to use for the route. Valid values are <i>time</i> or <i>distance</i> . The default is time. |

| Parameter | Type | Required | Description |
|-----------------------------|---------|----------|--|
| <code>returnDistance</code> | Boolean | No | The route directions include the distance traveled. The default is true. |
| <code>distanceUnit</code> | String | No | The units to return distance. The default is m (meter). Available values are: m(meter), km(kilometer), yd(yard), ft(foot), mi(mile). |
| <code>returnTime</code> | Boolean | No | The route directions include the time it takes to follow a direction. The default is true. |
| <code>timeUnit</code> | String | No | The units to return time. The default is min (minute). Available values are: min(minute), msec (millisecond), s(second), h(hour). |

| Parameter | Type | Required | Description |
|-----------|--------|----------|---|
| language | String | No | <p>The language the travel directions should be returned, only if route directions are returned (if <code>directionsStyle</code> is defined as Normal or Terse). The default being English (en).</p> <p>Directions can be returned in the following languages:</p> <p>sq Return directions in Albanian.</p> <p>zh_CN Return directions in Chinese.</p> <p>zh_TW Return directions in Chinese (Taiwan).</p> <p>hr Return directions in Croatian.</p> <p>cs Return directions in Czech.</p> <p>da Return directions in Danish.</p> <p>nl Return directions in Dutch.</p> <p>en Return directions in English. Default</p> <p>en-US Return directions in American English.</p> <p>et Return directions in Estonian.</p> <p>fi Return directions in Finnish.</p> <p>fr Return directions in French.</p> <p>de Return directions in German.</p> <p>hu Return directions in Hungarian.</p> <p>it Return directions in Italian.</p> <p>ja Return directions in Japanese.</p> <p>lv Return directions in Latvian.</p> <p>lt Return directions in Lithuanian.</p> <p>no Return directions in Norwegian.</p> <p>pt Return directions in Portuguese.</p> <p>ro Return directions in Romanian.</p> <p>sk Return directions in Slovak.</p> <p>sl Return directions in Slovenian.</p> <p>es Return directions in Spanish.</p> <p>sv Return directions in Swedish.</p> <p>ru Return directions in Russian.</p> <p>tr Return directions in Turkish.</p> |

| Parameter | Type | Required | Description |
|-------------------------|---------|----------|---|
| returnDirectionGeometry | Boolean | No | Include the geometry associated with each route instruction in route response. The default is false. |
| directionsStyle | String | No | <p>The type of route directions to return. Default value is None. Specify this parameter if you required route directions to be returned. The options when specifying route directions are:</p> <p>None No directions returned. Default, if not specified.</p> <p>Normal Directions are returned in a full format, appropriate for web-based applications.</p> <p>Terse Directions are returned in a short format, appropriate for mobile applications.</p> |
| segmentGeometryStyle | String | No | <p>The format of the geometry that represents a segment of the route. Default value is None. Specify this parameter if you required segment geometries to be returned. The options when specifying route directions are:</p> <p>None No geometric representation of a segment will be returned. Default, if not specified.</p> <p>End Each segment of the route will be returned with just its endpoints in a LineString.</p> <p>All Each segment will be returned with all its shape points as a LineString. The LineString can be used as an overlay on a map.</p> |
| primaryNameOnly | Boolean | No | Whether to return all names for a given street in the directions or to return just the primary name for a street. Only used when route directions are returned. The default being false. |
| majorRoads | Boolean | No | Whether to include all roads in the calculation or just major roads. If you choose to include only major roads, performance will improve but accuracy may decrease. The default is false. |

| Parameter | Type | Required | Description |
|--|--------|----------|---|
| <code>historicTrafficTimeBucket</code> | String | No | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |
| <code>avoid</code> | String | No | <p>Specifies a comma-separated list of road types to be avoided during route calculation. This is a String parameter. When a road type is provided as the value of the parameter, the route excludes that type of roads in route calculation. For example, if <code>tollRoad</code> is provided as the parameter value, the calculated route contains a route without any toll roads.</p> |
| <code>version</code> | String | No | <p>Specifies the version of the <code>GetRoute</code> REST service. Valid values are <code>1</code> and <code>2</code>. The default value for <code>version</code> is <code>1</code>.</p> |

| Parameter | Type | Required | Description |
|----------------------|--------|----------|--|
| localRoadsLoadFactor | String | No | Specifies the number of local roads that can be loaded into memory during route or matrix calculation. The number of roads that can load is directly proportional to the value selected in the parameter where 1 is the minimum and 3 is the maximum value. Valid values can be 1,2, or 3. The default is 1. See Local Roads Load Factor for a detailed description and impact of the parameter on routing or matrix calculation. Note: The parameter does not accept decimal values. |

Examples

Simple Route with start and end points.

```
http://<server>:<port>/rest/Spatial/em/databases/usroute/database.json?route=startPoint=-73.97,40.79,eps:4326&endPoint=-73.98,40.74,eps:4326
```

Response

```
{
  "distance": 7779,
  "distanceUnit": "m",
  "time": 16.75,
  "timeUnit": "min"
}
```

Route with intermediate points.

```
http://<server>:<port>/rest/Spatial/em/databases/usroute/database.json?route=startPoint=-73.970257,40.794045,eps:4326&endPoint=-73.972103,40.7865,eps:4326&intermediatePoints=-73.97266,40.788717,-73.973562,40.792193,-73.971802,40.79463,eps:4326&returnIntermediatePoints=true'>
```

Response

```
{
  "distance": 1921,
  "distanceUnit": "m",
  "intermediatePoints": {
    "type": "MultiPoint",
    "coordinates": [
      [-73.971802, 40.79463],
      [-73.973562, 40.792193],
      [-73.97266, 40.788717]
    ]
  },
}
```

```

    "time": 4.2,
    "timeUnit": "min"
  }

```

Route with directions enabled.

```

http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=route&startPoint=-73.97,40.79,epsg:4326&endPoint=-73.98,40.74,
epsg:4326&language=en&directionsStyle=Normal&returnDirectionGeometry=true

```

Response

```

{
  "time": 10.58,
  "timeUnit": "min",
  "distance": 9035,
  "distanceUnit": "m",
  "language": "en",
  "directionsStyle": "Normal",
  "routeDirections": [
    {
      "time": 0.03,
      "timeUnit": "min",
      "distance": 25,
      "distanceUnit": "m",
      "instruction": "",
      "directionGeometry":
      {
        "type": "LineString",
        "coordinates":
        [
          [
            -76.421169,
            42.69302
          ],
          [
            -76.421353,
            42.692645
          ],
          ...
        ]
      }
    },
    {
      "time": 0.7,
      "timeUnit": "min",
      "distance": 394,
      "distanceUnit": "m",
      "instruction": "Turn right on W 91st St and travel West 394.0 m
(0.7 min).",
      "directionGeometry":
      {

```

```

    "type": "LineString",
    "coordinates":
    [
      [
        -76.429896,
        42.67153
      ],
      ...
    ]
  }
}
]
}

```

Version specific error response

When you enter an invalid parameter value (for example, points falling outside of the boundaries) in a request, the error response you get depends on the version entered by you. When the version is 1, you get value and error whereas when the version is 2, the response only contains the error.

- Request when version is 1:

```

http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
&q=route&startPoint=-14.321600,60.662859,epsg:4326&endPoint=-74.035208,40.695624,
epsg:4326&distanceUnit=km&version=1

```

- Response:

```

{
  "value": "Point outside boundaries: (-14.3216,60.662859,0)",
  "errors": [
    {
      "errorCode": 5008,
      "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
    }
  ]
}

```

- Request when version is 2:

```

http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
&q=route&startPoint=-14.321600,60.662859,epsg:4326&endPoint=-74.035208,40.695624,
epsg:4326&distanceUnit=km&version=2

```

- Response:

```

{
  "errors": [
    {

```

```

    "errorCode": 5008,
    "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
  }
]
}

```

GetRoute HTTP POST Options

HTTP POST URL Format

In addition to the regular HTTP GET parameters, you can add HTTP POST payload options to your request that specifies intermediate points, transient updates, and priority for road types. The content type must be set to application/json. The following format is used for HTTP POST requests:

```

HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=route&query_parameters
POST BODY: Content-Type:application/json {Route Data}

```

Intermediate Points

A list of intermediate points to include along the route. To include a set of intermediate points in a HTTP POST request, add the MultiPoint JSON payload indicating the points that the route will include. If intermediate points are specified both in the URL and in the json payload, the json payload is given preference, and the intermediate points in URL are ignored.

Example intermediate points HTTP POST payload.

```

{
  "intermediatePoints": {"type": "MultiPoint", "crs": {"type":
"name", "properties": {"name": "epsg:4326"}}, "coordinates": [[
-73.976266, 40.788717], [ -73.973562, 40.792193], [ -73.971802, 40.794630]]}
}

```

Transient Updates

This set of preferences allows you to set transient updates for each request. For instance, you can request that the server attempt to avoid all of the major road types. Each request can contain one or more updates. For speed updates, positive speed value is a speed increase and negative speed value is a speed decrease. The following is a description of the transient update types:

| Update Type | Description |
|-------------|--|
| point | <p>Point updates are changes applied to a corresponding point (Latitude, Longitude). For a particular point, you can: exclude the point, set the speed of the point or change (increase or decrease) the speed of the point by a value or percentage. Use one of the following types of updates:</p> <p>percentage This is a speed update where you define an increase in the speed of the point by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> <p>speed This is a speed update where you define the new speed of the point by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>speedAdjustment This is a speed update where you define a change in the speed of the point by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>exclude This is a string value to exclude the specified point from the route calculation. To exclude a point you need to specify the point and include the exclude parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Update Type | Description |
|-------------|-------------|
|-------------|-------------|

| | |
|-----------|--|
| segmentID | |
|-----------|--|

| Update Type | Description |
|------------------------|--|
| | <p>Segment updates are changes applied to a corresponding segment ID. For a particular segment, you can: exclude the segment, set the speed of the segment, change (increase or decrease) the speed of the segment by a value or percentage, or change the road type of the segment. Use one of the following types of updates:</p> |
| percentage | <p>This is a speed update where you define an increase in the speed of the segmentID by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> |
| speed | <p>This is a speed update where you define the new speed of the segmentID by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| speedAdjustment | <p>This is a speed update where you define a change in the speed of the segmentID by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| exclude | <p>This is a string value to exclude the specified segmentID from the route calculation. To exclude a segmentID you need to specify the segmentID and include the exclude parameter defined as Y. Valid values are Y (yes) and N (no).</p> |
| roadType | <p>This is a string value to change the value of the road type for the segment for the route calculation.</p> <p>The <code>roadType</code> can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban |

| Update Type | Description |
|-------------|--|
| | <ul style="list-style-type: none">• major road dense urban• major road rural• major road suburban• major road urban• minor local road dense Urban• minor local road rural• minor local road suburban• minor local road urban• normal road dense urban• normal road rural• normal road rural• normal road urban• primary highway dense urban• primary highway rural• primary highway suburban• primary highway urban• ramp dense urban• ramp limited access• ramp major road• ramp primary highway• ramp rural• ramp secondary highway• ramp urban• ramp suburban• secondary highway dense urban• secondary highway rural• secondary highway suburban• secondary highway urban |

| Update Type | Description |
|-------------|---|
| roadType | <p>Road type updates are changes applied to a corresponding road type. For a particular road type, you can: set the speed of the roadtype, or change (increase or decrease) the speed of the road type by a value or percentage. Use one of the following types of updates:</p> <p>percentage This is a speed update where you define an increase in the speed of the road type by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> <p>speed This is a speed update where you define the new speed of the road type by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>speedAdjustment This is a speed update where you define a change in the speed of the road type by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |

Example transient update HTTP POST payload.

```
{
  "transientUpdates": [
    {
      "segmentID": "7e3396fc:151186f",
      "updates": [
        {"percentage": 26.0}
      ]
    },
    {
      "point": {"type": "Point",
      "crs":
      {
        "type": "name",
        "properties":
        {
          "name": "epsg:4326"
        }
      }
    },
      "coordinates":
      [
        -73.972776,
        40.795076
      ]
    }
  ]
}
```

```

    },
    "updates": [
      {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}
    ]
  },
  {
    "roadType": "major road dense urban",
    "updates": [
      {"speed": { "velocity": 25, "velocityUnit": "kph"}}
    ]
  }
]
}

```

Commercial Vehicle Restrictions

Commercial vehicle restrictions are composed of directives to the routing engine that guides the behavior and attributes of commercial vehicles making trips along the route. Depending upon vehicle attributes provided (such as height, width, length, weight) and the commercial vehicle restriction attributes present in the road network, decision is made whether to allow to route a particular vehicle over a segment or not. If there is no commercial vehicle restriction attribute present in road network, input restriction parameters will have no effect in the resultant route.

Following are the set of parameters for commercial vehicle restrictions:

| Option | Description |
|------------------------------|--|
| looseningBarrierRestrictions | Specifies that barriers will be removed when determining the route. These restrictions are most often when a commercial vehicle is prohibited from traversing a segment due to local ordinance or a commercial vehicle is allowed on the segment but only when it must (for example, last mile access, local delivery, and so on). Routes where a barrier has been removed will still have a higher route cost even if the route is shorter/faster than a route with no barrier. |

| Option | Description |
|-------------------|--|
| vehicleAttributes | <p>Specifies that details of the vehicle that will be restricted based on the type, height, weight, length, or width when determining the route. Commercial vehicles are divided into different types ranging from short trailers to long triples. The Commercial Vehicle Restrictions attribution is organized on a per-vehicle type basis. This means it is entirely possible for a segment to be preferred for one vehicle type and the same segment have a restriction for another type of vehicle. Use the following types of vehicle information:</p> <p>vehicleType Choose either ALL or one of the types of vehicles: STRAIGHT, SEMI_TRAILOR, STANDARD_DOUBLE, INTERMEDIATE_DOUBLE, LONG_DOUBLE, TRIPLE, OTHER_LONG_COMBINATION_VEHICLE.</p> <p>weight Specifies the maximum weight of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of weight are: kg, lb, mt, t.</p> <p>height Specifies the maximum height of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of height are: ft, yd, mi, m, km.</p> <p>length Specifies the maximum length of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of length are: ft, yd, mi, m, km.</p> <p>width Specifies the maximum width of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of width are: ft, yd, mi, m, km.</p> <p>Note: You need to specify either weight/height or length/width along with its corresponding unit.</p> |

Examples

Without Vehicle Restrictions

Request:

```
HTTP GET
/rest/Spatial/em/databases/US_CVR.json?q=routeCostMatrix&startPoints=-74.7221203,42.9737073,eps:4326&
endPoints=-74.6671887,42.8097083,eps:4326
```

Response:

```
{
  "matrix": [{
    "distance": 40025,
    "distanceUnit": "m",
    "endPoint": {
      "type": "Point",
```

```

    "coordinates": [-74.6671887, 42.8097083]
  },
  "startPoint": {
    "type": "Point",
    "coordinates": [-74.7221203, 42.9737073]
  },
  "time": 36.57,
  "timeUnit": "min"
}]
}

```

With Vehicle Restrictions

Request:

```

HTTP GET
/rest/Spatial/em/databases/US_CVR.json?q=routeCostMatrix&startPoints=-74.7221203,42.9737073,eps:4326&
endPoints=-74.6671887,42.8097083,eps:4326

```

Commercial Vehicle Restriction HTTP POST payload.

```

{
  "cvr": {
    "looseningBarrierRestrictions": "n", "vehicleAttributes": {
      "vehicleType": "ALL", "heightUnit": "meter", "height": "4", "weightUnit": "Kilogram", "weight": "40000"
    }
  }
}

```

Response:

```

{
  "matrix": [{
    "distance": 44933,
    "distanceUnit": "m",
    "endPoint": {
      "type": "Point",
      "coordinates": [-74.6671887, 42.8097083]
    },
    "startPoint": {
      "type": "Point",
      "coordinates": [-74.7221203, 42.9737073]
    },
    "time": 37.48,
    "timeUnit": "min"
  }]
}

```

the distance and time value has changed with CVR.

Road Type Priority

Specifies the priority to give to different types of roads when determining the route. The following is a description of the road type priority options:

| Option | Description |
|--------|---|
| high | Prefer the road type over other road types. |
| medium | Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium. |
| low | Prefer other road types over this road type. |
| avoid | Exclude the road type from routes if possible. It is not always possible to exclude a road type from the travel directions. Depending on the situation, the alternative to an avoided road type may be so poor that the software will choose a route that uses an avoided road type. Also, if the starting or ending point lies along a segment whose road type has been avoided, the software will still use that segment. |

Example road type priority HTTP POST payload.

```
{
  " roadTypesPriority ": {
    "RoadType.MajorRoadDenseUrban": "High",
    "RoadType.LimitedAccessDenseUrban": "Low",
    "RoadType.LimitedAccessRural": "Medium",
    "RoadType.PrimaryHighwayUrban": "Avoid"
  }
}
```

GetRouteCostMatrix

Description

The GetRouteCostMatrix service calculates the travel time and distances between an array of start and end locations and returns the route that is either the fastest or the shortest. The result determines the total time and distance of the individual routes (the route costs). For example if you input four start points and four end points, a total of 16 routes will be calculated.

Note: The response from the REST service will be in JSON format and the geometry returned will be in GeoJSON format. When a request contains invalid query parameters in the GET URL or an invalid payload for POST, a cumulative error response will be returned in a JSON array. The `value` node in the response JSON is deprecated. For error checking, only the `errors` node should be utilized.

HTTP GET URL Format

The following format is used for HTTP GET requests. HTTP GET is used for simple cost calculations where no additional JSON payload is required.

```
HTTP GET
/rest/Spatial/erm/databases/dbsource.json?q=routeCostMatrix&query_parameters
```

Where *dbsource* is the name of the database that contains the data to use for the route. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

HTTP POST URL Format

The following format is used for HTTP POST requests:

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=routeCostMatrix&query_parameters
POST BODY: Content-Type:application/json {Route Data}
```

Route Data is the POST json body (Content-Type: application/json) for the additional route information to be used in the calculation if the list of input points exceeds the limits of the caller's URL buffer, as well as including transient updates, or priority for road-types. For information and examples on these options, see [GetRouteCostMatrix HTTP POST Options](#) on page 337.

Query Parameters

This operation takes the following query parameters.

| Parameter | Type | Required | Description |
|--------------------------|--------|----------|--|
| <code>startPoints</code> | String | Yes | The start locations of the route in the format: long,lat,long,lat,...,coordSys. For example: -74.2,40.8,-73,42,epsg:4326 |
| <code>endPoints</code> | String | Yes | The end locations of the route in the format: long,lat,long,lat,...,coordSys. For example: -74.2,40.8,-73,42,epsg:4326 |

| Parameter | Type | Required | Description |
|--------------------------------------|---------|----------|---|
| <code>destinationSrs</code> | String | No | The coordinate system to return the route and resulting geometries. The default is the coordinate system of the data used. |
| <code>optimizeBy</code> | String | No | The type of optimizing to use for the route. Valid values are <i>time</i> or <i>distance</i> . The default is <i>time</i> . |
| <code>returnDistance</code> | Boolean | No | The route directions include the distance traveled. The default is true. Both <code>returnDistance</code> and <code>returnTime</code> parameters cannot be false in the same request. |
| <code>distanceUnit</code> | String | No | The units to return distance. The default is m (meter). Available values are: m(meter), km(kilometer), yd(yard), ft(foot), mi(mile). |
| <code>returnTime</code> | Boolean | No | The route directions include the time it takes to follow a direction. The default is true. Both <code>returnDistance</code> and <code>returnTime</code> parameters cannot be false in the same request. |
| <code>timeUnit</code> | String | No | The units to return time. The default is min (minute). Available values are: min(minute), msec (millisecond), s(second), h(hour). |
| <code>majorRoads</code> | Boolean | No | Whether to include all roads in the calculation or just major roads. If you choose to include only major roads, performance will improve but accuracy may decrease. The default is false. |
| <code>returnOptimalRoutesOnly</code> | Boolean | No | Whether to return only the optimized route for each start point/end point combination. The default is true. The optimized route is either the fastest route or the shortest distance, depending on the <i>optimizeBy</i> parameter. |

| Parameter | Type | Required | Description |
|--|--------|----------|---|
| <code>historicTrafficTimeBucket</code> | String | No | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |
| <code>avoid</code> | String | No | <p>Specifies a comma-separated list of road types to be avoided during route calculation. This is a String parameter. When a road type is provided as the value of the parameter, the route excludes that type of roads in route calculation. For example, if <code>tollRoad</code> is provided as the parameter value, the calculated route contains a route without any toll roads.</p> |
| <code>version</code> | String | No | <p>Specifies the version of the <code>GetRouteCostMatrix</code> REST service. Valid values are 1 and 2. The default value for <code>version</code> is 1.</p> |

| Parameter | Type | Required | Description |
|----------------------|--------|----------|---|
| localRoadsLoadFactor | String | No | <p>Specifies the number of local roads that can be loaded into memory during route or matrix calculation. The number of roads that can load is directly proportional to the value selected in the parameter where 1 is the minimum and 3 is the maximum value. Valid values can be 1,2, or 3. The default is 1.</p> <p>See Local Roads Load Factor for a detailed description and impact of the parameter on routing or matrix calculation.</p> <p>Note: The parameter does not accept decimal values.</p> |

Examples

Cost matrix route with two start and two end points.

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
routeCostMatrix&startPoints=-73.5656,40.5545,-73.4656,40.4545,eps:4326&endPoints=-73.3434,40.667,-73.1434,40.267,
eps:4326&returnOptimalRoutesOnly=true&optimizeBy=distance&distanceUnit=km&timeUnit=min&
majorRoads=true&destinationSrs=eps:4322&returnTime=false
```

Response

```
{
  "matrix": [{
    "distance": 35.258,
    "distanceUnit": "km",
    "endPoint": {
      "type": "Point",
      "coordinates": [-73.34345711862802, 40.66688488742393],
      "crs": {
        "type": "name",
        "properties": {
          "name": "epsg:4322"
        }
      }
    }
  }],
  "startPoint": {
    "type": "Point",
    "coordinates": [-73.56567672202618, 40.554384822358614],
    "crs": {
      "type": "name",
      "properties": {
        "name": "epsg:4322"
      }
    }
  }
}
```

```

    }
  },
  {
    "distance": 41.761,
    "distanceUnit": "km",
    "endPoint": {
      "type": "Point",
      "coordinates": [-73.34345711862802, 40.66688488742393],
      "crs": {
        "type": "name",
        "properties": {
          "name": "epsg:4322"
        }
      }
    },
    "startPoint": {
      "type": "Point",
      "coordinates": [-73.46567684021008, 40.454384834155185],
      "crs": {
        "type": "name",
        "properties": {
          "name": "epsg:4322"
        }
      }
    }
  }
}
]]
}

```

Version specific error response

When you enter an invalid parameter value (for example, points falling outside of the boundaries) in a request, the error response you get depends on the version entered by you. When the version is 1, you get value and error whereas when the version is 2, the response only contains the error.

- Request when version is 1:

```

http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=routeCostMatrix&startPoints=-73.56565,40.5545,-73.46565,40.4545,eps:4326&
endPoints=-14.321600,60.662859,-73.14343,40.267,eps:4326&version=1

```

- Response:

```

{
  "value": "Point outside boundaries: (-14.3216,60.662859,0)",
  "errors": [
    {
      "errorCode": 5008,
      "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
    }
  ]
}

```

```
]
}
```

- Request when version is 2:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=routeCostMatrix&startPoints=-73.56565,40.5545,-73.46565,40.4545,epsg:4326&
endPoints=-14.321600,60.662859,-73.14343,40.267,epsg:4326&version=2
```

- Response:

```
{
  "errors": [
    {
      "errorCode": 5008,
      "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
    }
  ]
}
```

GetSegmentData

Description

The GetSegmentData service returns segment information for a point or segment ID. When a point is specified, the closest route segments are returned. When a segment ID is specified, the route data for that specified route segment is returned.

Note: The response from the REST service will be in JSON format. When a request contains invalid query parameters in the GET URL or an invalid payload for POST, a cumulative error response will be returned in a JSON array. The `value` node in the response JSON is deprecated. For error checking, only the `errors` node should be utilized.

HTTP GET URL Format

The following format is used for HTTP GET requests. The HTTP GET requests are different for either returning segment data at a point, or returning segment data for a segment ID.

Returning data for a segment at a specified point:

```
HTTP GET
/rest/Spatial/erm/databases/dbsource/segments.json?point=x,y,srsName&query_parameters
```

Returning data for a specified segment:

```
HTTP GET
/rest/Spatial/erm/databases/dbsource/segments/segmentID.json?query_parameters
```

Where *dbsource* is the name of the database that contains the data to use for the route. Use the database name specified in the Spectrum Spatial Routing Database Resource tool. The *segmentID* is segment identifier you want to return the data.

Query Parameters

This operation takes these query parameters.

| Parameter | Type | Required | Description |
|-----------------------------|--------|----------|---|
| <code>destinationSrs</code> | String | no | The coordinate system to return the segment data and resulting geometry. The default is the coordinate system of the data used. |
| <code>distanceUnit</code> | String | no | The units to return distance. The default is m (meter). Available values are: m(meter), km(kilometer), yd(yard), ft(foot), mi(mile). |
| <code>timeUnit</code> | String | no | The units to return time. The default is min (minute). Available values are: min(minute), msec(millisecond), s(second), h(hour). |
| <code>velocityUnit</code> | String | no | The units to return speed. The default is mph (miles per hour). Available values are: mph(miles per hour) and kph(kilometers per hour). |
| <code>angularUnit</code> | String | no | The units to return turn angles. The default is deg(degree). Available values are: deg(degree), rad(radian), minute(minute), sec(second), grad(grad). |

| Parameter | Type | Required | Description |
|----------------------|--------|----------|---|
| segmentGeometryStyle | String | no | <p>The format of the geometry that represents a segment of the route. Default value is None. Specify this parameter if you required segment geometries to be returned. The options when specifying route directions are:</p> <p>None No geometric representation of a segment will be returned. Default, if not specified.</p> <p>End Each segment of the route will be returned with just its endpoints in a LineString.</p> <p>All Each segment will be returned with all its shape points as a LineString. The LineString can be used as an overlay on a map.</p> |
| version | String | No | Specifies the version of the <code>GetSegmentData</code> REST service. Valid values are 1 and 2. The default value for version is 1. |

Examples

Return segment data specifying a point.

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase/segments.json?
point=-77,38,epsg:4326&segmentGeometryStyle=all
```

Response

```
[{
  "segmentID": "aa18eb33:1b7bbe",
  "primaryName": "VA-631",
  "primaryNameLanguage": "en",
  "alternateNames": [{
    "alternateName": "Lloyds Rd",
    "language": "en"
  }],
  {
    "alternateName": "VA-631",
    "language": "en"
  }],
  "segmentLength": 4.954,
  "segmentLengthUnit": "mi",
  "timeTaken": 5.9333,
  "timeUnit": "min",
  "turnAngle": 0.0,
  "turnAngleUnit": "deg",
```

```

"compassDirection": "",
"speedOfTravel": 49.9955,
"speedOfTravelUnit": "mph",
"roadType": "major road rural",
"segmentDirection": "bidirectional",
"startJunctionType": "",
"endJunctionType": "Other",
"isRoundabout": false,
"isTollRoad": false,
"geometry": {
  "type": "LineString",
  "crs": {
    "type": "name",
    "properties": {
      "name": "epsg:4326"
    }
  },
  "coordinates": [[...]]
},
{
  "segmentID": "46ed0e49:d9a7dc",
  "primaryName": "VA-631",
  "primaryNameLanguage": "en",
  "alternateNameList": [{
    "alternateName": "Lloyds Rd",
    "language": "en"
  }],
  "segmentLength": 1.198,
  "segmentLengthUnit": "mi",
  "timeTaken": 1.433,
  "timeUnit": "min",
  "turnAngle": 0.0,
  "turnAngleUnit": "degree",
  "compassDirection": "",
  "speedOfTravel": 49.9955,
  "speedOfTravelUnit": "mph",
  "roadType": "major road rural",
  "segmentDirection": "bidirectional",
  "startJunctionType": "Other",
  "endJunctionType": "",
  "isRoundabout": false,
  "isTollRoad": false,
  "pointsInSegment": {
    "type": "LineString",
    "crs": {
      "type": "name",
      "properties": {
        "name": "epsg:4326"
      }
    },
    "coordinates": [[...]]
  },
  "coordinates": [[...]]
}

```

```
}
}]
```

Return segment data specifying a segmentID.

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase/segments/aa18eb33:1b7bbe.json?
distanceUnits=mi
```

Response

```
[{
  "segmentID": "aa18eb33:1b7bbe",
  "primaryName": "VA-631",
  "primaryNameLanguage": "en",
  "alternateNames": [{
    "alternateName": "Lloyds Rd",
    "language": "en"
  }],
  {
    "alternateName": "VA-631",
    "language": "en"
  }],
  "segmentLength": 4.954,
  "segmentLengthUnit": "mi",
  "timeTaken": 5.9333,
  "timeUnit": "min",
  "turnAngle": 0.0,
  "turnAngleUnit": "deg",
  "compassDirection": "",
  "speedOfTravel": 49.9955,
  "speedOfTravelUnit": "mph",
  "roadType": "major road rural",
  "segmentDirection": "bidirectional",
  "startJunctionType": "",
  "endJunctionType": "Other",
  "isRoundabout": false,
  "isTollRoad": false
}]
```

Version specific error response

When you enter an invalid parameter value (for example, point missing) in a request, the error response you get depends on the version entered by you. When the version is 1, you get value and error whereas when the version is 2, the response only contains the error.

- Request when version is 1:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?version=1
```

- Response:

```
{
  "value": "Point cannot be empty.",
  "errors": [{
    "errorCode": 4139,
    "userMessage": "Point cannot be empty."
  }]
}
```

- Request when version is 2:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?version=2
```

- Response:

```
{"errors": [{
  "errorCode": 4139,
  "userMessage": "Point cannot be empty."
}]}
```

GetRouteCostMatrix HTTP POST Options

HTTP POST URL Format

In addition to the regular HTTP GET parameters, you can add HTTP POST payload options to your request that specifies transient updates and priority for road types. The HTTP POST payload can also be used if the list of input points exceeds the limits of the caller's URL buffer. The content type must be set to application/json. The following format is used for HTTP POST requests:

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=routeCostMatrix&query_parameters
POST BODY: Content-Type:application/json {Route Data}
```

Define Start and End Points

To include a set of start or end points in a HTTP POST request, add the MultiPoint JSON payload indicating the points that the route will include. When Start and End points are defined in the HTTP POST payload, the *startPoints* and *endPoints* parameters are not mandatory query parameters in the URL. If they are defined in the URL they are ignored. A warning message is logged in spectrum-server.log file when points in URL are ignored.

Example start points HTTP POST payload.

```
{
  "startPoints": {"type": "MultiPoint", "crs": {"type": "name", "properties":
```

```
{ "name": "epsg:4326"}, "coordinates": [[ -73.976266, 40.788717], [
-73.973562, 40.792193], [ -73.971802, 40.794630]]]
}
```

Example end points HTTP POST payload.

```
{
"endPoints": { "type": "MultiPoint", "crs": { "type": "name", "properties":
{ "name": "epsg:4326"}, "coordinates": [[ -73.976266, 40.788717], [
-73.973562, 40.792193], [ -73.971802, 40.794630]]]
}
```

Transient Updates

This set of preferences allows you to set transient updates for each request. For instance, you can request that the server attempt to avoid all of the major road types. Each request can contain one or more updates. For speed updates, positive speed value is a speed increase and negative speed value is a speed decrease. The following is a description of the transient update types:

| Update Type | Description |
|-------------|--|
| point | <p>Point updates are changes applied to a corresponding point (Latitude, Longitude). For a particular point, you can: exclude the point, set the speed of the point or change (increase or decrease) the speed of the point by a value or percentage. Use one of the following types of updates:</p> <p>percentage This is a speed update where you define an increase in the speed of the point by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> <p>speed This is a speed update where you define the new speed of the point by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>speedAdjustment This is a speed update where you define a change in the speed of the point by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>exclude This is a string value to exclude the specified point from the route calculation. To exclude a point you need to specify the point and include the exclude parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Update Type | Description |
|-------------|-------------|
|-------------|-------------|

| | |
|-----------|--|
| segmentID | |
|-----------|--|

| Update Type | Description |
|------------------------|--|
| | <p>Segment updates are changes applied to a corresponding segment ID. For a particular segment, you can: exclude the segment, set the speed of the segment, change (increase or decrease) the speed of the segment by a value or percentage, or change the road type of the segment. Use one of the following types of updates:</p> |
| percentage | <p>This is a speed update where you define an increase in the speed of the segmentID by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> |
| speed | <p>This is a speed update where you define the new speed of the segmentID by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| speedAdjustment | <p>This is a speed update where you define a change in the speed of the segmentID by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| exclude | <p>This is a string value to exclude the specified segmentID from the route calculation. To exclude a segmentID you need to specify the segmentID and include the exclude parameter defined as Y. Valid values are Y (yes) and N (no).</p> |
| roadType | <p>This is a string value to change the value of the road type for the segment for the route calculation.</p> <p>The <code>roadType</code> can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban |

Update Type

Description

-
- major road dense urban
 - major road rural
 - major road suburban
 - major road urban
 - minor local road dense Urban
 - minor local road rural
 - minor local road suburban
 - minor local road urban
 - normal road dense urban
 - normal road rural
 - normal road rural
 - normal road urban
 - primary highway dense urban
 - primary highway rural
 - primary highway suburban
 - primary highway urban
 - ramp dense urban
 - ramp limited access
 - ramp major road
 - ramp primary highway
 - ramp rural
 - ramp secondary highway
 - ramp urban
 - ramp suburban
 - secondary highway dense urban
 - secondary highway rural
 - secondary highway suburban
 - secondary highway urban
-

| Update Type | Description |
|-------------|---|
| roadType | <p>Road type updates are changes applied to a corresponding road type. For a particular road type, you can: set the speed of the roadtype, or change (increase or decrease) the speed of the road type by a value or percentage. Use one of the following types of updates:</p> <p>percentage This is a speed update where you define an increase in the speed of the road type by specifying a percentage to increase(positive value) or decrease(negative value) the speed.</p> <p>speed This is a speed update where you define the new speed of the road type by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>speedAdjustment This is a speed update where you define a change in the speed of the road type by specifying the change in velocity (unit and value). Speed values can be increased(positive value) or decreases(negative value). For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |

Example transient update HTTP POST payload.

```
{
  "transientUpdates": [
    {
      "segmentID": "7e3396fc:151186f",
      "updates": [
        {"percentage": 26.0}
      ]
    },
    {
      "point": {"type": "Point",
      "crs":
      {
        "type": "name",
        "properties":
        {
          "name": "epsg:4326"
        }
      }
    },
      "coordinates":
      [
        -73.972776,
        40.795076
      ]
    }
  ]
}
```

```

    },
    "updates": [
      {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}
    ]
  },
  {
    "roadType": "major road dense urban",
    "updates": [
      {"speed": { "velocity": 25, "velocityUnit": "kph"}}
    ]
  }
]
}

```

Commercial Vehicle Restrictions

Commercial vehicle restrictions are composed of directives to the routing engine that guides the behavior and attributes of commercial vehicles making trips along the route. Depending upon vehicle attributes provided (such as height, width, length, weight) and the commercial vehicle restriction attributes present in the road network, decision is made whether to allow to route a particular vehicle over a segment or not. If there is no commercial vehicle restriction attribute present in road network, input restriction parameters will have no effect in the resultant route.

Following are the set of parameters for commercial vehicle restrictions:

| Option | Description |
|------------------------------|--|
| looseningBarrierRestrictions | Specifies that barriers will be removed when determining the route. These restrictions are most often when a commercial vehicle is prohibited from traversing a segment due to local ordinance or a commercial vehicle is allowed on the segment but only when it must (for example, last mile access, local delivery, and so on). Routes where a barrier has been removed will still have a higher route cost even if the route is shorter/faster than a route with no barrier. |

| Option | Description |
|-------------------|--|
| vehicleAttributes | <p>Specifies that details of the vehicle that will be restricted based on the type, height, weight, length, or width when determining the route. Commercial vehicles are divided into different types ranging from short trailers to long triples. The Commercial Vehicle Restrictions attribution is organized on a per-vehicle type basis. This means it is entirely possible for a segment to be preferred for one vehicle type and the same segment have a restriction for another type of vehicle. Use the following types of vehicle information:</p> <p>vehicleType Choose either ALL or one of the types of vehicles: STRAIGHT, SEMI_TRAILOR, STANDARD_DOUBLE, INTERMEDIATE_DOUBLE, LONG_DOUBLE, TRIPLE, OTHER_LONG_COMBINATION_VEHICLE.</p> <p>weight Specifies the maximum weight of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of weight are: kg, lb, mt, t.</p> <p>height Specifies the maximum height of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of height are: ft, yd, mi, m, km.</p> <p>length Specifies the maximum length of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of length are: ft, yd, mi, m, km.</p> <p>width Specifies the maximum width of a vehicle. Any vehicles over this value will be restricted when determining the route. The units of width are: ft, yd, mi, m, km.</p> <p>Note: You need to specify either weight/height or length/width along with its corresponding unit.</p> |

Examples

Without Vehicle Restrictions

Request:

```
HTTP GET
/rest/Spatial/em/databases/US_CVR.json?q=routeCostMatrix&startPoints=-74.7221203,42.9737073,eps:4326&
endPoints=-74.6671887,42.8097083,eps:4326
```

Response:

```
{
  "matrix": [{
    "distance": 40025,
    "distanceUnit": "m",
    "endPoint": {
      "type": "Point",
```

```

    "coordinates": [-74.6671887, 42.8097083]
  },
  "startPoint": {
    "type": "Point",
    "coordinates": [-74.7221203, 42.9737073]
  },
  "time": 36.57,
  "timeUnit": "min"
}]
}

```

With Vehicle Restrictions

Request:

```

HTTP GET
/rest/Spatial/em/databases/US_CVR.json?q=routeCostMatrix&startPoints=-74.7221203,42.9737073,eps:4326&
endPoints=-74.6671887,42.8097083,eps:4326

```

Commercial Vehicle Restriction HTTP POST payload.

```

{
  "cvr": {
    "looseningBarrierRestrictions": "n", "vehicleAttributes": {
      "vehicleType": "ALL", "heightUnit": "meter", "height": "4", "weightUnit": "Kilogram", "weight": "40000"
    }
  }
}

```

Response:

```

{
  "matrix": [{
    "distance": 44933,
    "distanceUnit": "m",
    "endPoint": {
      "type": "Point",
      "coordinates": [-74.6671887, 42.8097083]
    },
    "startPoint": {
      "type": "Point",
      "coordinates": [-74.7221203, 42.9737073]
    },
    "time": 37.48,
    "timeUnit": "min"
  }]
}

```

the distance and time value has changed with CVR.

Road Type Priority

Specifies the priority to give to different types of roads when determining the route. The following is a description of the road type priority options:

| Option | Description |
|--------|---|
| high | Prefer the road type over other road types. |
| medium | Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium. |
| low | Prefer other road types over this road type. |
| avoid | Exclude the road type from routes if possible. It is not always possible to exclude a road type from the travel directions. Depending on the situation, the alternative to an avoided road type may be so poor that the software will choose a route that uses an avoided road type. Also, if the starting or ending point lies along a segment whose road type has been avoided, the software will still use that segment. |

Example road type priority HTTP POST payload.

```
{
  "roadTypesPriority": {
    "RoadType.MajorRoadDenseUrban": "High",
    "RoadType.LimitedAccessDenseUrban": "Low",
    "RoadType.LimitedAccessRural": "Medium",
    "RoadType.PrimaryHighwayUrban": "Avoid"
  }
}
```

Matrix Partial Response and Warnings

The Matrix request excludes problematic points from the calculation to process the request. Any problematic points are, therefore, excluded from the response.

A response, which contains time and distance between points, also includes a warnings section with information about problematic points in the request that were excluded from the calculation. Warnings list the problematic points. Each object in the list contains the error code for that point and an error description.

The warning message is available in REST version 2, SOAP services introduced in version 2020.1 and later, SDK(GRS), and stages introduced in version 2020.1 and later.

Request

Example start points HTTP POST payload.

```
http://www.precisely.com/rest/Spatial/erm/databases/usroutedatabase.json?
q=routeCostMatrix&startPoints=-74.015547,40.756962,-73.46565,40.4545,
epsg:4326&endPoints=-73.47565,40.4645,eps:4326&version=2
```

Response

```
{
  "matrix": [
    {
      "distanceUnit": "m",
      "distance": 30771.0,
      "timeUnit": "min",
      "time": 89.93,
      "startPoint": {
        "type": "Point",
        "coordinates": [
          -73.46565,
          40.4545
        ]
      },
      "endPoint": {
        "type": "Point",
        "coordinates": [
          -73.47565,
          40.4645
        ]
      }
    }
  ],
  "warnings": [
    {
      "code": 6002,
      "message": "Path could not be calculated between start point
(-74.015547,40.756962,0) and end point (-73.47565,40.4645,0)."
    }
  ]
}
```

GetTravelBoundary

Description

GetTravelBoundary determines a drive or walk time or distance boundary from a location. This feature obtains polygons corresponding to an isochrone or isodistance calculation. An isochrone is a polygon or set of points representing an area that can be traversed in a network from a starting point in a given amount of time. An isodistance is a polygon or set of points representing the area that is a certain distance from the starting point. The GetTravelBoundary operation (also known as an iso definition) takes a starting point, a unit (linear or time), one or more costs as input and returns the resulting travel boundary. Costs refer to the amount of time or distance to use in calculating an iso. Multiple costs can also be given as input. In case of multiple costs, costs can also be provided as a comma delimited string.

Note: The response from the REST service will be in JSON format. When a request contains invalid query parameters in the GET URL or an invalid payload for POST, a cumulative error response will be returned in a JSON array. The `value` node in the response JSON is deprecated. For error checking, only the `errors` node should be utilized.

HTTP GET URL Format

The following format is used for HTTP GET requests. HTTP GET is used for all travel boundaries where no additional JSON payload is required (ambient speed changes).

```
HTTP GET
/rest/Spatial/erm/databases/dbsource.json?q=travelBoundary&query_parameters
```

Where *dbsource* is the name of the database that contains the data to use for the route. Use the database name specified in the Enterprise Routing Module Routing Database Resource tool.

HTTP POST URL Format

The following format is used for HTTP POST requests:

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=travelBoundary&query_parameters
POST BODY: Content-Type:application/json {Route Data}
```

`Route Data` is the POST json body (Content-Type: application/json) for the additional route information to be used in the calculation containing ambient speeds for road types. For information and examples on these options, see [GetTravelBoundary HTTP POST Options](#) on page 355.

Query Parameters

This operation takes the following query parameters.

| Parameter | Type | Required | Description |
|---------------------------------|--------|----------|---|
| <code>point</code> | String | Yes | Specifies the start location from where to calculate the travel boundary in the format: x, y, coordSys. For example: -74.2,40.8,epsg:4326 |
| <code>costs</code> | Double | Yes | Specifies the cost distance or time, in the cost units specified (can be a decimal value). For example, if the unit specified is miles and you specify 10 in this parameter, the travel boundary will be calculated for how far you can travel 10 miles. You can also specify multiple costs by specifying the values as a comma delimited string. It will return a separate travel boundary for every cost specified. If you specify multiple costs, every response will have cost and cost units associated with that response. |
| <code>costUnit</code> | String | Yes | Specifies the type of metric used to calculate the travel boundary. Available distance values are: <ul style="list-style-type: none"> • m (meter) • km (kilometer) • yd (yard) • ft (foot) • mi (mile) Available time values are: <ul style="list-style-type: none"> • min (minute) • msec (millisecond) • s (second) • h (hour) |
| <code>maxOffroadDistance</code> | Double | No | The maximum distance to allow travel off the road network using the <code>maxOffroadDistanceUnit</code> . Examples of off-network roads include driveways and access roads. For example, if you specify a maximum off road distance of 1 mile the travel boundary will extend no further than one mile from the network road. If you specify a value of 0 the travel boundary will not extend beyond the road itself. Use the ambient speed options to specify the speed of travel along non-network roads. |

| Parameter | Type | Required | Description |
|-------------------------------------|---------|----------|---|
| <code>maxOffroadDistanceUnit</code> | String | No | Specifies the distance unit defining the <i>maxOffroadDistance</i> . You must also define <i>maxOffroadDistance</i> if you define this parameter. Available distance values are: <ul style="list-style-type: none"> • m (meter) • km (kilometer) • yd (yard) • ft (foot) • mi (mile) |
| <code>destinationSrs</code> | String | No | Specifies the coordinate system to return the travel boundary geometries. The default is the coordinate system of the data used (for example, epsg:4326). |
| <code>majorRoads</code> | Boolean | No | Specifies whether to include all roads in the calculation or just major roads. If you choose to include only major roads, performance will improve but accuracy may decrease. The default is true. |
| <code>returnHoles</code> | Boolean | No | Specifies whether you want to return holes, which are areas within the larger boundary that cannot be reached within the desired time or distance, based on the road network. The default is false. |
| <code>returnIslands</code> | Boolean | No | Specifies whether you want to return islands, which are small areas outside the main boundary that can be reached within the desired time or distance. The default is false. |
| <code>simplificationFactor</code> | Integer | No | Specifies the percentage of the original points to be returned or upon which the resulting complexity of geometries should be based. A number between 0.0 and 1.0 is accepted, exclusive of 0.0 but inclusive of 1.0. Complexity increases as the value increases, therefore 1.0 means the most complex. The default is 0.5. |

| Parameter | Type | Required | Description |
|---------------------------|--------|----------|---|
| bandingStyle | String | No | <p>Specifies the style of banding to be used in the result. Banding styles are the types of multiple distance bands that can be displayed based on multiple costs. Banding styles can be returned in the following formats:</p> <p>Donut Each boundary is determined by subtracting out the next smallest boundary. This is the default method.</p> <p>Encompassing Each boundary is determined independent of all others.</p> |
| historicTrafficTimeBucket | String | No | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |

| Parameter | Type | Required | Description |
|---------------------|--------|----------|---|
| defaultAmbientSpeed | String | No | <p>Specifies the speed to travel when going off a network road to find the travel boundary (for all road types). To control how off-network travel is used in the travel boundary calculation, you need to specify the speed of travel off the road network (the ambient speed). Ambient speed can affect the size and shape of the travel boundary polygon. In general, the faster the ambient speed, the larger the polygon. For example, if you were at a point with 5 minutes left, and if the ambient speed were 15 miles per hour, boundary points would be put at a distance of 1.25 miles. If the ambient speed were reduced to 10 miles per hour, boundary points would be put at a distance of 0.83 miles.</p> <p>Note: Default defaultAmbientSpeed is 15</p> |
| ambientSpeedUnit | String | No | <p>Specifies the unit of measure to calculate the ambient speed. Available speed units are:</p> <ul style="list-style-type: none"> • MPH (miles per hour) • KPH (kilometers per hour) • MTPS (meters per second) • MTPM (meters per minute) <p>Note: Default ambientSpeedUnit is MPH</p> |
| propagationFactor | String | No | <p>Specifies the percentage of the cost used to calculate the distance between the starting point and the isodistance (for all road types). Propagation factor serves the same purpose for isodistances as ambient speed does for isochrones, that is, it controls how off-network travel is used in the travel boundary calculation. Propagation factor can affect the size and shape of the travel boundary polygon. In general, more the propagation factor value, the larger the polygon.</p> <p>It applies to isodistances. If this property is not specified, then the calculation uses the server setting. Valid values are between 0.0 and 1.0, both inclusive.</p> <p>Note:</p> <ul style="list-style-type: none"> • Default propagationFactor is 0.16 • This parameter can also be specified in the POST body. If the same parameter is set in both GET and POST, then the value in POST is considered. |

| Parameter | Type | Required | Description |
|-----------|--------|----------|--|
| version | String | No | Specifies the version of the <code>GetTravelBoundary</code> REST service. Valid values are 1 and 2. The default value for <code>version</code> is 1. |

Examples

Travel boundary with a single cost.

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=travelBoundary&point=-77.092609,38.871256,epsg:4326&costs=5&costUnit=m
```

Response

```
{
  "travelBoundary": {
    "costs": [
      {
        "cost": 5,
        "costUnit": "m",
        "geometry": {"type": "MultiPolygon", "coordinates": [[[...]]]}
      }
    ]
  }
}
```

Travel boundary with multiple costs.

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=travelBoundary&point=-77.092609,38.871256,epsg:4326&costs=2,5&costUnit=m
```

Response

```
{
  "travelBoundary": {
    "costs": [
      {
        "cost": 2,
        "costUnit": "m",
        "geometry": {"type": "MultiPolygon", "coordinates": [[[...]]]}
      },
      {
        "cost": 5,
        "costUnit": "m",
        "geometry": {"type": "MultiPolygon", "coordinates": [[[...]]]}
      }
    ]
  }
}
```

```
}
}
```

Version specific error response

When you enter an invalid parameter value (for example, points falling outside of the boundaries) in a request, the error response you get depends on the version entered by you. When the version is 1, you get value and error whereas when the version is 2, the response only contains the error.

- Request when version is 1:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
q=travelBoundary&costs=5&costUnit=min&point=-14.321600,60.662859,epsg:4326&version=1
```

- Response:

```
{
  "value": "Point outside boundaries: (-14.3216,60.662859,0)",
  "errors": [
    {
      "errorCode": 5008,
      "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
    }
  ]
}
```

- Request when version is 2:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?q=travelBoundary
&costs=5&costUnit=min&point=-14.321600,60.662859,epsg:4326&version=2
```

- Response:

```
{
  "errors": [
    {
      "errorCode": 5008,
      "userMessage": "Point outside boundaries: (-14.3216,60.662859,0)"
    }
  ]
}
```

GetTravelBoundary HTTP POST Options

HTTP POST URL Format

In addition to the regular HTTP GET parameters, you can add an HTTP POST payload option to your request that specifies ambient speed changes for road types. The content type must be set to `application/json`. The following format is used for HTTP POST requests:

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource.json?q=travelBoundary&query_parameters
POST BODY: Content-Type:application/json {Route Data}
```

Route Data is the POST json body (Content-Type: `application/json`) for the additional route information to be used in the calculation containing ambient speeds for road types.

Ambient Speeds

This set of preferences allows you to set ambient speed changes for each request. An ambient speed is a change to the speed in the normal data to travel off a network road when finding the travel boundary. Examples of off-network travel include driveways and access roads. The following is a description of the ambient speed parameters:

| Parameter | Description |
|----------------------------|---|
| <i>DefaultAmbientSpeed</i> | Specifies the speed to travel when going off a network road to find the travel boundary (for all road types). To control how off-network travel is used in the travel boundary calculation, you need to specify the speed of travel off the road network (the ambient speed). Ambient speed can affect the size and shape of the travel boundary polygon. In general, the faster the ambient speed, the larger the polygon. For example, if you were at a point with 5 minutes left, and if the ambient speed were 15 miles per hour, boundary points would be put at a distance of 1.25 miles. If the ambient speed were reduced to 10 miles per hour, boundary points would be put at a distance of 0.83 miles. |

Note: Default `DefaultAmbientSpeed` is **15**

| Parameter | Description |
|-------------------------|--|
| <i>AmbientSpeedUnit</i> | <p>Specifies the unit of measure to calculate the ambient speed. Available speed units are:</p> <ul style="list-style-type: none">• MPH (miles per hour)• KPH (kilometers per hour)• MTPS (meters per second)• MTPM (meters per minute) <p>Note: Default <code>AmbientSpeedUnit</code> is MPH</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|------------------------------|--|
| <i>AmbientSpeed.RoadType</i> | |
|------------------------------|--|

| Parameter | Description |
|-----------|--|
| | <p data-bbox="548 369 1425 491">Specifies the ambient speed to use for off-network travel based on the road type. You must specify both the road type and the new speed for that road type. The speed is defined in the defined <code>AmbientSpeedUnit</code>. Road types can be returned in all supported types. The following road types can be used:</p> <ul data-bbox="548 516 886 1858" style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban |

| Parameter | Description |
|-------------------|--|
| | <ul style="list-style-type: none">• ramp suburban• secondary highway dense urban• secondary highway rural• secondary highway suburban• secondary highway urban |
| propagationFactor | See Query Parameters on page 348 for description of propagationFactor. Note: This parameter is only supported in <code>getTravelBoundary</code> version 2 is specified. |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|----------|--|
| RoadType | |
|----------|--|

| Parameter | Description |
|-----------|---|
| | <p>Specifies the <code>propagationFactor</code> to use for off-network travel based on the road type. You must specify both the road type and the new value of <code>propagationFactor</code> for that road type. See propagationFactor to know more. Road types can be returned in all supported types. The following road types can be used:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban • ramp suburban |

| Parameter | Description |
|--------------------|--|
| | <ul style="list-style-type: none"> secondary highway dense urban secondary highway rural secondary highway suburban secondary highway urban <p>Note: This parameter is only supported in <code>getTravelBoundary</code> version 2 is specified.</p> |
| <code>avoid</code> | Specifies a comma-separated list of road types to be avoided during travel boundary calculation. This is a String parameter. When a road type is provided as the value of the parameter, the boundary excludes that type of roads in the calculation. For example, if <code>tollRoad</code> is provided as the parameter value, the calculated boundary will have no toll roads. |

Example with ambient speed parameters in HTTP POST payload in `gettravelboundary` version 1.

```
{
  "DefaultAmbientSpeed": 45,
  "AmbientSpeedUnit": "MPH"

  "AmbientSpeed.RoadType.PrimaryHighwayUrban": 15,
  "AmbientSpeed.RoadType.SecondaryHighwayUrban": 10
}
```

Example with ambient speed and `propagationFactor` parameters in HTTP POST payload in `gettravelboundary` version 2.

```
{
  "ambientSpeeds": {
    "defaultAmbientSpeed": 24,
    "ambientSpeedUnit": "MPH",
    "ambientSpeedOverrides": {
      "Primary Highway Urban": ".51",
      "Secondary Highway Urban": ".1"
    }
  },
  "propagationFactors": {
    "propagationFactor": "1",
    "propagationFactorOverrides": {
      "Primary Highway Urban": ".51",
      "Secondary Highway Urban": ".1"
    }
  }
}
```

```

}
{
  "ambientSpeeds": {
    "ambientSpeedOverrides": {
      "Primary Highway Urban": 25,
      "Secondary Highway Urban": 10
    }
  },
  "propagationFactors": {
    "propagationFactor": "0.2",
    "propagationFactorOverrides": {
      "Primary Highway Urban": "0.51",
      "Secondary Highway Urban": "0.1"
    }
  }
}
}

```

Note: The response from REST request will be in JSON format and the geometries will be of GEOJSON format.

GetTravelBoundary (Deprecated)

Important: This stage has been deprecated in the 12.2 release. The **Travel Boundary** stage should be used instead when creating new dataflows.

GetTravelBoundary determines a drive or walk time or distance boundary from a location. This feature obtains polygons corresponding to an isochrone or isodistance calculation. An isochrone is a polygon or set of points representing an area that can be traversed in a network from a starting point in a given amount of time. An isodistance is a polygon or set of points representing the area that is a certain distance from the starting point. The Get Travel Boundary operation (also known as an iso definition) takes a starting point, a unit (linear or time), one or more costs and their associated tags as input and returns the resulting travel boundary. Cost refers to the amount of time or distance to use in calculating an iso. A tag is a string that identifies the cost and is used to match the corresponding result. Multiple costs can be given as input by providing the costs as a “;” delimited string.

GetTravelBoundary is part of Spectrum Spatial.

Note: GetTravelBoundary is only available as a web service. GetTravelBoundary is not available through the Java, C++, C, .NET, or COM APIs.

Resource URL

JSON endpoint:

```
http://server:port/rest/GetTravelBoundary/results.json
```

XML endpoint:

```
http://server:port/rest/GetTravelBoundary/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://<server>:<port>/rest/GetTravelBoundary/results.json?
Data.Latitude=51.5072&Data.Longitude=0.1275&Data.&Data.TravelBoundaryCost=10&
Data.TravelBondaryCostUnits=Kilometers&Option.DataSetResourceName=Routing_db_gb
```

The JSON returned by this request would be:

Note: Some of the points have been removed from this example to shorten it.

Note: The response in this example shows coordinates for the returned geometries with values to 14 decimal places. This contributes to a larger than necessary payload in the JSON response, especially when returning large polygons or many records. It also can lead to invalid geometries, such as self-intersections. To reduce the payload by returning shortened coordinate values, add the following to %Spectrum%\server\bin\wrapper\wrapper.conf and restart the server: wrapper.java.additional.xx=-Dcom.pb.midev.useprecision=true.

```
{
  "output_port": [
    {
      "IsoNodeResponse": [],
      "IsoPolygonResponse": {
        "srsName": "epsg:4326",
        "Polygon": [
          {
            "srsName": "epsg:4326",
            "Exterior": {
              "LineString": [
                {
                  "Pos": [
                    {
                      "X": -84.34868168466456,
                      "Y": 33.68373169496257
                    },
                    {
                      "X": -84.36945064055561,
                      "Y": 33.69293307108579
                    },
                    {
                      "X": -84.3694506405556,
                      "Y": 33.69293307108579
                    },
                    {
                      "X": -84.3694506405556,
                      "Y": 33.69303002973829
                    },
                    {
                      "X": -84.37104825254721,
                      "Y": 33.69391558543121
                    }
                  ]
                }
              ]
            }
          }
        ]
      }
    }
  ]
}
```

```

    },
    {
      "X": -84.37104825254721,
      "Y": 33.6936408692491
    },
    {
      "X": -84.42163929894845,
      "Y": 33.716054477754355
    },
    {
      "X": -84.4440058668311,
      "Y": 33.710741143596806
    },
    {
      "X": -84.43921303085625,
      "Y": 33.72800947960886
    },
    {
      "X": -84.45678676276404,
      "Y": 33.73376559161287
    },
    {
      "X": -84.43921303085625,
      "Y": 33.73996448146335
    },
    ...
  ]]]}
  ]}
},
"user_fields": [ {
  "name": "TravelBoundaryCostUnits",
  "value": "Kilometers"
}]
}]

```

Example with XML Response

The following example requests an XML response:

```

http://<server>:<port>/rest/GetTravelBoundary/results.xml?
Data.TravelBoundaryCostUnits=Kilometers&Data.Latitude=33.751748&
Data.Longitude=-84.364014&Data.TravelBoundaryCost=10

```

The XML returned by this request would be:

Note: Some of the points have been removed from this example to shorten it.

```

<ns3:xml.GetTravelBoundaryResponse
xmlns:ns2="http://www.mapinfo.com/midev/service/geometries/v1"
xmlns:ns3="http://<server>:<port>/spectrum/services/GetTravelBoundary">

```

```

<ns3:output_port>
  <ns3:IsoRouteResponse>
    <ns3:IsoNodeResponse/>
    <ns3:IsoPolygonResponse
      xsi:type="ns2:MultiPolygon"
      srsName="epsg:4326"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <ns2:Polygon srsName="epsg:4326">
      <ns2:Exterior>
        <ns2:LineString>
          <ns2:Pos>
            <ns2:X>-84.34868168466456</ns2:X>
            <ns2:Y>33.68373169496257</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.36945064055561</ns2:X>
            <ns2:Y>33.69293307108579</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.3694506405556</ns2:X>
            <ns2:Y>33.69293307108579</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.3694506405556</ns2:X>
            <ns2:Y>33.69303002973829</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.37104825254721</ns2:X>
            <ns2:Y>33.69391558543121</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.37104825254721</ns2:X>
            <ns2:Y>33.6936408692491</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.42163929894845</ns2:X>
            <ns2:Y>33.716054477754355</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.4440058668311</ns2:X>
            <ns2:Y>33.710741143596806</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.43921303085625</ns2:X>
            <ns2:Y>33.72800947960886</ns2:Y>
          </ns2:Pos>
          <ns2:Pos>
            <ns2:X>-84.45678676276404</ns2:X>
            <ns2:Y>33.73376559161287</ns2:Y>
          </ns2:Pos>
          ...
        </ns2:LineString>

```

```

        </ns2:Exterior>
        </ns2:Polygon>
    </ns3:IsoPolygonResponse>
    <ns3:user_fields/>
</ns3:IsoRouteResponse>
</ns3:output_port>
</ns3:xml.GetTravelBoundaryResponse>

```

Request

Parameters for Input Data

GetTravelBoundary takes cost, cost unit, point latitude, and point longitude as input. The following table provides information on the format and layout of the input.

Table 12: GetTravelBoundary Input Data

| Parameter | Format | Description |
|-------------------------|--------|--|
| Data.Latitude | String | Latitude of the point. Specify latitude in the format chosen in the Option.CoordinateFormat parameter. |
| Data.Longitude | String | Longitude of the point. Specify longitude in the format chosen in the Option.CoordinateFormat parameter. |
| Data.TravelBoundaryCost | String | <p>(Optional) The cost distance or time, in the units specified by either the Data.TravelBoundaryCostUnits parameter or the Option.DefaultTravelBoundaryCostUnits parameter. For example, if the unit specified is miles and you specify 10 in this field, the cost would be 10 miles.</p> <p>Use this field to override the default travel boundary cost on a record-by-record basis.</p> <p>You can also specify multiple costs by specifying the values as a ";" delimited string. It will return a separate Iso Route Response for every cost specified. If you specify multiple costs, every response will have cost and costUnits associated with that response.</p> |

| Parameter | Format | Description |
|------------------------------|--------|---|
| Data.TravelBoundaryCostUnits | String | <p>(Optional) The type of metric used to calculate the travel boundary. One of the following:</p> <ul style="list-style-type: none"> • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Yards <p>Use this field to override the default travel boundary cost units on a record-by-record basis.</p> |

Parameters for Options

Input

Table 13: GetTravelBoundary Input Options

| Parameter | Description |
|----------------------------|---|
| Option.DataSetResourceName | The name of the database that contains the data to use in the search process. Use a valid routing database resource name defined in the Resources section of the Management Console. For more information, see the <i>Spectrum Technology Platform Spatial Guide</i> . |
| Option.CoordinateSystem | <p>The coordinate system of the input coordinates.</p> <p>For more information on EPSG codes, see www.spatialreference.org. To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <code>http://<server>:<port>/Spatial/GeometryService/DemoPage.html</code></p> |

| Parameter | Description | | | | | | | | | | |
|---------------------------------------|--|----------------|-------------------------|-----------------------|------------------------------------|------------------------------|-------------------------|----------------|----------------------------|-----------------------|----------------------------|
| Option.CoordinateFormat | <p>Specifies the format of latitude/longitude coordinates in the input.</p> <p>Note: Use this option only if you specify a Latitude/Longitude coordinate system. If the coordinate system is not a Latitude/Longitude coordinate system, set the coordinate format to Decimal.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Decimal</td> <td>(90.000000, 180.000000)</td> </tr> <tr> <td>DecimalAssumed</td> <td>(90000000, 180000000). Default.</td> </tr> <tr> <td>DegreesMinutesSeconds</td> <td>(90 00 00N, 180 00 00W)</td> </tr> <tr> <td>PreZero</td> <td>(0900000000N, 1800000000W)</td> </tr> <tr> <td>PreZeroDecimal</td> <td>(090.000000N, 180.000000W)</td> </tr> </table> | Decimal | (90.000000, 180.000000) | DecimalAssumed | (90000000, 180000000). Default. | DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | PreZero | (0900000000N, 1800000000W) | PreZeroDecimal | (090.000000N, 180.000000W) |
| Decimal | (90.000000, 180.000000) | | | | | | | | | | |
| DecimalAssumed | (90000000, 180000000). Default. | | | | | | | | | | |
| DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | | | | | | | | | | |
| PreZero | (0900000000N, 1800000000W) | | | | | | | | | | |
| PreZeroDecimal | (090.000000N, 180.000000W) | | | | | | | | | | |
| Option.DefaultTravelBoundaryCost | Number of cost units. This can be any double value (includes decimals). The default is 10. | | | | | | | | | | |
| Option.DefaultTravelBoundaryCostUnits | <p>Type of metric you want to use to calculate the travel boundary. One of the following:</p> <ul style="list-style-type: none"> • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Milliseconds • Yards | | | | | | | | | | |

| Parameter | Description |
|----------------------------------|---|
| Option.HistoricTrafficTimeBucket | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |

Output

Table 14: GetTravelBoundary Output Options

| Parameter | Description |
|-----------------------------|--|
| Option.ResultType | <p>Specifies the type of result you want returned. One of the following:</p> <p>AccessibleNodes Returns all of the points along the road network that can be reached for the isoChrone calculation.</p> <p>Geometry Returns the entire isoChrone.</p> <p>StartNodes Returns the location specified.</p> |
| Option.SimplificationFactor | <p>Specifies what percentage of the original points should be returned or upon which the resulting polygon should be based.</p> |

| Parameter | Description |
|----------------------|---|
| Option.BandingStyle | <p>Specifies the style of banding to be used in the result. Banding styles are the types of multiple isoChrones or distance bands that can be displayed based on multiple costs.</p> <p>Donut Each boundary is determined by subtracting out the next smallest boundary.</p> <p>Encompassing Each boundary is determined independent of all others.</p> |
| Option.ReturnHoles | <p>Specifies whether you want to return holes, which are areas within the larger boundary that cannot be reached within the desired time or distance, based on the road network.</p> <p>Y Yes, return holes.</p> <p>N Do not return holes. Default.</p> |
| Option.ReturnIslands | <p>Specifies whether you want to return islands, which are small areas outside the main boundary that can be reached within the desired time or distance.</p> <p>Y Yes, return islands.</p> <p>N Do not return islands. Default.</p> |

Travel

Travel options specify assumptions to make about travel speed off network roads and whether to use only major roads when calculating the travel boundary. Most travel options have to do with ambient speed.

Table 15: GetTravelBoundary Travel Options

| Parameter | Description |
|-------------------------------|---|
| Option.MaximumOffRoadDistance | <p>Specifies the maximum distance to allow travel off the road network. Examples of off-network roads include driveways and access roads. For example, if you specify a maximum off road distance of 1 mile the travel boundary will extend no further than one mile from the network road. If you specify a value of 0 the travel boundary will not extend beyond the road itself. Use the ambient speed options to specify the speed of travel along non-network roads.</p> |

| Parameter | Description |
|--------------|--|
| Option.Units | The units of measure in which you want the data returned. One of the following: <ul style="list-style-type: none"><li data-bbox="677 459 894 487">• Kilometer (default)<li data-bbox="677 493 760 520">• Meter<li data-bbox="677 527 743 554">• Mile |

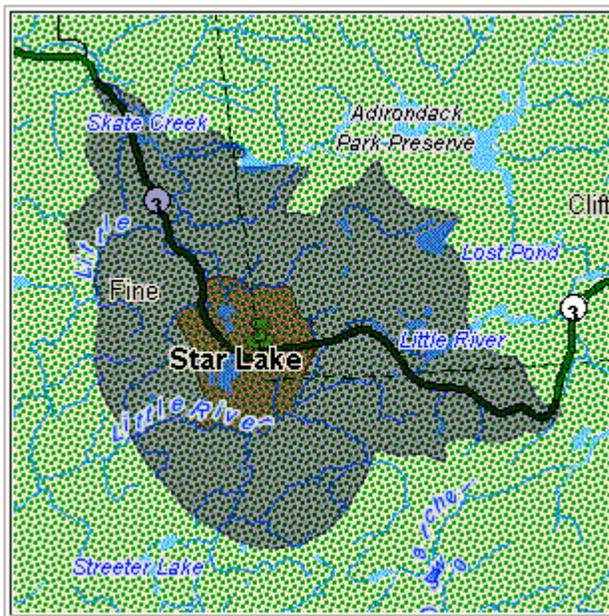
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-------------------|--|
| Option.MajorRoads | |
|-------------------|--|

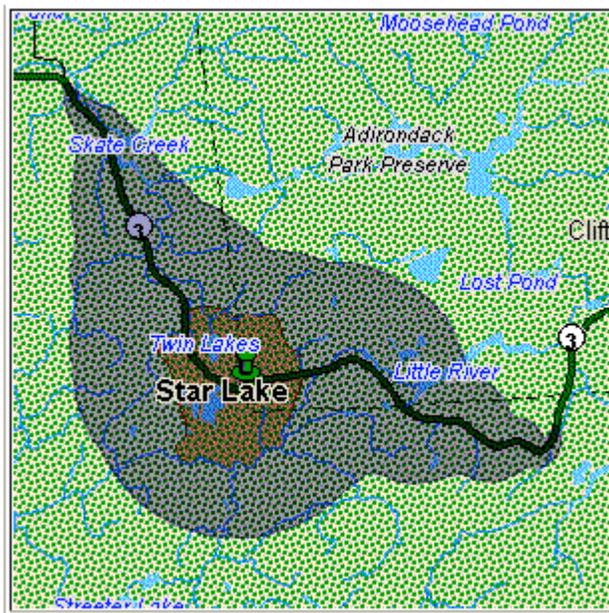
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Specifies whether to include all roads in the calculation or just major roads. By default, the Get Travel Boundary is calculated with major roads set to true. This improves performance but the accuracy may decrease.

This map represents a travel boundary with travel allowed on all roads:



This map represents a travel boundary with travel restricted to major roads only:



| Parameter | Description |
|----------------------------|--|
| | <p>One of the following:</p> <p>Y Include only major roads in the calculation. Default.</p> <p>N Include all roads in the calculation.</p> |
| Option.DefaultAmbientSpeed | <p>Specifies the speed to travel when going off a network road to find the travel boundary. Examples of off-network travel include driveways and access roads.</p> <p>This option is available only when you specify a time value in the Option.DefaultCostUnits parameter or the Data.TravelBoundaryCostUnits parameter. The default is 15. Specify the speed units in the Option.AmbientSpeedUnit parameter.</p> <p>To control how off-network travel is used in the travel boundary calculation, you need to specify the speed of travel off the road network (the ambient speed). Ambient speed can affect the size and shape of the travel boundary polygon. In general, the faster the ambient speed, the larger the polygon. For example, if you were at a point with 5 minutes left, and if the ambient speed were 15 miles per hour, boundary points would be put at a distance of 1.25 miles. If the ambient speed were reduced to 10 miles per hour, boundary points would be put at a distance of 0.83 miles. Note that you can limit the distance allowed off a network road by using the Option.MaximumOffRoadDistance parameter.</p> <p>Note: If you are calculating pedestrian travel boundaries we recommend that you change the default ambient speed to 3 MPH (5 KPH).</p> |
| Option.AmbientSpeedUnit | <p>The unit of measure to use with the value specified in the Option.DefaultAmbientSpeed parameter.</p> <p>KPH Kilometers per hour.</p> <p>MPH MILES per hour. Default.</p> <p>MTPS Meters per second.</p> <p>MTPM Meters per minute.</p> |

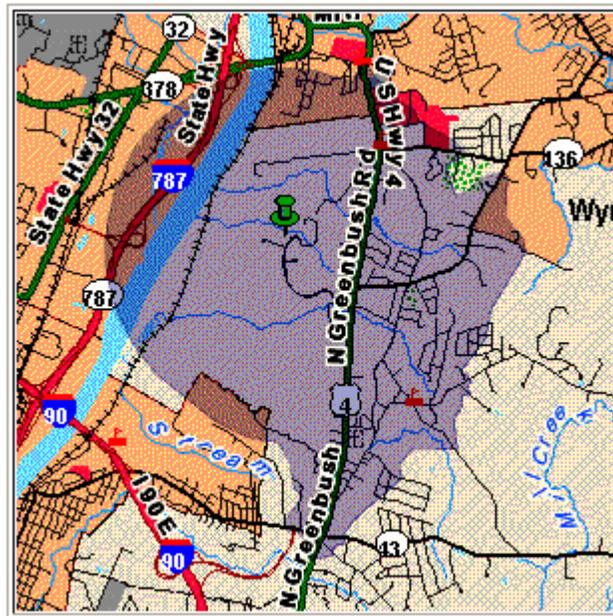
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-------------------------------------|--|
| Option.AmbientSpeed.RoadType.<Type> | |
|-------------------------------------|--|

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

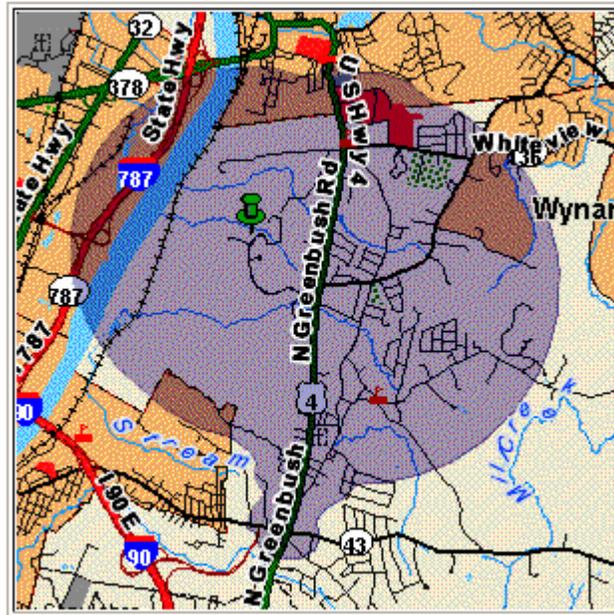
Specifies the ambient speed to use for off-network travel based on the road type. If you do not specify an ambient speed for a road type, the default ambient speed will be used, as specified in the Option.DefaultAmbientSpeed parameter.

The following map shows an example of a travel boundary without ambient speed overrides:



For comparison, this map shows the same travel boundary with ambient speed overrides:

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|



<Type> can be one of the following:

| Parameter | Description |
|-----------|--|
| | <ul style="list-style-type: none"> • AccessWay • Backroad • Connector • Ferry • Footpath • LimitedAccessDenseUrban • LimitedAccessRural • LimitedAccessSuburban • LimitedAccessUrban • LocalRoadDenseUrban • LocalRoadRural • LocalRoadSuburban • LocalRoadUrban • MajorLocalRoadDenseUrban • MajorLocalRoadRural • MajorLocalRoadSuburban • MajorLocalRoadUrban • MajorRoadDenseUrban • MajorRoadRural • MajorRoadSuburban • MajorRoadUrban • MinorLocalRoadDenseUrban • MinorLocalRoadRural • MinorLocalRoadSuburban • MinorLocalRoadUrban • NormalRoadDenseUrban • NormalRoadRural • NormalRoadRural • NormalRoadUrban • PrimaryHighwayDenseUrban • PrimaryHighwayRural • PrimaryHighwaySuburban • PrimaryHighwayUrban • RampDenseUrban • RampLimitedAccess • RampMajorRoad • RampPrimaryHighway • RampRural • RampSecondaryHighway • RampUrban • RampSuburban • SecondaryHighwayDenseUrban • SecondaryHighwayRural • SecondaryHighwaySuburban • SecondaryHighwayUrban |

service will be a success message. When a request contains invalid query parameters in the GET URL or an invalid payload for POST, a cumulative error response will be returned in a JSON array. The `value` node in the response JSON is deprecated. For error checking, only the `errors` node should be utilized.

Version specific error response

When you enter an invalid parameter value (for example, multiple updates) in a request, the error response you get depends on the version entered by you. When the version is 1, you get value and error whereas when the version is 2, the response only contains the error.

- Request when version is 1:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
velocity=15.912&velocityUnit=KPH&velocityAdjustment=34&velocityPercentage=56&version=1
```

- Response:

```
{
  "value": "One of either Velocity or SpeedIncrease or SpeedDecrease is
  expected.",
  "errors": [
    {
      "errorCode": 3733,
      "userMessage": "One of either Velocity or SpeedIncrease or
      SpeedDecrease is expected."
    }
  ]
}
```

- Request when version is 2:

```
http://<server>:<port>/rest/Spatial/erm/databases/usroutedatabase.json?
velocity=15.912&velocityUnit=KPH&velocityAdjustment=34&velocityPercentage=56&version=2
```

- Response:

```
{
  "errors": [
    {
      "errorCode": 3733,
      "userMessage": "One of either Velocity or SpeedIncrease or
      SpeedDecrease is expected."
    }
  ]
}
```

Types of Persistent Updates

See the following sections for information and examples of the persistent update request types:

Point Updates

HTTP POST URL Format

The following format is used for HTTP POST requests. HTTP POST is used to set a persistent update to a point.

```
HTTP POST :
/rest/Spatial/em/databases/dbsource/persistentUpdates.json?point=x,y,srsName&query_parameters
```

Where *dbsource* is the name of the database to update the route data. Use the database name specified in the Database Resource tool.

HTTP DELETE URL Format

The following format is used for HTTP DELETE requests. HTTP DELETE is used to remove a specific persistent update to a point.

```
HTTP DELETE :
/rest/Spatial/em/databases/dbsource/persistentUpdates.json?point=x,y,srsName&resetType=query_parameters
```

Where *dbsource* is the name of the database that contains the persistent update to remove. Use the database name specified in the Database Resource tool.

Query Parameters

The HTTP POST operation takes the following query parameters.

| Parameter | Type | Required | Description |
|-----------------|--------|----------|---|
| <i>exclude</i> | String | no | Exclude the specified point from all route calculations. The parameter's existence in the URL specifies whether to exclude, not the parameter value. |
| <i>velocity</i> | String | no | This is a speed update where you define the new speed of the point by specifying the new velocity. The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |

| Parameter | Type | Required | Description |
|---------------------------|---------|----------|--|
| <i>velocityUnit</i> | String | no | This is a unit of speed for the <i>velocity</i> or <i>velocityAdjustment</i> (miles per hour). For speed updates, the velocity unit can have one of the following values: mph (miles per hour) and kph (kilometers per hour). The default value is mph. |
| <i>velocityAdjustment</i> | String | no | This is a speed update where you define a change in the speed of the point by specifying the change in velocity (unit and value). Speed values can be increased (positive value) or decreases (negative value). The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |
| <i>velocityPercentage</i> | Integer | no | This is a speed update where you define an increase in the speed of the point by specifying a percentage to increase (positive value) or decrease (negative value) the speed. |

Reset Parameter

The HTTP DELETE operation takes the following query parameter.

| Parameter | Type | Required | Description |
|------------------|--------|----------|--|
| <i>resetType</i> | String | no | Reset (undo) a type of update for a point. <ul style="list-style-type: none"> speed Reset the speed update for a specific point. exclude Reset the exclude for a specific point. |

Examples

Exclude a point (HTTP POST)

```
http://<server>:<port>/rest/spatial/em/databases/usroute/database/persistentUpdates.json?point=-73.6,43.5,eps:4326&exclude=true
```

Remove a point exclude persistent update (HTTP DELETE)

```
http://<server>:<port>/rest/spatial/em/databases/usroute/database/persistentUpdates.json?point=-73.6,43.5,eps:4326&resetType=exclude
```

Segment Updates

HTTP POST URL Format

The following format is used for HTTP POST requests. HTTP POST is used to set a persistent update to a segment.

```
HTTP POST:
/rest/Spatial/erm/databases/dbsource/persistentUpdates/segments/segment_id.json?query_parameters
```

Where *dbsource* is the name of the database to update the route data, and *segment_id* is the identifier of the segment to update. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

HTTP GET URL Format

The following format is used for HTTP GET requests. HTTP GET is used to return a list of persistent updates for segments.

```
HTTP GET:
/rest/Spatial/erm/databases/dbsource/persistentUpdates/segments/segment_id.json
```

or

```
HTTP GET:
/rest/Spatial/erm/databases/dbsource/persistentUpdates/segments.json?segments=segment_id
```

Where *dbsource* is the name of the database to return to persistent updates from, and *segment_id* is the segment to return updates.

Note: The first format is used to return the persistent update for only one segment. The second format is used to return either multiple segments or all segments. For multiple segments, use a comma separated list of segment ids. For all segments, use an empty segments= parameter. See examples below.

HTTP DELETE URL Format

The following format is used for HTTP DELETE requests. HTTP DELETE is used to remove a specific persistent update to a segment.

```
HTTP DELETE:
/rest/Spatial/erm/databases/dbsource/persistentUpdates/segments?segmentID=segment_id&resetType=query_parameters
```

Where *dbsource* is the name of the database, and *segment_id* is the identifier of the segment to update that contains the persistent update to remove. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

Query Parameters

The HTTP POST operation takes the following query parameters.

| Parameter | Type | Required | Description |
|---------------------------|---------|----------|--|
| <i>exclude</i> | String | no | Exclude the specified segment from all route calculations. The parameter's existence in the URL specifies whether to exclude, not the parameter value. |
| <i>velocity</i> | String | no | This is a speed update where you define the new speed of the segment by specifying the new velocity. The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |
| <i>velocityUnit</i> | String | no | This is a unit of speed for the <i>velocity</i> or <i>velocityAdjustment</i> . For speed updates, the velocity unit can have one of the following values: mph (miles per hour) and kph (kilometers per hour). The default value is mph. |
| <i>velocityAdjustment</i> | String | no | This is a speed update where you define a change in the speed of the segment by specifying the change in velocity (unit and value). Speed values can be increased (positive value) or decreases (negative value). The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |
| <i>velocityPercentage</i> | Integer | no | This is a speed update where you define an increase in the speed of the segment by specifying a percentage to increase (positive value) or decrease (negative value) the speed. |

| Parameter | Type | Required | Description |
|-----------------|--------|----------|-------------|
| <i>roadType</i> | String | no | |

| Parameter | Type | Required | Description |
|-----------|------|----------|---|
| | | | <p>This is an update where you define the new road type of the segment. The road type can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban • ramp suburban • secondary highway dense urban • secondary highway rural |

| Parameter | Type | Required | Description |
|-----------|------|----------|---|
| | | | <ul style="list-style-type: none"> secondary highway suburban secondary highway urban |

Reset Parameter

The HTTP DELETE operation takes the following query parameter.

| Parameter | Type | Required | Description |
|------------------|--------|----------|---|
| <i>resetType</i> | String | Yes | Reset (undo) a type of update for a segment. <ul style="list-style-type: none"> speed Reset the speed update for a specific segment. exclude Reset the exclude for a specific segment. roadType Reset the road type for a specific segment. |

Examples

Exclude a segment (HTTP POST)

```
http://<server>:<port>/rest/Spatial/em/databases/US_NE/persistentUpdates/segments/9f5c5a5a:5174e2.json?exclude=true
```

Return a list of updates for a single segment (HTTP GET)

```
http://<server>:<port>/rest/Spatial/em/databases/US_NE/persistentUpdates/segments/efec6c1:a59ac5.json?velocityUnit=kph
```

Return a list of all segment updates for the US_NE routing database resource (HTTP GET)

```
http://<server>:<port>/rest/Spatial/em/databases/US_NE/persistentUpdates/segments.json?segments=
```

Return a list of updates for the multiple segments (HTTP GET)

```
http://<server>:<port>/rest/Spatial/em/databases/US_NE/persistentUpdates/segments.json?segments=27e20762:4718b9,7e3396fc:14c92c
```

Remove a segment speed persistent update (HTTP DELETE)

```
http://<server>:<port>/rest/Spatial/em/databases/US_NE/persistentUpdates/segments?segmentID=9f5c5a5a:5174e2&resetType=speed
```

Road Type Updates

HTTP POST URL Format

The following format is used for HTTP POST requests. HTTP POST is used to set a persistent update to a road type.

```
HTTP POST :
/rest/Spatial/erm/databases/dbsource/persistentUpdates/roadTypes/roadtype.json?query_parameters
```

Where *dbsource* is the name of the database to update the route data, and *roadtype* is the type of road to update. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

HTTP GET URL Format

The following format is used for HTTP GET requests. HTTP GET is used to return a list of persistent updates for road types.

```
HTTP GET :
/rest/Spatial/erm/databases/dbsource/persistentUpdates/roadTypes/road_type.json
```

or

```
HTTP GET :
/rest/Spatial/erm/databases/dbsource/persistentUpdates/roadTypes.json?roadTypes=road_type
```

Where *dbsource* is the name of the database to return to persistent updates from, and *roadtype* is the type of road return updates.

Note: The first format is used to return the persistent update for only one road type. The second format is used to return either multiple road types or all road types. For multiple road types, use a comma separated list of road types. For all road types, use an empty *roadtypes=* parameter. See examples below.

HTTP DELETE URL Format

The following format is used for HTTP DELETE requests. HTTP DELETE is used to remove a specific persistent update to a road type.

```
HTTP DELETE :
/rest/Spatial/erm/databases/dbsource/persistentUpdates/roadTypes/roadtype
```

Where *dbsource* is the name of the database, and *roadtype* is the type of road that contains the persistent update to remove. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

The *roadtype* can be one of the following for both the HTTP POST and HTTP DELETE:

- access way
- back road
- connector
- ferry
- footpath
- limited access dense urban
- limited access rural
- limited access suburban
- limited access urban
- local road dense urban
- local road rural
- local road suburban
- local road urban
- major local road dense urban
- major local road rural
- major local road suburban
- major local road urban
- major road dense urban
- major road rural
- major road suburban
- major road urban
- minor local road dense Urban
- minor local road rural
- minor local road suburban
- minor local road urban
- normal road dense urban
- normal road rural
- normal road rural
- normal road urban
- primary highway dense urban
- primary highway rural
- primary highway suburban
- primary highway urban
- ramp dense urban
- ramp limited access
- ramp major road
- ramp primary highway
- ramp rural
- ramp secondary highway
- ramp urban

- ramp suburban
- secondary highway dense urban
- secondary highway rural
- secondary highway suburban
- secondary highway urban

Query Parameters

The HTTP POST operation takes the following query parameters.

| Parameter | Type | Required | Description |
|---------------------------|---------|----------|--|
| <i>velocity</i> | String | no | This is a speed update where you define the new speed of the road type by specifying the new velocity. The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |
| <i>velocityUnit</i> | String | no | This is a unit of speed for the <i>velocity</i> or <i>velocityAdjustment</i> . For speed updates, the velocity unit can have one of the following values: mph (miles per hour) and kph (kilometers per hour). The default value is mph. |
| <i>velocityAdjustment</i> | String | no | This is a speed update where you define a change in the speed of the road type by specifying the change in velocity (unit and value). Speed values can be increased (positive value) or decreases (negative value). The default unit is mph (miles per hour) unless you specify the <i>velocityUnit</i> parameter. |
| <i>velocityPercentage</i> | Integer | no | This is a speed update where you define an increase in the speed of the road type by specifying a percentage to increase (positive value) or decrease (negative value) the speed. |

Examples

Set a new velocity of a road type (HTTP POST)

```
http://<server>:<port>/rest/spatial/em/databases/usroute/database/persistentupdates/roadTypes/ferry.json?velocity=5&velocityUnit=mph
```

Return a list of updates for the ferry road type (HTTP GET)

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates/roadTypes/ferry.json?velocityUnit=kph
```

Return a list of all road type updates for the US_NE routing database resource (HTTP GET)

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates/roadTypes.json?roadTypes=
```

Return a list of updates for the ferry, connector, and normal road urban road types (HTTP GET)

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates/roadTypes.json?roadTypes=ferry,connector,normal  
road urban
```

Remove a road type persistent update (HTTP DELETE)

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates/roadTypes/back  
road
```

Remove All Updates

HTTP DELETE URL Format

The following format is used for HTTP DELETE requests. HTTP DELETE is used to remove all persistent update for a specified database.

```
HTTP DELETE: /rest/Spatial/erm/databases/dbsource/persistentUpdates
```

Where *dbsource* is the name of the database that contains the persistent updates to remove. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

Example

Removes all persistent updates for the US_NE routing database resource.

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates
```

Get All Updates

HTTP GET URL Format

The following format is used for HTTP GET requests. This HTTP GET operation is used to list all the persistent updates for a specified routing database resource.

```
HTTP GET: /rest/Spatial/erm/databases/dbsource/persistentUpdates.json
```

Where *dbsource* is the name of the database that contains the persistent updates to remove. Use the database name specified in the Spectrum Spatial Routing Database Resource tool.

Query Parameters

This operation takes the following query parameter.

| Parameter | Type | Required | Description |
|---------------------|--------|----------|---|
| <i>velocityUnit</i> | String | no | The updates saved in the server will be returned in this specified unit. If this parameter is not mentioned, response will be returned in default unit. For speed updates, the velocity unit can have one of the following values: mph (miles per hour), kph (kilometers per hour), mtps (meters per second), and mtpm (meters per minute). The default value is mph. |

Example

Return a list of updates for the US_NE routing database resource.

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates.json
```

Response

```
{
  "roadTypeUpdates":
  [
    {
      "roadType": "major road dense urban",
      "speed":
      {
        "velocity": 90,
        "velocityUnit": "MPH"
      }
    }
  ],
  "segmentUpdates":
  [
    {
      "exclude": true,
      "roadType": "major road dense urban",
      "segmentID": "c75994cc:12d916",
      "speed":
      {
        "velocity": 65,
```

```

        "velocityUnit": "MPH"
    },
    {
        "exclude": true,
        "roadType": "major road dense urban",
        "segmentID": "7ac5401f:6b1bf7",
        "speed": {
            "velocity": 65,
            "velocityUnit": "MPH"
        }
    }
]
}

```

When velocity unit parameter is specified in kph.

```
http://<server>:<port>/rest/Spatial/erm/databases/<database_name>/persistentUpdates.json?velocityUnit=kph
```

Response

```

{
  "roadTypeUpdates": [{
    "roadType": "major road dense urban",
    "speed": {
      "velocity": 145,
      "velocityUnit": "KPH"
    }
  }]
}

```

GetCapabilities

Description

The `GetCapabilities` service enables user to retrieve the metadata about the deployed routing engine. The user can use the metadata to explore a service and its capabilities, therefore optimizing their experience using the routing services.

This is available as REST service only.

HTTP GET URL Format

The following format is used for HTTP GET requests.

```
http://<server>:<port>/rest/Spatial/erm/v1/capabilities.json
```

Query Parameters

The table describes the query parameters for `GetCapabilities`.

| Parameter | Required | Description |
|-----------------------------|---|--|
| <code>acceptVersions</code> | Optional | Specifies a placeholder (not functional) |
| <code>sections</code> | Optional When omitted, returns information about all sections. | Specifies a comma-separated unordered list of zero or more names of sections of service metadata document to be returned in the service metadata document. Section values are case-insensitive. Accepted section values are: <ul style="list-style-type: none"> • <code>ServiceIdentification</code> • <code>ServiceProvider</code> • <code>operationsMetadata</code> • <code>databases</code> |

Response

The response will be in line with OGC `GetCapabilities`. It is in JSON format and has these sections:

- `serviceIdentification`
- `serviceProvider`
- `operationsMetadata`
- `databases`

serviceIdentification

This section contains basic metadata about this specific server. Its content look like:

```
"serviceIdentification":
  {
    "title": "PBS Routing Service",
    "abstract": "Routing service maintained by PBS",
    "keywords":
      {
        "keyword":
          [
            ]
      },
    "serviceType": "Routing",
    "serviceTypeVersion": "v1",
    "fees": "none",
```

```

    "accessConstraints": "none"
  }

```

This information will be the same as what is available in the `getCapabilities.json` configuration file.

This file is available at `SpectrumDirectory\server\modules\routing`. The server restart is required for any changes to be effective. The administrator can allow what information the user is allowed to view, modify, or delete in the corresponding entries in the JSON file. All fields in the JSON file are optional.

serviceProvider

This section contains metadata about the organization operating this server. Its content look like:

```

"serviceProvider":
  {
    "providerName": "Routing Service Provider",
    "providerSite":
      {
        "href": "http://www.yourcompany.com/",
        "type": "simple"
      },
    "serviceContact":
      {
        "contactInfo":
          {
            "address":
              {
                "administrativeArea": "Province",
                "city": "City",
                "country": "Country",
                "deliveryPoint": "Mail Delivery Location",
                "electronicMailaddress": "mailto://support@yourcompany.com",
                "postalCode": "PostCode"
              },
            "contactInstructions": "Contact Instructions",
            "hoursOfservice": "24 Hours",
            "phone":
              {
                "facsimile": "1.800.000.0000",
                "voice": "1.800.000.0000"
              }
          },
        "individualName": "Contact Person",
        "positionName": "Contact Person's Title",
        "role": "Contact Person's Role"
      }
  }

```

This will also be configured using the `getCapabilities.json` configuration file as described above.

operationsMetadata

This section contains metadata about the operations implemented by this server, including the URLs for operation requests. These fixed operations or services are listed in this section:

- **GetRoute**: point to point service
- **GetRouteCostMatrix**: matrix of points processing service
- **GetTravelBoundary** : generates a drive or walk time or distance boundary
- **DescribeDatasets**: gives information about the datasets configured
- **DescribeDatabases**: gives information about all the databases configured
- **GetSegmentDataForPoint**: returns segment information for a point
- **GetSegmentDataForSegment**: returns segment information for a segment ID
- **ListPersistentUpdates**: lists down all the persistent updates that exists in the server
- **DeletePersistentUpdates**: deletes all the persistent updates that exists in the server
- **SetPersistentUpdatesAtPoint**: saves persistent update for the specified point in the server
- **SetPersistentUpdatesForSegment**: saves persistent update for the specified segment ID in the server
- **SetPersistentUpdatesForRoadType**: saves persistent update for the specified road type in the server

Its content look like:

```
{
  "operationsMetadata": [
    {
      "name": "GetRoute",
      "DCP": {
        "HTTP": {
          "GET":
            "<schema>://<server>:<port>/rest/Spatial/em/databases/<DB_NAME>.json?q=route&version=2",
          "POST":
            "<schema>://<server>:<port>/rest/Spatial/em/databases/<DB_NAME>.json?q=route&version=2"
        }
      },
      "parameter": {
        "name": "OutputFormat",
        "value": "text/json"
      }
    },
    {
      "name": "GetRouteCostMatrix",
      "DCP": {
        "HTTP": {
          "GET":
            "<schema>://<server>:<port>/rest/Spatial/em/databases/<DB_NAME>.json?q=routeCostMatrix&version=2",

```

```

    "POST":
    "<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=routeCostMatrix&version=2"
  }
}, "parameter": {
  "name": "OutputFormat",
  "value": "text/json"
},
{
  "name": "GetTravelBoundary",
  "DCP": {
    "HTTP": {
      "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=travelBoundary&version=2",
      "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=travelBoundary&version=2"
    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},
{
  "name": "DescribeDatasets",
  "DCP": {
    "HTTP": {
      "GET": "<schema>://<server>:<port>/rest/Spatial/erm/v1/datasets.json"
    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},
{
  "name": "DescribeDatabases",
  "DCP": {
    "HTTP": {
      "GET": "<schema>://<server>:<port>/rest/Spatial/erm/v1/databases.json"
    }
  },
  "parameter": {
    "name": "OutputFormat",

```

```

    "value": "text/json"
  }
},

{
  "name": "GetSegmentDataForPoint",
  "DCP": {
    "HTTP": {
      "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/segments.json"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "GetSegmentDataForSegment",
  "DCP": {
    "HTTP": {
      "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/segments/<segmentID>.json"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "ListPersistentUpdates",
  "DCP": {
    "HTTP": {
      "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/persistentUpdates.json"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "DeletePersistentUpdates",
  "DCP": {

```

```

    "HTTP": {
      "DELETE":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/persistentUpdates"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "SetPersistentUpdatesAtPoint",
  "DCP": {
    "HTTP": {
      "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/persistentUpdates.json"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "SetPersistentUpdatesForSegment",
  "DCP": {
    "HTTP": {
      "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/persistentUpdates/segments/<segmentID>.json"

    }
  },
  "parameter": {
    "name": "OutputFormat",
    "value": "text/json"
  }
},

{
  "name": "SetPersistentUpdatesForRoadType",
  "DCP": {
    "HTTP": {
      "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>/persistentUpdates/roadTypes/<roadtype>.json"

    }
  },
  "parameter": {

```

```

    "name": "OutputFormat",
    "value": "text/json"
  }
}
]]}

```

databases

This section will contain the list of names of databases which are configured in the server.

For example:

```

{
  "databases":
  [
    "US_NE",
    "US"
  ]
}

```

If no database is configured on the server, this is returned:

```

{
  "databases": [
  ]
}

```

DescribeDatasets

Description

The `DescribeDatasets` service enables user to get information metadata about the datasets corresponding to the routing databases added to the Spectrum Technology Platform server. The response will be analogous with the metadata information present in the dataset path.

This feature is available as REST service only.

HTTP GET URL Format

All Datasets: The following format is used for HTTP GET requests for all datasets.

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets.json
```

Single Dataset: The following format is used for HTTP GET requests for a single dataset.

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets/<dataset_ID>.json
```

The `dataset_ID` is the 'id' corresponding to the elements in the 'dataSets' array from the `DescribeDatabases` service.

Response

The response of this service is a JSON array.

For all datasets, the length of the JSON array is the same as the total number of the dataset paths (with metadata available) added against the databases configured in Management Console. If a dataset path does not have metadata available, that entry will be ignored.

For a single dataset, the length of the JSON array will be one if and only if the metadata is available in the dataset path. Else, an empty JSON array will be returned.

Example

Two routing databases have been added in the Spectrum Technology Platform. The name and dataset paths of the databases are as follows:

1. **US_NE:** E:\\db\\ERM-US\\2014.09\\driving\\northeast
2. **US:** E:\\db\\ERM-US\\2014.09\\driving\\midwest and
E:\\db\\ERM-US\\2014.09\\driving\\south

Sample Request (All Datasets):

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets.json
```

Response:

```
{
  "dataSets": [{
    "component": "routing",
    "description": "USA Test dataset",
    "ext": {
      "bbox": [68.291015625, 7.9721977144, 97.55859375, 35.4606699515],

      "crs": "epsg:4326",
      "cvr": true,
      "historicTrafficTimeBuckets": {
        "amPeak": {
          "lowerBound": 700,
          "upperBound": 1000
        },
        "nightTime": {
          "lowerBound": 2200,
          "upperBound": 400
        },
        "offPeak": {
          "lowerBound": 1000,
          "upperBound": 1600
        },
        "pmPeak": {
```

```

        "lowerBound": 1600,
        "upperBound": 1900
    }
},
"locale": "EN",
"type": "driving"
},
"id": "US dataset",
"name": "USA",
"product": "Spatial",
"vintage": "September 2015"
}]
}

```

Sample Request (Single Dataset):

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets/US%20dataset.json
```

Response:

```

{
  "dataSets": [{
    "component": "routing",
    "description": "USA Test dataset",
    "ext": {
      "bbox": [68.291015625, 7.9721977144, 97.55859375, 35.4606699515],

      "crs": "epsg:4326",
      "cvr": true,
      "historicTrafficTimeBuckets": {
        "amPeak": {
          "lowerBound": 700,
          "upperBound": 1000
        },
        "nightTime": {
          "lowerBound": 2200,
          "upperBound": 400
        },
        "offPeak": {
          "lowerBound": 1000,
          "upperBound": 1600
        },
        "pmPeak": {
          "lowerBound": 1600,
          "upperBound": 1900
        }
      },
      "locale": "EN",
      "type": "driving"
    },
    "id": "US dataset",
  }
]
}

```

```

    "name": "USA",
    "product": "Spatial",
    "vintage": "September 2015"
  }]
}

```

DescribeDatabases

Description

The `DescribeDatabases` operation returns name of all the database resources that are configured in the system and can be used in a request. This operation returns a list containing the names of all databases in the system and an array containing the datasets for each database.

HTTP GET URL Format (All Databases)

The format below is used for HTTP GET requests. If no data resource exists on the server, an empty list is returned.

```
http://<server>:<port>/rest/Spatial/erm/v1/databases.json
```

Example (All Databases)

Request:

```
http://<server>:<port>/rest/Spatial/erm/v1/databases.json
```

Response:

```

{
  "databases":
  [
    {
      "dataSets":
      [
        "US_Central"
      ],
      "name": "US_CN"
    },
    {
      "dataSets":
      [
        "US_NorthEast"
      ],
      "name": "US_NE"
    },
    {

```

```

    "dataSets":
    [
      "US_Central",
      "US_Midwest",
      "US_NorthEast",
      "US_Pacific",
      "US_South"
    ],
    "name": "US"
  }
]
}

```

HTTP GET URL Format (Single Database)

The format below is used for HTTP GET requests. This request is used if to get the dataset information for a particular data resource. If no data resource with the specified name exists on the server, an exception is returned.

```
http://<server>:<port>/rest/Spatial/erm/v1/<database_name>.json
```

Example (Single Database)

Request:

```
http://<server>:<port>/rest/Spatial/erm/v1/databases/US.json
```

Response:

```

{
  "databases":
  [
    {
      "dataSets":
      [
        "US_Central",
        "US_Midwest",
        "US_NorthEast",
        "US_Pacific",
        "US_South"
      ],
      "name": "US"
    }
  ]
}

```

Spectrum Universal Address

AutoCompleteLoqate

AutoCompleteLoqate offers real-time entry of address data for fast, accurate results. Users are returned instant results based on each character entered into the form, ensuring only accurate data is entered into the database. AutoCompleteLoqate also includes the Powersearch option, which reduces input time by up to 80% for 238 countries by using data in the form of an index file.

Resource URL

JSON endpoint:

```
http://server:port/rest/AutoCompleteLoqate/results.json
```

XML endpoint:

```
http://server:port/rest/AutoCompleteLoqate/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/AutoCompleteLoqate/results.json?Data.AddressLine1=1+Global
```

The JSON returned by this request would be:

Note: To make the example easier to read, empty response elements have been removed and only the first three address matches are shown.

```
{ "output_port": [
  {
    "ProcessedBy": "LOQATE",
    "HouseNumber": "1",
    "AddressLine1": "1 Global Vw",
    "FirmName": "Map Info",
    "City": "Troy",
    "StateProvince": "NY",
    "PostalCode": "12180-8399",
    "Country": "United States",
    "PostalCode.AddOn": "8399",
    "user_fields": []
  },
  {
```

```

    "ProcessedBy": "LOQATE",
    "HouseNumber": "1",
    "AddressLine1": "1 Global Pl",
    "City": "Glendale",
    "StateProvince": "AZ",
    "PostalCode": "85306-3216",
    "Country": "United States",
    "PostalCode.AddOn": "3216",
    "user_fields": []
  },
  {
    "ProcessedBy": "LOQATE",
    "HouseNumber": "1",
    "AddressLine1": "1 Global Dr",
    "City": "Olive Hill",
    "StateProvince": "KY",
    "PostalCode": "41164-6739",
    "Country": "United States",
    "PostalCode.AddOn": "6739",
    "user_fields": []
  }
]

```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/AutoCompleteLoqate/results.xml?Data.AddressLine1=1+Global
```

The XML returned by this request would be:

Note: To make the example easier to read, empty response elements have been removed and only the first three address matches are shown.

```

<ns2:xml.AutoCompleteLoqateResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/AutoCompleteLoqate">

  <ns2:output_port>
    <ns2:Address>
      <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
      <ns2:HouseNumber>1</ns2:HouseNumber>
      <ns2:AddressLine1>1 Global Vw</ns2:AddressLine1>
      <ns2:FirmName>Map Info</ns2:FirmName>
      <ns2:City>Troy</ns2:City>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:PostalCode>12180-8399</ns2:PostalCode>
      <ns2:PostalCode.AddOn>8399</ns2:PostalCode.AddOn>
      <ns2:Country>United States</ns2:Country>
    </ns2:Address>
    <ns2:Address>

```

```

    <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
    <ns2:HouseNumber>1</ns2:HouseNumber>
    <ns2:AddressLine1>1 Global Pl</ns2:AddressLine1>
    <ns2:City>Glendale</ns2:City>
    <ns2:StateProvince>AZ</ns2:StateProvince>
    <ns2:PostalCode>85306-3216</ns2:PostalCode>
    <ns2:PostalCode.AddOn>3216</ns2:PostalCode.AddOn>
    <ns2:Country>United States</ns2:Country>
  </ns2:Address>
  <ns2:Address>
    <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
    <ns2:HouseNumber>1</ns2:HouseNumber>
    <ns2:AddressLine1>1 Global Dr</ns2:AddressLine1>
    <ns2:City>Olive Hill</ns2:City>
    <ns2:StateProvince>KY</ns2:StateProvince>
    <ns2:PostalCode>41164-6739</ns2:PostalCode>
    <ns2:PostalCode.AddOn>6739</ns2:PostalCode.AddOn>
    <ns2:Country>United States</ns2:Country>
  </ns2:Address>
</ns2:output_port>
</ns2:xml.AutoCompleteLoqateResponse>

```

Request

Parameters for Input Data

The following table lists the input for AutoCompleteLoqate.

Table 17: Input Format

| Parameter | Description |
|-------------------|--------------------------|
| Data.AddressLine1 | The first address line. |
| Data.AddressLine2 | The second address line. |
| Data.AddressLine3 | The third address line. |
| Data.AddressLine4 | The fourth address line. |
| Data.City | The city name. |

| Parameter | Description |
|--------------------|---|
| Data.Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| Data.FirmName | The company or firm name. |
| Data.PostalCode | The postal code for the address. |
| Data.StateProvince | The state or province. |

Parameters for Options

Table 18: AutoCompleteLoqate Options

| Parameter | Description |
|------------------------|---|
| Option.Database.Loqate | Specifies the database to be used for address processing. Only databases that have been defined in the Database Resources panel in the Management Console are available. |
| Option.OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p>123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p>123 MAIN ST MYTOWN FL 12345</p> |

| Parameter | Description |
|----------------------------|---|
| Option.HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent and the Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |
| Option.OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |

| Parameter | Description |
|----------------------------|--|
| Option.OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |
| Option.MaximumResults | <p>The maximum number of addresses that AutoCompleteLoqate should return. The default is 10.</p> |
| Option.isPowersearchEnable | <p>Reduces input time by up to 80% for 240 countries by using data in the form of an index file. When you conduct a search, the Loqate Engine will first look for the corresponding index. If present, the method will attempt to instantly return a list of candidate addresses. If the index is not present, or if the index does not return any results, the original search process will be triggered.</p> <p>Note: Powersearch can be performed when there are two and only two fields in the input file: the Country field and any one of the AddressLine fields. If you select this option and your input file contains additional fields, the original search process will automatically be triggered.</p> <p>To conduct its search, Auto Complete indexes use up to the first 10 characters for searches within the United States and up to the first 15 characters for searches within all other eligible countries. Spaces and punctuation are not factored into this count.</p> <p>Powersearch cannot be used for the following countries: Botswana, Ethiopia, India, Kazakhstan, Malaysia, Mongolia, Saint Kitts and Nevis, and San Marino.</p> <p>Note: You must have a valid license for Powersearch processing. If you select this option but are not licensed for Powersearch, or if your license has expired, you will receive an error.</p> |

| Parameter | Description |
|--------------------------------------|--|
| Option.IsDuplicateHandlingMaskEnable | <p>Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options:</p> <p>S Selected by default. Pre-process the input and remove duplicates that occur in a single field.</p> <p>C Selected by default. Preprocess the input and remove duplicates across all fields.</p> <p>T Preprocess the input and remove duplicates in fields that are not standard address fields.</p> <p>F Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.</p> |
| Option.FailJobOnDataLicenseError | <p>Specifies how you want Spectrum Technology Platform to respond when a data license error occurs.</p> <p>Fail the job Fail the entire job if a data license error occurs.</p> <p>Fail the record Fail the record(s) for which the data license error occurs and continue processing.</p> |

Response

The output from AutoCompleteLoqate is optional and corresponds directly to the fields you selected in the Output Fields section of the AutoCompleteLoqate Options dialog box.

Table 19: AutoCompleteLoqate Output

| Response Element | Description |
|------------------|--------------------------|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |

| Response Element | Description |
|------------------|--|
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| FirmName | The firm name. |
| HouseNumber | The ending house number for the range in which the candidate address's house number falls. |
| PostalCode | The postal code. |
| PostalCode.AddOn | The last four digits of the ZIP + 4 [®] Code. |
| ProcessedBy | Indicates which address coder processed the address. LOQATE The Loqate coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. <ul style="list-style-type: none"> • DisabledCoder • RequestFailed • NoLookupAddressFound |

| Response Element | Description |
|--------------------|---|
| Status.Description | A description of the problem, if there is one. |
| | <p>Did not return multiples The input address matched only one address in the database. AutoCompleteLoqate returns data only if multiple possible matches were found.</p> <p>Not able to look up the address pattern AutoCompleteLoqate is not able to process the partial address.</p> |

AutoCompleteLoqate Sample Web Application

You can access a sample web application that demonstrates the Auto Complete Loqate functionality. When you enter a partial address, this application makes a call to the Auto Complete Loqate REST web service, which returns a suggested address.

Note: Prior to using this feature, you must add an Auto Complete Loqate database resource in Management Console and save the database resource in the Auto Complete Loqate Service.

1. Be sure the Spectrum Technology Platform server is running.
2. Open a web browser and go to: `http://<servername>:<port>/autocomplete`. For example, if your server is named "myserver" and it uses the default HTTP port 8080, you would go to: `http://myserver:8080/autocomplete`.

Note: This site is best viewed in Internet Explorer 8.0 or later, Chrome, or Mozilla Firefox.

3. When the login screen appears, enter "**guest**" as the user name and leave the password field blank.
4. Press **OK**.
5. Select a country from the drop-down list.
6. Begin typing your address in any of the fields provided.
7. Select from the list of suggested addresses.
8. To begin a new call, click **Reset**, which will clear the fields you used in your previous call.

GetCandidateAddresses

GetCandidateAddresses returns a list of addresses that are considered matches for a given input address. GetCandidateAddresses returns candidate addresses only if the input address matches multiple addresses in the postal database. If the input address matches only one address in the postal database, then no address data is returned.

For addresses outside the U.S. and Canada, you may notice inconsistent results between the multiple matches returned by ValidateAddress and the results for that same address returned by

GetCandidateAddresses. If you experience inconsistent results, it is likely because you set the performance tuning setting in ValidateAddress to a value other than 100. To obtain consistent results between GetCandidateAddresses and ValidateAddress, set the performance tuning option to 100.

Note: By default, GetCandidateAddresses does not match to individual house numbers. Rather, it uses house number ranges for each street. After GetCandidateAddresses has determined the street name, city name, state/province name, and postal code, it checks to make sure the input house number falls within one of the ranges of house numbers given for the matched street name. The same type of logic applies to unit numbers. If you want to determine that an individual house number is valid, you should use the ValidateAddress Delivery Point Validation (DPV) processing option. DPV processing is only available for U.S. addresses.

The Canadian coder contains a reverse lookup routine that takes as input a specific postal code and returns the street information stored in the database for that postal code. To use this function enter nothing but a Canadian postal code in the PostalCode field. See the second example to view the return from a sample postal code.

GetCandidateAddresses is part of Spectrum Universal Address.

Resource URL

JSON endpoint:

```
http://server:port/rest/GetCandidateAddresses/results.json
```

XML endpoint:

```
http://server:port/rest/GetCandidateAddresses/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/GetCandidateAddresses/results.json?
Data.AddressLine1=P.O.+Box+1&Data.City=New+York&Data.StateProvince=NY
```

The JSON returned by this request would be:

```
{ "output_port": [
  {
    "ProcessedBy": "USA",
    "RecordType": "PostOfficeBox",
    "MatchLevel": "A",
    "AddressLine1": "PO Box 1",
    "HouseNumberLow": "1",
    "HouseNumberHigh": "60",
    "HouseNumberParity": "B",
```

```

    "UnitNumberLow": "",
    "UnitNumberHigh": "",
    "UnitNumberParity": " ",
    "FirmName": "",
    "City": "New York",
    "USUrbanName": "",
    "StateProvince": "NY",
    "PostalCode": "10002",
    "Country": "USA",
    "PostalCode.AddOn": "0001",
    "user_fields": []
  },
  {
    "ProcessedBy": "USA",
    "RecordType": "PostOfficeBox",
    "MatchLevel": "A",
    "AddressLine1": "PO Box 1",
    "HouseNumberLow": "1",
    "HouseNumberHigh": "9",
    "HouseNumberParity": "B",
    "UnitNumberLow": "",
    "UnitNumberHigh": "",
    "UnitNumberParity": " ",
    "FirmName": "",
    "City": "New York",
    "USUrbanName": "",
    "StateProvince": "NY",
    "PostalCode": "10008",
    "Country": "USA",
    "PostalCode.AddOn": "0001",
    "user_fields": []
  },
  {
    "ProcessedBy": "USA",
    "RecordType": "PostOfficeBox",
    "MatchLevel": "A",
    "AddressLine1": "PO Box 1",
    "HouseNumberLow": "1",
    "HouseNumberHigh": "60",
    "HouseNumberParity": "B",
    "UnitNumberLow": "",
    "UnitNumberHigh": "",
    "UnitNumberParity": " ",
    "FirmName": "",
    "City": "New York",
    "USUrbanName": "",
    "StateProvince": "NY",
    "PostalCode": "10009",
    "Country": "USA",
    "PostalCode.AddOn": "0001",
    "user_fields": []
  }
]}

```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/GetCandidateAddresses/results.xml?
Data.AddressLine1=P.O.+Box+1&Data.City=New+York&Data.StateProvince=NY
```

The XML returned by this request would be:

```
<ns2:xml.GetCandidateAddressesResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GetCandidateAddresses">

  <ns2:output_port>
    <ns2:Address>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:RecordType>PostOfficeBox</ns2:RecordType>
      <ns2:MatchLevel>A</ns2:MatchLevel>
      <ns2:AddressLine1>PO Box 1</ns2:AddressLine1>
      <ns2:HouseNumberLow>1</ns2:HouseNumberLow>
      <ns2:HouseNumberHigh>60</ns2:HouseNumberHigh>
      <ns2:HouseNumberParity>B</ns2:HouseNumberParity>
      <ns2:UnitNumberLow/>
      <ns2:UnitNumberHigh/>
      <ns2:UnitNumberParity></ns2:UnitNumberParity>
      <ns2:FirmName/>
      <ns2:City>New York</ns2:City>
      <ns2:USUrbanName/>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:PostalCode>10002</ns2:PostalCode>
      <ns2:PostalCode.AddOn>0001</ns2:PostalCode.AddOn>
      <ns2:Country>USA</ns2:Country>
      <ns2:user_fields/>
    </ns2:Address>
    <ns2:Address>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:RecordType>PostOfficeBox</ns2:RecordType>
      <ns2:MatchLevel>A</ns2:MatchLevel>
      <ns2:AddressLine1>PO Box 1</ns2:AddressLine1>
      <ns2:HouseNumberLow>1</ns2:HouseNumberLow>
      <ns2:HouseNumberHigh>9</ns2:HouseNumberHigh>
      <ns2:HouseNumberParity>B</ns2:HouseNumberParity>
      <ns2:UnitNumberLow/>
      <ns2:UnitNumberHigh/>
      <ns2:UnitNumberParity></ns2:UnitNumberParity>
      <ns2:FirmName/>
      <ns2:City>New York</ns2:City>
      <ns2:USUrbanName/>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:PostalCode>10008</ns2:PostalCode>
      <ns2:PostalCode.AddOn>0001</ns2:PostalCode.AddOn>
      <ns2:Country>USA</ns2:Country>
      <ns2:user_fields/>
    </ns2:Address>
  </ns2:output_port>
</ns2:xml.GetCandidateAddressesResponse>
```

```

    </ns2:Address>
  </ns2:output_port>
</ns2:xml.GetCandidateAddressesResponse>

```

Request

Parameters for Input Data

The following table lists the input for GetCandidateAddresses.

Table 20: Input Format

| Parameter | Description |
|--------------------|--|
| Data.AddressLine1 | The first address line. |
| Data.AddressLine2 | The second address line. |
| Data.AddressLine3 | The third address line. Does not apply to U.S. and Canadian addresses. |
| Data.AddressLine4 | The fourth address line. Does not apply to U.S. and Canadian addresses. |
| Data.AddressLine5 | The fifth address line. Applies only to U.K. addresses. May contain street name, unit number, building number, and so on. |
| Data.City | The city name. |
| Data.StateProvince | The state or province. For U.S. addresses only, you may put the state in the City field instead of the StateProvince field. |

| Parameter | Description |
|------------------|--|
| Data.PostalCode | <p>The postal code for the address. For U.S. addresses this is the ZIP Code™ in one of the following formats:</p> <p>99999 99999-9999 A9A9A9 A9A 9A9 9999 999</p> <p>Note: For Canadian addresses you can complete just this field and have candidate address data returned. For other countries, AddressLine1 and AddressLine2 must also be completed.</p> |
| Data.Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none">• 2-digit ISO country code• 3-digit UPU Country code• English country name• French country name• German country name• Spanish country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| Data.FirmName | The company or firm name. |
| Data.USUrbanName | U.S. address urbanization name. Used primarily for Puerto Rico addresses. |

Parameters for Options

Table 21: GetCandidateAddresses Options

| Parameter | Description |
|----------------------------------|---|
| Option.PerformUSProcessing | <p>Specifies whether or not to process U.S. addresses. If you enable U.S. address processing GetCandidateAddresses will attempt to retrieve candidate addresses for U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process U.S. addresses (default).</p> <p>N No, do not process U.S. addresses.</p> |
| Option.Database.US | <p>Specifies the database to be used for U.S. address processing. Only databases that have been defined in the US Database Resources panel in the Management Console are available.</p> |
| Option.PerformCanadianProcessing | <p>Specifies whether or not to process Canadian addresses. If you enable Canadian address processing GetCandidateAddresses will attempt to retrieve candidate addresses for Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process Canadian addresses (default).</p> <p>N No, do not process Canadian addresses.</p> |

| Parameter | Description |
|---------------------------------------|--|
| Option.Database.Canada | Specifies the database to be used for Canadian address processing. Only databases that have been defined in the Canadian Database Resources panel in the Management Console are available. |
| Option.PerformInternationalProcessing | <p>Specifies whether or not to process international addresses (addresses outside the U.S. and Canada). If you enable international address processing GetCandidateAddresses will attempt to retrieve candidate addresses for international addresses. If you disable international address processing, international addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for international address processing you must disable international address processing in order for your jobs to complete successfully, regardless of whether or not they contain international addresses.</p> <p>Note: You must have a valid license for international address processing to successfully process international addresses. If you enable international address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process international addresses (default).</p> <p>N No, do not process international addresses.</p> |
| Option.Database.International | Specifies the database to be used for international address processing. Only databases that have been defined in the International Database Resources panel in the Management Console are available. |
| Option.OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p>123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p>123 MAIN ST MYTOWN FL 12345</p> |
| Option.MaximumResults | The maximum number of candidate addresses that GetCandidateAddresses should return. The default is 10. The maximum is 10. |

| Parameter | Description |
|---------------------------------|--|
| Option.OutputShortCityName | <p>For U.S. addresses, specifies whether or not to return the USPS®-approved abbreviation for the city, if there is one. The USPS® provides abbreviations for city names that are 14 characters long or longer. City abbreviations are 13 characters or less and can be used when there is limited space on the mailing label. If there is no short city name for the city, then the full city name is returned.</p> <p>Y Yes, return the short city name.</p> <p>N No, do not return the short city name.</p> |
| Option.DualAddressLogic | <p>(U.S. addresses only). Controls whether GetCandidateAddresses should return a street match or a PO Box/Rural Route/Highway Contract match when the address contains both street and PO Box/Rural Route/Highway Contract information. For more information, see About Dual Address Logic on page 465.</p> <p>N (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority:</p> <ol style="list-style-type: none"> 1. PO Box 2. Firm 3. Highrise 4. Street 5. Rural Route 6. General Delivery <p>S Return a street match, regardless of the address line.</p> <p>P Return a PO Box match, regardless of the address line.</p> |
| Option.StreetMatchingStrictness | <p>The strictness of the street name match (U.S. addresses only).</p> <p>E The input street name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| Option.FirmMatchingStrictness | <p>The strictness of the firm name match (U.S. addresses only).</p> <p>E The input firm name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |

| Parameter | Description |
|---|---|
| Option.DirectionalityMatchingStrictness | <p>The strictness of the directional match.</p> <p>E The input directional must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| Option.PerformESM | <p>Specifies whether or not to perform Enhanced Street Matching (ESM). ESM applies extra matching logic with additional data to any input address that is not matched through the regular address validation process. ESM applies to U.S. addresses only.</p> <p>Y Yes, perform ESM processing.</p> <p>N No, do not perform ESM processing (default).</p> |
| Option.AddressLineSearchOnFail | <p>Specifies whether ValidateAddress will search address lines for the city, state/province, and postal code.</p> <p>This option enables ValidateAddress to search the AddressLine input fields for the city, state/province, postal code, and country when the address cannot be matched using the values in the City, StateProvince, and PostalCode input fields.</p> <p>Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields.</p> <p>Consider disabling this option if your input addresses use the City, State/Province and PostalCode fields. If you enable this option and these fields are used, there is an increased possibility that ValidateAddress will fail to correct values in these fields (for example a misspelled city name).</p> <p>Y Yes, search the address line fields (default).</p> <p>N No, do not search the AddressLine fields.</p> |

Response

GetCandidateAddresses returns the following output.

Table 22: GetCandidateAddresses Output

| Response Element | Description |
|---|--|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| AddressLine5 | For U.K. addresses only. If the address was validated, the fifth line of the validated and standardized address. If the address could not be validated, the fifth line of the input address without any changes. |
| CanadianDeliveryInstallation AreaName | Delivery installation name (Canadian addresses only) |
| CanadianDeliveryInstallation QualifierName | Delivery installation qualifier (Canadian addresses only) |
| CanadianDeliveryInstallation Type | Delivery installation type (Canadian addresses only) |
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| FirmName | The firm name. |

| Response Element | Description |
|-------------------|---|
| HouseNumberHigh | The ending house number for the range in which the candidate address's house number falls. |
| HouseNumberLow | The beginning house number for the range in which the candidate address's house number falls. |
| HouseNumberParity | <p>Indicates the numbering scheme for the house numbers between HouseNumberLow and HouseNumberHigh, as follows:</p> <p>E Only even values</p> <p>O Only odd values</p> <p>B Both</p> |
| MatchLevel | <p>For addresses outside the U.S. and Canada, identifies the match level for the candidate address. U.S. and Canadian addresses are always "A." One of the following:</p> <p>A The candidate matches the input address at the street level.</p> <p>B The candidate matches the input address at the state/province level.</p> |
| PostalCode | The postal code. In the U.S. this is the ZIP Code™. |
| PostalCode.AddOn | The last four digits of the ZIP + 4® Code. U.S. addresses only. |
| RecordType | <p>The type of address record, as defined by U.S. and Canadian postal authorities (U.S. and Canadian addresses only):</p> <ul style="list-style-type: none"> • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal |

| Response Element | Description |
|--------------------|--|
| RecordType.Default | Code indicating the "default" match: Y The address matches a default record. null The address does not match a default record. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. There is only one possible value: <ul style="list-style-type: none"> • DisabledCoder • RequestFailed |
| Status.Description | A description of the problem, if there is one. Did not return multiples The input address matched only one address in the database. GetCandidateAddresses only returns data if multiple possible matches were found. Number of candidates is not greater than 1 The input address matched more than one address in the database but no addresses were returned. PerformUSProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformCanadianProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformInternationalProcessing disabled This value will appear if Status.Code=DisabledCoder. |
| UnitNumberHigh | The ending unit number for the range in which the candidate address's unit number falls. |

| Response Element | Description | | | | | | |
|------------------|---|----------|------------------|----------|-----------------|----------|------|
| UnitNumberLow | The beginning unit number for the range in which the candidate address's unit number falls. | | | | | | |
| UnitNumberParity | Indicates the numbering scheme for the unit numbers between UnitNumberLow and UnitNumberHigh, as follows: <table border="0" data-bbox="552 577 974 714"> <tr> <td>E</td> <td>Only even values</td> </tr> <tr> <td>O</td> <td>Only odd values</td> </tr> <tr> <td>B</td> <td>Both</td> </tr> </table> | E | Only even values | O | Only odd values | B | Both |
| E | Only even values | | | | | | |
| O | Only odd values | | | | | | |
| B | Both | | | | | | |
| USUrbanName | The validated city urbanization name. Urbanization names are used primarily for Puerto Rico addresses. | | | | | | |

GetCandidateAddressesLoqate

GetCandidateAddressesLoqate returns a list of addresses that are considered matches for a given input address. GetCandidateAddressesLoqate returns candidate addresses only if the input address matches multiple addresses in the postal database. If the input address matches only one address in the postal database, then no address data is returned. The Country input field is required; if this field is blank, no output will be returned.

Note: By default, GetCandidateAddressesLoqate does not match to individual house numbers. Rather, it uses house number ranges for each street. After GetCandidateAddressesLoqate has determined the street name, city name, state/province name, and postal code, it checks to make sure the input house number falls within one of the ranges of house numbers given for the matched street name. The same type of logic applies to unit numbers.

GetCandidateAddressesLoqate is part of Spectrum Universal Address.

Resource URL

JSON endpoint:

```
http://server:port/rest/GetCandidateAddressesLoqate/results.json
```

XML endpoint:

```
http://server:port/rest/GetCandidateAddressesLoqate/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/GetCandidateAddressesLoqate/results.json?
Data.AddressLine1=PO+Box+1&Data.City=New+York&Data.StateProvince=NY
```

The JSON returned by this request would be:

Note: Empty response elements have been removed from this example. Only the first two candidate address are shown.

```
{ "output_port": [
  {
    "ProcessedBy": "LOQATE",
    "AddressLine1": "PO Box 101",
    "City": "New York Mls",
    "StateProvince": "NY",
    "PostalCode": "13417-0101",
    "Country": "USA",
    "PostalCode.AddOn": "0101",
    "user_fields": []
  },
  {
    "ProcessedBy": "LOQATE",
    "AddressLine1": "PO Box 102",
    "City": "New York Mls",
    "StateProvince": "NY",
    "PostalCode": "13417-0102",
    "Country": "USA",
    "PostalCode.AddOn": "0102",
    "user_fields": []
  }
] }
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/GetCandidateAddressesLoqate/results.xml?
Data.AddressLine1=PO+Box+1&Data.City=New+York&Data.StateProvince=NY
```

The XML returned by this request would be:

Note: Empty response elements have been removed from this example. Only the first two candidate address are shown.

```
<ns2:xml.GetCandidateAddressesLoqateResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GetCandidateAddressesLoqate">
```

```

<ns2:output_port>
  <ns2:Address>
    <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
    <ns2:AddressLine1>PO Box 101</ns2:AddressLine1>
    <ns2:City>New York Mls</ns2:City>
    <ns2:StateProvince>NY</ns2:StateProvince>
    <ns2:PostalCode>13417-0101</ns2:PostalCode>
    <ns2:PostalCode.AddOn>0101</ns2:PostalCode.AddOn>
    <ns2:Country>USA</ns2:Country>
  </ns2:Address>
  <ns2:Address>
    <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
    <ns2:AddressLine1>PO Box 102</ns2:AddressLine1>
    <ns2:City>New York Mls</ns2:City>
    <ns2:StateProvince>NY</ns2:StateProvince>
    <ns2:PostalCode>13417-0102</ns2:PostalCode>
    <ns2:PostalCode.AddOn>0102</ns2:PostalCode.AddOn>
    <ns2:Country>USA</ns2:Country>
  </ns2:Address>
</ns2:output_port>
</ns2:xml.GetCandidateAddressesLoqateResponse>

```

Request

Parameters for Input Data

The following table lists the input for GetCandidateAddressesLoqate.

Table 23: Input Format

| Parameter | Description |
|-------------------|--------------------------|
| Data.AddressLine1 | The first address line. |
| Data.AddressLine2 | The second address line. |
| Data.AddressLine3 | The third address line. |
| Data.AddressLine4 | The fourth address line. |
| Data.City | The city name. |

| Parameter | Description |
|--------------------|---|
| Data.Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> <p>Note: This field is required. If this field is blank, no output will be returned.</p> |
| Data.FirmName | The company or firm name. |
| Data.PostalCode | The postal code for the address. For U.S. addresses this is the ZIP Code™ in one of the following formats: |
| Data.StateProvince | <p>The state or province.</p> <p>For U.S. addresses only, you may put the state in the City field instead of the StateProvince field.</p> |

Parameters for Options

Table 24: GetCandidateAddressesLoqate Options

| Parameter | Description |
|------------------------|--|
| Option.Database.Loqate | Specifies the database to be used for address processing. Only databases that have been defined in the Management Console are available. |

| Parameter | Description |
|-------------------------------|--|
| Option.OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p>123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p>123 MAIN ST MYTOWN FL 12345</p> |
| Option.CandidateProcessOption | <p>Specifies the method of searching for candidates. One of the following:</p> <p>S Enter a full or partial address as input and return as output a list of closely matching results (default).</p> <p>V Enter address information in address lines, address components, or a combination of both as input and return as output results that more closely match the input.</p> |

| Parameter | Description |
|----------------------------|---|
| Option.HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. GetCandidateAddressLoqate uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent And The Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |
| Option.OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |

| Parameter | Description |
|-----------------------|--|
| Option.OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |
| Option.MaximumResults | The maximum number of candidate addresses that GetCandidateAddressesLoqate should return. The default is 10. The maximum is 99. |

Response

GetCandidateAddressesLoqate returns the following output.

Table 25: GetCandidateAddressesLoqate Output

| Response Element | Description |
|------------------|--|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |

| Response Element | Description |
|--------------------|--|
| FirmName | The firm name. |
| PostalCode | The postal code. In the U.S. this is the ZIP Code™. |
| PostalCode.AddOn | The last four digits of the ZIP + 4® Code. U.S. addresses only. |
| ProcessedBy | Indicates which address coder processed the address. LOQATE The Loqate coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. There is only one possible value: • RequestFailed |
| Status.Description | A description of the problem, if there is one. There is only one possible value: Did not return multiples The input address matched only one address in the database. GetCandidateAddressesLoqate only returns data if multiple possible matches were found. |

GetCityStateProvince

GetCityStateProvince returns a city and state/province for a given input postal code.

Note: GetCityStateProvince works with U.S. and Canadian addresses only.

GetCityStateProvince is part of Spectrum Universal Address.

Resource URL

JSON endpoint:

```
http://server:port/rest/GetCityStateProvince/results.json
```

XML endpoint:

```
http://server:port/rest/GetCityStateProvince/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/GetCityStateProvince/results.json?
Data.PostalCode=12180
```

The JSON returned by this request would be:

```
{ "output_port": [ {
  "ProcessedBy": "USA",
  "PostalCode": "12180",
  "City": "TROY",
  "StateProvince": "NY",
  "Country": "USA",
  "City.Type": "P",
  "user_fields": [ ]
} ] }
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/GetCityStateProvince/results.xml?
Data.PostalCode=12180
```

The XML returned by this request would be:

```
<ns2:xml.GetCityStateProvinceResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GetCityStateProvince">

  <ns2:output_port>
    <ns2:Result>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:PostalCode>12180</ns2:PostalCode>
      <ns2:City>TROY</ns2:City>
      <ns2:City.Type>P</ns2:City.Type>
      <ns2:StateProvince>NY</ns2:StateProvince>
    </ns2:Result>
  </ns2:output_port>
</ns2:xml.GetCityStateProvinceResponse>
```

```

    <ns2:Country>USA</ns2:Country>
    <ns2:user_fields/>
  </ns2:Result>
</ns2:output_port>
</ns2:xml.GetCityStateProvinceResponse>

```

Request

Parameters for Input Data

The following table shows the input fields.

Table 26: GetCityStateProvince Input

| Parameter | Description |
|-----------------|---|
| Data.PostalCode | A U.S. ZIP Code™ or Canadian postal code in one of the following formats: 99999 99999-9999 A9A9A9 A9A 9A9 |

Parameters for Options

Table 27: GetCityStateProvince Options

| Parameter Name | Description |
|----------------------------------|---|
| Option.PerformUSProcessing | <p>Specifies whether or not to process U.S. addresses. If you enable U.S. address processing GetCityStateProvince will attempt to return the state for U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process U.S. addresses (default).</p> <p>N No, do not process U.S. addresses.</p> |
| Option.Database.US | <p>Specifies the database to be used for U.S. address processing. Only databases that have been defined in the US Database Resources panel in the Management Console are available.</p> |
| Option.PerformCanadianProcessing | <p>Specifies whether or not to process Canadian addresses. If you enable Canadian address processing GetCityStateProvince will attempt to return the province for Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process Canadian addresses (default).</p> <p>N No, do not process Canadian addresses.</p> |

| Parameter Name | Description |
|-------------------------|--|
| Option.Database.Canada | Specifies the database to be used for Canadian address processing. Only databases that have been defined in the Canadian Database Resources panel in the Management Console are available. |
| Option.OutputVanityCity | Specifies whether or not to include non-mailing city names in the output. A non-mailing city name is an alternate name for the primary city name. For example, Hollywood is a non-mailing city name for Los Angeles. Y Yes, include non-mailing city names. N No, do not include non-mailing city names (default). |
| Option.MaximumResults | Specifies the maximum number of city-state/province pairs to return. The default value is 10. |

Response

GetCityStateProvince returns the matching city and state/province for the input postal code as well as a code to indicate the success or failure of the match attempt. If more than one city/state or city/province matches the input postal code, multiple output records are returned.

Table 28: GetCityStateProvince Output

| Response Element | Description |
|------------------|--|
| City | The matched city name. |
| City.Type | The USPS® standardized city name type (U.S. addresses only). V Vanity (non-mailing) city name. P Primary. The city name is the primary mailing city name. S Secondary. The city name is an alternate city name but is acceptable. A city can have multiple secondary city names. |
| PostalCode | The input postal code. |

| Response Element | Description |
|--------------------|---|
| ProcessedBy | Indicates which address coder processed the address. One of the following: USA The U.S. address coder processed the address. CAN The Canadian address coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. The only valid value is: <ul style="list-style-type: none"> • DisabledCoder • UnrecognizedPostalCode |
| Status.Description | The description of the failure. The valid values are: Postal code not found This value will appear if Status.Code=UnrecognizedPostalCode. PerformUSProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformCanadianProcessing disabled This value will appear if Status.Code=DisabledCoder. |

GetCityStateProvinceLoqate

GetCityStateProvinceLoqate returns a city and state/province for a given input postal code.

This stage is part of the Spectrum Universal Adresse.

Resource URL

JSON endpoint:

```
http://server:port/rest/GetCityStateProvinceLoqate/results.json
```

XML endpoint:

```
http://server:port/rest/GetCityStateProvinceLoqate/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/GetCityStateProvinceLoqate/results.json?
Data.Country=USA&Data.PostalCode=60510
```

The JSON returned by this request would be:

```
{ "output_port": [{
  "ProcessedBy": "LOQATE",
  "PostalCode": "60510",
  "City": "Batavia",
  "StateProvince": "IL",
  "Country": "United States",
  "Status": "",
  "Status.Code": "",
  "Status.Description": "",
  "user_fields": []
}]}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/GetCityStateProvinceLoqate/results.xml?Data.Country=USA&
Data.PostalCode=60510
```

The XML returned by this request would be:

```
<ns2:xml.GetCityStateProvinceLoqateResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GetCityStateProvinceLoqate">

  <ns2:output_port>
    <ns2:Result>
      <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
      <ns2:PostalCode>60510</ns2:PostalCode>
      <ns2:City>Batavia</ns2:City>
      <ns2:StateProvince>IL</ns2:StateProvince>
      <ns2:Country>United States</ns2:Country>
      <ns2:Status/>
      <ns2:Status.Code/>
      <ns2:Status.Description/>
      <ns2:user_fields/>
    </ns2:Result>
  </ns2:output_port>
</ns2:xml.GetCityStateProvinceLoqateResponse>
```

```
</ns2:output_port>
</ns2:xml.GetCityStateProvinceLoqateResponse>
```

Request

Parameters for Input Data

The following table shows the input fields.

Table 29: GetCityStateProvinceLoqate Input

| Parameter | Description |
|-----------------|---|
| Data.Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| Data.PostalCode | The postal code for the address. |

Options

Table 30: GetCityStateProvinceLoqate Options

| Description / Valid Values |
|---|
| Specifies the database to be used for address processing. Only databases that have been defined in the Database Resources panel in the Management Console are available. |
| The maximum number of addresses that GetCityStateProvinceLoqate should return. The default is 10. |

Description / Valid Values

| | |
|------------------------|--|
| Option.OutputScript | Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native. |
| Input | Do not perform transliteration and provide output in the same script as the input (default). |
| Native | Output in the native script for the selected country wherever possible. |
| Latn | Use English values. |
| | Specifies how you want Spectrum Technology Platform to respond when a data license error occurs. |
| Fail the job | Fail the entire job if a data license error occurs. |
| Fail the record | Fail the record(s) for which the data license error occurs and continue processing. |

Response

GetCityStateProvinceLoqate returns the matching city and state/province for the input postal code as well as a code to indicate the success or failure of the match attempt. If more than one city/state or city/province matches the input postal code, multiple output records are returned.

Table 31: GetCityStateProvinceLoqate Output

| Response Element | Description |
|------------------|---|
| City | The matched city name. |
| Country | The country in the format determined by what you selected in: <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| PostalCode | The input postal code. |

XML endpoint:

```
http://server:port/rest/GetPostalCodes/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/GetPostalCodes/results.json?
Data.City=Holland&Data.StateProvince=MI
```

The JSON returned by this request would be:

```
{ "output_port": [
  {
    "ProcessedBy": "USA",
    "PostalCode": "49422",
    "Status": "",
    "City.Type": " ",
    "Status.Code": "",
    "Status.Description": "",
    "user_fields": []
  },
  {
    "ProcessedBy": "USA",
    "PostalCode": "49423",
    "Status": "",
    "City.Type": " ",
    "Status.Code": "",
    "Status.Description": "",
    "user_fields": []
  },
  {
    "ProcessedBy": "USA",
    "PostalCode": "49424",
    "Status": "",
    "City.Type": " ",
    "Status.Code": "",
    "Status.Description": "",
    "user_fields": []
  }
] }
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/GetPostalCodes/results.xml?Data.City=Holland&
Data.StateProvince=MI
```

The XML returned by this request would be:

```
<ns2:xml.GetPostalCodesResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GetPostalCodes">
  <ns2:output_port>
    <ns2:Result>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:PostalCode>49422</ns2:PostalCode>
      <ns2:City.Type></ns2:City.Type>
      <ns2:Status/>
      <ns2:Status.Code/>
      <ns2:Status.Description/>
      <ns2:user_fields/>
    </ns2:Result>
    <ns2:Result>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:PostalCode>49423</ns2:PostalCode>
      <ns2:City.Type></ns2:City.Type>
      <ns2:Status/>
      <ns2:Status.Code/>
      <ns2:Status.Description/>
      <ns2:user_fields/>
    </ns2:Result>
    <ns2:Result>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:PostalCode>49424</ns2:PostalCode>
      <ns2:City.Type></ns2:City.Type>
      <ns2:Status/>
      <ns2:Status.Code/>
      <ns2:Status.Description/>
      <ns2:user_fields/>
    </ns2:Result>
  </ns2:output_port>
</ns2:xml.GetPostalCodesResponse>
```

Request

Parameters for Input Data

GetPostalCodes takes a city, state/province, and country as input.

Table 32: GetPostalCodes Input

| Parameter | Description |
|--------------------|---|
| Data.City | <p>The city whose postal codes you want to look up.</p> <p>You may put the city and state in the City field. If you do this, you must leave the StateProvince field blank.</p> <p>The total length of the City and StateProvince fields cannot exceed 100 characters.</p> |
| Data.StateProvince | <p>The state or province of the city whose postal codes you want to look up.</p> <p>You may also put the state in the City field instead of the StateProvince field.</p> <p>The total length of the City and StateProvince fields cannot exceed 100 characters.</p> |
| Data.Country | <p>The country code or name of the city whose postal codes you want to look up. The only valid value is US.</p> |

Parameters for Options**Table 33: GetPostalCodes Options**

| Parameter | Description |
|--------------------------|---|
| Option.Database.US | <p>Specifies the database to be used for postal code look-ups. Only databases that have been defined in the US Database Resources panel in the Management Console are available.</p> |
| Option.IncludeVanityCity | <p>Specifies whether or not to include postal codes for the city's non-mailing city names. A non-mailing city name is an alternate name for the primary city name. For example, Hollywood is a non-mailing city name for Los Angeles.</p> <p>Y Yes, include postal codes for non-mailing city names.</p> <p>N No, do not include postal codes for non-mailing city names (default).</p> |
| Option.OutputCityType | <p>Specifies whether or not to return the city type in the output. If enabled, the city type is returned in the City.Type field.</p> <p>Y Yes, include the city type in the output.</p> <p>N No, do not include the city type in the output (default).</p> |

Response

GetPostalCodes returns the postal codes for a specified city. Each postal code is returned in a separate record along with the data listed in the following table.

Table 34: GetPostalCodes Output

| Response Element | Description |
|------------------|---|
| City.Type | <p>The USPS® city type (U.S. addresses only). The city type is determined by looking at the ZIP Code and the city name. For example, the city Lanham MD has the postal codes 20703, 20706, and 20784. Lanham is the primary city in 20703 and 20706 but is a vanity city in 20784.</p> <p>This field column is only populated if <code>Option.OutputCityType=Y</code>. The possible values are:</p> <p>V Vanity (non-mailing) city name.</p> <p>P Primary. The city name is the primary mailing city name.</p> <p>S Secondary. The city name is an alternate city name but is acceptable. A city can have multiple secondary city names.</p> |
| PostalCode | A postal code in the specified city. |
| ProcessedBy | Because this service only works for U.S. addresses, ProcessedBy will always contain one value: USA. |
| Status | <p>Reports the success or failure of the match attempt.</p> <p>null Success</p> <p>F Failure</p> |
| Status.Code | <p>Reason for failure, if there is one. One of the following:</p> <ul style="list-style-type: none"> • CountryNotSupported • UnableToLookup |

| Response Element | Description |
|--------------------|--|
| Status.Description | <p>Description of failure.</p> <ul style="list-style-type: none"> • Input country is not supported • Input city was blank • Input city & state / province was blank, or no match found • City-state mismatch (different spelling found, or city-state was a vanity name and vanity matching was not allowed, or city-state did not match ZIP Code) |

ValidateAddress

ValidateAddress standardizes and validates addresses using postal authority address data. It can correct information and format the address using the format preferred by the applicable postal authority. It also adds missing postal information, such as postal codes, city names, state/province names, and more.

ValidateAddress also returns result indicators about validation attempts, such as whether or not it validated the address, the level of confidence in the returned address, the reason for failure if the address could not be validated, and more.

During address matching and standardization, ValidateAddress separates address lines into components and compares them to the contents of the Universal Addressing Module databases. If a match is found, the input address is *standardized* to the database information. If no database match is found, it optionally *formats* the input addresses. The formatting process attempts to structure the address lines according to the conventions of the appropriate postal authority.

ValidateAddress is part of the Universal Addressing Module.

Resource URL

JSON endpoint:

```
http://server:port/rest/ValidateAddress/results.json
```

XML endpoint:

```
http://server:port/rest/ValidateAddress/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/ValidateAddress/results.json?Data.AddressLine1=1825+Kramer+Ln&Data.PostalCode=78758
```

The JSON returned by this request would be:

```
{
  "output_port": [
    {
      "Confidence": "100",
      "RecordType": "Normal",
      "CountryLevel": "A",
      "ProcessedBy": "USA",
      "MatchScore": "0",
      "AddressLine1": "1825 Kramer Ln",
      "City": "Austin",
      "StateProvince": "TX",
      "PostalCode": "78758-4260",
      "Country": "United States Of America",
      "PostalCode.Base": "78758",
      "PostalCode.AddOn": "4260",
      "user_fields": [

    ]
  ]
}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/ValidateAddress/results.xml?Data.AddressLine1=1825+Kramer+Ln&Data.PostalCode=78758
```

The XML returned by this request would be:

```
<ns2:xml.ValidateAddressResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/ValidateAddress">

  <ns2:output_port>
    <ns2:Address>
      <ns2:Confidence>93</ns2:Confidence>
      <ns2:RecordType>Normal</ns2:RecordType>
      <ns2:CountryLevel>A</ns2:CountryLevel>
      <ns2:ProcessedBy>USA</ns2:ProcessedBy>
      <ns2:MatchScore>0</ns2:MatchScore>
    
```

```

    <ns2:AddressLine1>1825 Kramer Ln</ns2:AddressLine1>
    <ns2:City>Austin</ns2:City>
    <ns2:StateProvince>TX</ns2:StateProvince>
    <ns2:PostalCode>78758-4260</ns2:PostalCode>
    <ns2:PostalCode.Base>78758</ns2:PostalCode.Base>
    <ns2:PostalCode.AddOn>4260</ns2:PostalCode.AddOn>
    <ns2:Country>United States Of America</ns2:Country>
    <ns2:user_fields/>
  </ns2:Address>
</ns2:output_port>
</ns2:xml.ValidateAddressResponse>

```

Request

Parameters for Input Data

ValidateAddress takes an address as input. All addresses use this format regardless of the address's country. See [Address Line Processing for U.S. Addresses](#) on page 452 for important information about how address line data is processed for U.S. addresses.

Table 35: Input Format

| Parameter | Format | Description |
|-------------------|-------------|--|
| Data.AddressLine1 | String [50] | The first address line. |
| Data.AddressLine2 | String [50] | The second address line. |
| Data.AddressLine3 | String [50] | The third address line. Does not apply to Canadian addresses. |
| Data.AddressLine4 | String [50] | The fourth address line. Does not apply to Canadian addresses. |
| Data.AddressLine5 | String [50] | The fifth address line. Applies only to U.K. addresses. May contain street name, unit number, building number, and so on. |

| Parameter | Format | Description |
|--------------------|-------------|--|
| Data.City | String [50] | <p>The city name.</p> <p>For U.S. addresses only, you may put the city, state, and ZIP Code™ in the City field. If you do this, you must leave the StateProvince and PostalCode fields blank.</p> |
| Data.StateProvince | String [50] | <p>The state or province.</p> <p>For U.S. addresses only, you may put the state in the City field instead of the StateProvince field.</p> |
| Data.PostalCode | String [10] | <p>The postal code for the address in one of the following formats:</p> <p>99999 99999-9999 A9A9A9 A9A 9A9 9999 999</p> <p>For U.S. addresses only, you may put the ZIP Code™ in the City field.</p> <p>For U.S. addresses only, if the city/state/ZIP Code™ is in the PostalCode field, ValidateAddress may parse the data and successfully process the address. For best results, put this data in the appropriate fields (City, StateProvince, and PostalCode).</p> |
| Data.Country | String [50] | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • Two-character ISO 3166-1 Alpha 2 country code • Three-character ISO 3166-1 Alpha 3 country code • English country name • French country name • German country name • Spanish country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| Data.FirmName | String [50] | The company or firm name. |
| Data.USUrbanName | String [50] | The U.S. address urbanization name. This is used primarily for Puerto Rico addresses. |

| Parameter | Format | Description |
|------------------|------------|---|
| Data.CustomerID | String [9] | If this mailpiece uses a generic barcode, specify your USPS®-assigned customer ID in this field. The ValidateAddress generic barcode is used for mailpieces that use the OneCode ACS® service. |
| Data.CanLanguage | String | For Canadian addresses only, indicates whether the address is in English or French, if the option <code>Option.CanFrenchFormat=T</code> is used. If this field is blank, the address is formatted in English. If the field contains any non-blank value, the address is formatted in French. Note that addresses in Quebec are always formatted in French regardless of the value in this field. |

Address Line Processing for U.S. Addresses

The input fields AddressLine1 through AddressLine4 are handled differently for U.S. addresses depending on whether the firm name extraction or urbanization code extraction options are enabled. If either of these options is enabled, ValidateAddress will look at the data in all four fields to validate the address and extract the requested data (firm name and/or urbanization code). If neither of these options is enabled, ValidateAddress uses only the first two non-blank address line fields in its validation attempt. The data in the other address line fields is returned in the output field AdditionalInputData. For example,

AddressLine1: A1 Calle A
AddressLine2:
AddressLine3: URB Alamar
AddressLine4: Precisely

In this address, if either firm name extraction or urbanization code extraction were enabled, ValidateAddress would examine all four address lines. If neither firm name extraction nor urbanization code extraction were enabled, ValidateAddress would examine AddressLine1 and AddressLine3 (the first two non-blank address lines) and attempt to validate the address using that data; the data in AddressLine4 would be returned in the output field AdditionalInputData.

Parameters for Options

Output Data Options

The following table lists the options that control the type of information returned by ValidateAddress. Some of these options can be overridden for Canadian addresses. For more information, see [Canadian Address Options](#) on page 480.

Table 36: Output Data Options

| Parameter | Description |
|-------------------------|--|
| Option.OutputRecordType | <p>Type of output record. For more than one, provide a list.</p> <ul style="list-style-type: none"> <li data-bbox="686 485 1422 890">A Returns 1 to 4 lines of address data plus city, state, postal code, firm name, and urbanization name information. Each address line represents an actual line of the address as it would appear on an envelope. For more information, see Standard Address Output on page 491. If the address is validated, the address lines contain the standardized address. When addresses are standardized, punctuation is removed, directionals are abbreviated, street suffixes are abbreviated, and address elements are corrected. If the address is not validated, the address lines contain the address as it appeared in the input ("pass through" data). Non-validated addresses are always included as pass through data in the address line fields even if you do not specify <code>OutputRecordType=A</code>. <li data-bbox="686 915 1422 1129">E Parsed address elements. Each part of the address, such as house number, street name, street suffix, directionals, and so on is returned in a separate field. For more information, see Parsed Address Elements Output on page 567. Note that if you specify "E" and specify <code>OutputFormattedOnFail=Y</code>, the parsed address elements will contain the input address for addresses that could not be validated. <li data-bbox="686 1155 1422 1436">I Parsed input. This option returns the input address in parsed form regardless of whether the address is validated. Each part of the input address, such as house number, street name, street suffix, directionals, and so on is returned in a separate field. Parsed input (value "I") differs from the combination of <code>OutputRecordType=E</code> and <code>OutputFormattedOnFail=Y</code> in that "I" returns all input address in parsed form, not just input that could not be validated. For more information, see Parsed Input on page 569. <li data-bbox="686 1461 1422 1545">P Postal data. Output addresses contain additional data for each validated address. For more information, see Postal Data Output on page 497. <p>Blank Do not return any address data or postal data.</p> |

| Parameter | Description |
|------------------------------------|--|
| Option.OutputFieldLevelReturnCodes | <p data-bbox="677 380 1422 554">Specifies whether to include field-level result indicators. Field-level result indicators describe how each address element was handled. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in HouseNumber.Result. For a complete listing of result indicator output fields, see Field-Level Result Indicators on page 504.</p> <p data-bbox="683 575 1333 602">N No, do not output field-level return codes (default).</p> <p data-bbox="683 623 1175 651">Y Yes, output field-level return codes.</p> |

| Parameter | Description |
|------------------------------|---|
| Option.OutputFormattedOnFail | <p>Specifies whether to return a formatted address when an address cannot be validated. The address is formatted using the preferred address format for the address's country. If this option is not selected, the output address fields are blank when the address cannot be validated.</p> <p>Note: This option applies only to U.S. and Canadian addresses. Formatted data will not be returned for any other address.</p> <p>N No, do not format failed addresses (default).</p> <p>Y Yes, format failed addresses.</p> <p>Formatted addresses are returned using the format specified by the <code>OutputRecordType</code> option. Note that if you specify <code>OutputRecordType=E</code>, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, specify <code>OutputRecordType=I</code>.</p> <p>Formatted addresses are returned using the format specified by the <code>Option.OutputRecordType</code> option. Note that if you specify <code>Option.OutputRecordType=E</code>, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, specify <code>Option.OutputRecordType=I</code>.</p> <p>Formatted addresses are returned using the format specified by the Include a standard address, Include address line elements, and Include postal information check boxes. Note that if you select Include address line elements, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, select Include standardized input address elements.</p> <p>If you specify Y, you must specify "A" and/or "E" for <code>OutputRecordType</code>.</p> <p>If you specify Y, you must specify "A" and/or "E" for <code>Option.OutputRecordType</code>.</p> <p>If you check this option, you must select Include a standard address and/or Include address line elements.</p> |

| Parameter | Description |
|------------------------------|---|
| Option.OutputStreetNameAlias | <p data-bbox="677 380 1427 464">For U.S. addresses only, specifies how to handle street name aliases used in the input. A street alias is an alternate name for a street and applies only to a specific range of addresses on the street.</p> <p data-bbox="677 485 1427 600">If you enable this option, street name aliases used in the input will appear in the output. If you do not enable this option, street name aliases in the input will be converted to the base street name in the output, with the following exceptions:</p> <ul data-bbox="677 621 1427 737" style="list-style-type: none"><li data-bbox="677 621 1427 674">• If a preferred alias is used in input the preferred alias will always be used in output.<li data-bbox="677 684 1427 737">• Changed aliases used in input are always converted to the base street name in output. <p data-bbox="677 758 1427 894">This is one of three options that control how ValidateAddress handles street name aliases. The other two are <code>Option.OutputPreferredAlias</code> and <code>Option.OutputAbbreviatedAlias</code>.</p> <p data-bbox="760 915 1427 1031">Note: If <code>Option.OutputAbbreviatedAlias</code> is enabled, the abbreviated alias will always appear in the output even if you have <code>Option.OutputStreetNameAlias</code> disabled.</p> <p data-bbox="677 1073 1427 1176">N No, do not return street name aliases in the output. Y Yes, return street name aliases in the output if the input street name is an alias (default).</p> |

| Parameter | Description |
|----------------------------|---|
| Option.OutputAddressBlocks | <p>Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p>For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p>ValidateAddress formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option <code>Option.OutputCountryFormat</code> does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p>For addresses outside the U.S. and Canada, if ValidateAddress is unable to validate the address, no address blocks are returned. For addresses in the U.S. and Canada, address blocks are returned even if validation fails.</p> <p>N No, do not return address blocks. Default.</p> <p>Y Yes, return address blocks.</p> |

| Parameter | Description |
|-------------------|---|
| Option.OutputAMAS | <p>Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p>For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p>ValidateAddress formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option <code>Option.OutputCountryFormat</code> does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p>For addresses outside the U.S. and Canada, if ValidateAddress is unable to validate the address, no address blocks are returned. For addresses in the U.S. and Canada, address blocks are returned even if validation fails.</p> <p>N No, do not return address blocks. Default. Y Yes, return address blocks.</p> |

Obtaining Congressional Districts

ValidateAddress can determine the U.S. congressional district for an address.

To obtain congressional districts, `Option.OutputRecordType` must contain P. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

Table 37: Congressional District Output

| Response Element | Description |
|-------------------------|--|
| USCongressionalDistrict | Congressional district number. If the address is a non-state address (for example Puerto Rico or Washington D.C.) this field is blank. |

Obtaining County Names

ValidateAddress can determine the county where a particular address is located and return the county name.

Note: County names are available for U.S. addresses only.

To obtain county names, `Option.OutputRecordType` must contain P. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

Table 38: County Name Output

| Response Element | Description |
|------------------|-------------|
| USCountyName | County name |

Obtaining FIPS County Numbers

Federal Information Processing Standards (FIPS) county numbers are numbers that identify each county in a state. Note that these numbers are only unique at the state level, not the national level. For more information, see <http://www.census.gov>.

Note: FIPS county numbers are available for U.S. addresses only.

To obtain FIPS county numbers, `Option.OutputRecordType` must contain P. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

Table 39: FIPS County Number Output

| Response Element | Description |
|--------------------|---|
| USFIPSCountyNumber | FIPS (Federal Information Processing Standards) county number |

Obtaining Carrier Route Codes

Carrier route codes are unique identifiers assigned to each mail carrier who delivers mail, allowing unique identification of each U.S. delivery route. ValidateAddress can return the code that represents an addressee's carrier route.

Note: Carrier route codes are available for U.S. addresses only.

To obtain carrier route codes, `Option.OutputRecordType` must contain P. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

Table 40: Carrier Route Code Output

| Response Element | Description |
|--------------------|--------------------|
| USCarrierRouteCode | Carrier route code |

Creating Delivery Point Barcodes

A Delivery Point Barcode (DPBC) is a POSTNET™ barcode representation of the address. It consists of 62 bars with beginning and ending frame bars and five bars each for the ZIP + 4® Code, a value calculated based on the street address number, and a correction digit. The DPBC allows automated sortation of letter mail to the carrier level in walk sequence. `ValidateAddress` generates the data you need to assemble a DPBC.

Note: Delivery Point Barcodes are available for U.S. addresses only. For more information on Delivery Point Barcodes, see <http://www.usps.com>.

To generate the data needed to assemble a DPBC, `Option.OutputRecordType` must contain P. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

Table 41: Delivery Point Barcode Output

| Response Element | Description |
|------------------|---|
| PostalBarCode | The delivery point portion of the delivery point barcode. |
| USBCCheckDigit | Check-digit portion of the 11-digit delivery point barcode. |

To assemble a DPBC you concatenate the values found in the `ValidateAddress` output as follows:

`PostalCode.Base` + `PostalCode.Addon` + `PostalBarcode` + `USBCCheckDigit`

For example, if you have the following:

- **PostalCode.Base** = 49423
- **PostalCode.Addon** = 4506
- **PostalBarcode** = 29
- **USBCCheckDigit** = 2

The assembled barcode would be:

494234506292

Default Options

The following table lists the options that control the format and processing of addresses. These are called "default options" because by default they apply to all addresses. Some of these options can be overridden for Canadian addresses. For more information, see [Canadian Address Options](#) on page 480.

Table 42: Default Options

| Parameter | Description |
|----------------------------------|--|
| Option.OutputCasing | <p>Specifies the casing of the output address. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p style="padding-left: 40px;">123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p style="padding-left: 40px;">123 MAIN ST MYTOWN FL 12345</p> |
| Option.OutputPostalCodeSeparator | <p>Specifies whether to use separators (spaces or hyphens) in ZIP™ Codes or Canadian postal codes.</p> <p>For example, a ZIP + 4® Code with the separator would be 20706-1844 and without the separator it would be 207061844. A Canadian postal code with the separator would be P5E"1S7 and without the separator it would be P5E1S7.</p> <p>Y Yes, use separator (default).</p> <p>N No, do not use separator.</p> <p>Note: Spaces are used in Canadian postal codes and hyphens in U.S. ZIP + 4® Codes.</p> |

| Parameter | Description |
|--------------------------------------|---|
| Option.OutputMultinationalCharacters | <p>Specifies whether or not to return multinational characters, including diacritical marks such as umlauts or accents. (Not supported for U.S. addresses).</p> <p>N No, do not use multinational characters in the output (default). Only standard ASCII characters is returned.</p> <p>Y Yes, use multinational characters in the output.</p> |
| Option.KeepMultimatch | <p>Indicates whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> <p>For more information, see Returning Multiple Matches on page 466.</p> |
| Option.StandardAddressFormat | <p>Specifies where to place secondary address information for U.S. addresses. Secondary address information refers to apartment numbers, suite numbers, and similar designators. For example, in this address the secondary address information is "Apt 10E" and the primary address information is "424 Washington Blvd".</p> <p>Apt 10E 424 Washington Blvd Springfield MI 49423</p> <p>C Place both primary and secondary address information in AddressLine1 (default).</p> <p>S Place the primary address information in AddressLine1 and the secondary address information in AddressLine2.</p> <p>D Place both primary and secondary address information in AddressLine1 and place dropped information from dual addresses in AddressLine2. A dual address is an address that contains both street information and PO Box/Rural Route/Highway Contract information. For more information, see About Dual Address Logic on page 465.</p> |

| Parameter | Description |
|----------------------------|--|
| Option.OutputShortCityName | <p data-bbox="678 380 1419 464">Specifies how to format city names that have short city name or non-mailing city name alternatives. Applies to U.S. and Canadian addresses.</p> <ul style="list-style-type: none"> <li data-bbox="686 485 1419 667">Y Returns the USPS®-approved abbreviation for the city, if there is one. The USPS® provides abbreviations for city names that are 14 characters long or longer. City abbreviations are 13 characters or less and can be used when there is limited space on the mailing label. If there is no short city name for the city, then the full city name is returned. <li data-bbox="686 695 1122 722">N Returns the long city name (default). <li data-bbox="686 743 1419 894">S Returns the abbreviated city name only if an abbreviated city name is used in the input address. If the input address does not use a short city name, either the long or short city name could be returned, depending on USPS® regulations for the particular city. Select this option if you are performing a CASS™ test. <li data-bbox="686 921 1419 1073">V Output the non-mailing city name (the vanity name) if the input city name is a non-mailing city name. For example, "Hollywood" is a non-mailing city name for "Los Angeles". If you do not select this option and the input city name is a non-mailing city name the long version of the mailing city is returned. |
| Option.OutputCountryFormat | <p data-bbox="678 1161 1419 1245">Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <ul style="list-style-type: none"> <li data-bbox="686 1266 1187 1293">E Use English country names (default). <li data-bbox="686 1314 1097 1341">S Use Spanish country names. <li data-bbox="686 1362 1089 1390">F Use French country names. <li data-bbox="686 1411 1097 1438">G Use German country names. <li data-bbox="686 1459 1419 1514">I Use two-letter ISO abbreviation for the countries instead of country names. <li data-bbox="686 1535 1403 1587">U Use Universal Postal Union abbreviation for the countries instead of country names. |

| Parameter | Description |
|--------------------|---|
| Option.HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Canada, specify Canada. ValidateAddress uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent and the Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |

| Parameter | Description |
|-------------------------|---|
| Option.DualAddressLogic | <p>Indicates how to return a match if multiple non-blank address lines are present or multiple address types are on the same address line. (U.S. addresses only.)</p> <p>N (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority:</p> <ol style="list-style-type: none"> 1. PO Box 2. Firm 3. Highrise 4. Street 5. Rural Route 6. General Delivery <p>S Return a street match, regardless of the address line.</p> <p>P Return a PO Box match, regardless of the address line.</p> <p>For more information, see About Dual Address Logic on page 465.</p> |

About Dual Address Logic

For U.S. addresses only, the `Option.DualAddressLogic` option controls whether `ValidateAddress` should return a street match or a PO Box/Rural Route/Highway Contract match when the address contains both street and PO Box/Rural Route/Highway Contract information in the same address line.

Note: The `Option.DualAddressLogic` option has no effect if the street information is in a different address line input field than the PO Box/Rural Route/Highway Contract information.

For example, given the following input address:

```
AddressLine1: 401 N Main St Apt 1 POB 1
City: Kemp
StateProvince: TX
PostalCode: 75143
```

`ValidateAddress` would return one of the following:

- If `Option.DualAddressLogic` is set to either N or P:

```
AddressLine1: PO Box 1
City: Kemp
StateProvince: TX
PostalCode: 75143-0001
```

- If `Option.DualAddressLogic` is set to S:

AddressLine1: 401 N Main St Apt 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-4806

The address data that is not used to standardize the address can be returned in one of two places:

- **AddressLine2**—The address information not used to standardize the address is returned in the **AddressLine2** field if you specify `Option.StandardAddressFormat=D`. For more information, see [Default Options](#) on page 461. For example, if you choose to return a street match for dual addresses,

AddressLine1: 401 N Main St Apt 1
 AddressLine2: PO Box 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-0001

- **AdditionalInputData**—If you do not specify `Option.StandardAddressFormat=D` then the address information not used to standardize the address is returned in the **AdditionalInputData** field. For more information on this option, see [Default Options](#) on page 461. For example, if you choose to return a street match for dual addresses,

AddressLine1: 401 N Main St Apt 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-0001
 AdditionalInputData: PO Box 1

Address information that is dropped can be retrieved by setting the `Option.StandardAddressFormat` option to D. For more information, see [Default Options](#) on page 461 .

Returning Multiple Matches

If `ValidateAddress` finds multiple address in the postal database that are possible matches for the input address, you can have `ValidateAddress` return the possible matches. For example, the following address matches multiple addresses in the U.S. postal database:

PO BOX 1
 New York, NY

Options

To return multiple matches, use the options described in the following table.

Table 43: Multiple Match Option

| Parameter | Description |
|------------------------------------|---|
| Option.KeepMultimatch | <p>Indicates whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> |
| Option.MaximumResults | <p>A number between 1 and 10 that indicates the maximum number of addresses to return.</p> <p>The default value is 1.</p> <p>Note: The difference between Option.Keepmultimatch=N and Option.KeepMultimatch=Y/Option.MaximumResults=1 is that a multiple match will return a failure if Option.KeepMultimatch=N, whereas a multiple match will return one record if Option.KeepMultimatch=Y and Option.MaximumResults=1.</p> |
| Option.OutputFieldLevelReturnCodes | <p>To identify which output addresses are candidate addresses, you must specify a value of Y for Option.OutputFieldLevelReturnCodes. When you do this, records that are candidate addresses will have one or more "M" values in the field-level result indicators.</p> |

Output

When you choose to return multiple matches, the addresses are returned in the address format you specify. For information on specifying address format, see [Output Data Options](#) on page 452. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information, see [Field-Level Result Indicators](#) on page 504.

U.S. Address Options

| Parameter | Description |
|----------------------------|---|
| Option.PerformUSProcessing | <p>Specifies whether to process U.S. addresses. If you enable U.S. address processing ValidateAddress will attempt to validate U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>N No, do not process U.S. addresses.</p> <p>Y Yes, process U.S. addresses. Default.</p> |
| Option.Database.US | <p>Specifies which database to use for validating U.S. addresses. Only databases that have been defined in the US Database Resources panel in the Management Console are available.</p> |
| Option.PerformLOT | <p>Enhanced Line of Travel (eLOT) processing assigns a Line of Travel sequence code to your addresses. Note that addresses are not sorted into eLOT sequence but they are assigned a Line of Travel sequence code that allows you to sort addresses into eLOT sequence.</p> <p>To perform eLOT processing you must have the eLOT database installed.</p> <p>N No, do not perform Line of Travel Processing. Default.</p> <p>Y Yes, perform Line of Travel processing.</p> <p>For a listing of the output fields returned by this option, see Enhanced Line of Travel Output on page 518.</p> |

| Parameter | Description |
|-------------------|---|
| Option.PerformRDI | <p>Residential Delivery Indicator (RDI™) processing checks if an address is a residential address (not a business address). To perform RDI™ processing, you must have the RDI™ database installed.</p> <p>If you enable both DPV® and RDI™ processing, RDI™ information is only returned if the address is a valid delivery point. If DPV® does not validate the address no RDI™ data is returned.</p> <p>N No, do not perform Residential Delivery Indicator processing. Default.</p> <p>Y Yes, perform Residential Delivery Indicator processing.</p> |
| Option.PerformESM | <p>Enhanced Street Matching (ESM) applies additional matching logic to correct misspelled or complex street names and obtain a match. ESM enables more addresses to be validated but it reduces performance. You cannot perform ESM when ASM is enabled.</p> <p>N No, do not perform enhanced street matching. Default.</p> <p>Y Yes, perform enhanced street matching.</p> |
| Option.PerformASM | <p>All Street Matching (ASM) applies ESM processing as well as additional matching logic to correct errors in street names and obtain a match. It is effective at matching streets when the first letter of the street is incorrect. ASM provides the best address validation but reduces performance.</p> <p>N No, do not perform all street matching.</p> <p>Y Yes, perform all street matching. Default.</p> |
| Option.PerformDPV | <p>Delivery Point Validation (DPV®) validates that a specific address exists, as opposed to validating that a specific address is within a range of valid addresses. CMRA processing checks if an address is for a mailbox rented from a private company, referred to as a Commercial Mail Receiving Agent (CMRA).</p> <p>To perform DPV and CMRA processing, you must have the DPV database installed. The DPV database contains both DPV and CMRA data.</p> <p>N No, do not perform Delivery Point Validation or CMRA processing. Default.</p> <p>Y Yes, perform Delivery Point Validation and CMRA processing.</p> <p>For a listing of the output fields returned by this option, see DPV and CMRA Output on page 521.</p> |

| Parameter | Description |
|------------------------|---|
| Option.PerformLACSLink | <p>The USPS® Locatable Address Conversion System (LACS) allows you to correct addresses that have changed as a result of a rural route address converting to street-style address, a PO Box renumbering, or a street-style address changing. When enabled, LACS^{Link} processing is attempted for addresses that could not be validated, or addresses were validated and flagged for LACS^{Link} conversion.</p> <p>To perform LACS^{Link} processing, you must have the LACS^{Link} database installed.</p> <p>N No, do not attempt LACS^{Link} conversion. Default.</p> <p>Y Yes, attempt LACS^{Link} conversion.</p> <p>For a listing of the output fields returned by this option, see LACSLink Output on page 519</p> |
| Option.PerformEWS | <p>The Early Warning System (EWS) uses the USPS® EWS File to validate addresses that are not in the ZIP + 4® database.</p> <p>To perform EWS processing, you must have the EWS database installed.</p> <p>If an input address matches an address in the EWS file, the following record-level result indicators are returned:</p> <ul style="list-style-type: none"> • Status="F" • Status.Code="EWSFailure" • Status.Description="Address found in EWS table" <p>N No, do not perform EWS processing. Default.</p> <p>Y Yes, perform EWS processing.</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|--------------------|--|
| Option.ExtractFirm | |
|--------------------|--|

| Parameter | Description |
|-----------|--|
| Y | Yes, extract the firm name. |
| N | No, do not extract the firm name. Default. |
| | <p>To identify firm names in address lines, the address lines are scanned for keywords and patterns that identify which fields are address lines and which are FirmName lines. Since this is done based on patterns, fields may be misidentified. The following tips can help ensure optimal firm extraction:</p> <ul style="list-style-type: none"> • If possible, place the primary address elements in AddressLine1, the secondary elements in AddressLine2, Urbanization in AddressLine3, and firm in AddressLine4. If the address has no urbanization code, then place the firm name in AddressLine3 and leave AddressLine4 blank. For example, <p data-bbox="570 852 984 879">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 884 846 911">AddressLine2: Suite 600</p> <p data-bbox="570 915 841 942">AddressLine3: Precisely</p> <p data-bbox="570 947 829 974">AddressLine4: <blank></p> • When you define just two address lines, AddressLine2 is assigned to the secondary address most of the time. If you want to increase the chance that AddressLine2 will be treated as a firm name, put the firm name in AddressLine3 and leave AddressLine2 blank. • Numbers in a firm name (such as the "1" in "1 Stop Software") will increase the likelihood that the field will be treated as an address line. <p>Here are some examples of firm name extraction:</p> <ul style="list-style-type: none"> • In this example, AddressLine2 would get extracted into the FirmName output field <p data-bbox="570 1293 789 1320">FirmName: <blank></p> <p data-bbox="570 1325 1089 1352">AddressLine1: 4200 Parliament Place Suite 600</p> <p data-bbox="570 1356 1094 1383">AddressLine2: International Goose Feathers inc.</p> • In this example, AddressLine3 would get extracted into the FirmName output field. <p data-bbox="570 1457 789 1484">FirmName: <blank></p> <p data-bbox="570 1488 984 1516">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 1520 846 1547">AddressLine2: Suite 600</p> <p data-bbox="570 1551 841 1579">AddressLine3: Precisely</p> • In this example, AddressLine3 would be placed in the AdditionalInputData output field. The firm name would not be extracted because the FirmName input field is not blank. <p data-bbox="570 1705 1057 1732">FirmName: International Goose Feathers Inc.</p> <p data-bbox="570 1736 984 1764">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 1768 846 1795">AddressLine2: Suite 600</p> <p data-bbox="570 1799 841 1827">AddressLine3: Precisely</p> • In this example, no firm name would be extracted because there is only one |

| Parameter | Description |
|-------------------|--|
| | <p>non-blank address line, which is always treated as the primary address element.</p> <p>FirmName: <blank> AddressLine1: 4200 Parliament Place Suite 600</p> <ul style="list-style-type: none"> In this example, AddressLine2 would be treated as a secondary address element because the numeral "1" causes that field to be treated as a secondary address element. <p>FirmName: <blank> AddressLine1: 4200 Parliament Place Suite 600 AddressLine2: 1 Stop Software</p> |
| Option.ExtractUrb | <p>Specifies whether to extract the urbanization name from AddressLine1 through AddressLine4 and place it in the USUrbanName output field. This option works in cases where the input record's USUrbanName field is blank and there is more than one address line.</p> <p>Y Yes, extract the urbanization name.</p> <p>N No, do not extract the urbanization name. Default.</p> <p>To identify urbanization names, the address lines are scanned for keywords and patterns that identify which fields are address lines and which are urbanization name lines. Since this is done based on patterns, it is possible for fields to be incorrectly identified. To help ensure optimal urbanization extraction, place the primary address elements in AddressLine1, the secondary elements in AddressLine2, Urbanization in AddressLine3, and firm in AddressLine4, if possible. For example,</p> <p>AddressLine1: A1 Calle A AddressLine2: AddressLine3: URB Alamar AddressLine4: Precisely</p> |

| Parameter | Description |
|-------------------------|--|
| Option.PerformSuiteLink | <p data-bbox="548 380 1089 407">Specifies whether to perform Suite^{Link™} processing.</p> <p data-bbox="548 422 1425 541">Suite^{Link} corrects secondary address information for U.S. business addresses whose secondary address information could not be validated. If Suite^{Link} processing is enabled, the firm name is matched to a database of known firm names and their secondary address information.</p> <p data-bbox="548 556 688 583">For example,</p> <p data-bbox="548 604 964 724">Firm Name: Precisely Address Line 1: 4200 Parliament Place Address Line 2: STE 1 Postal Code: 20706</p> <p data-bbox="548 739 1317 766">In this case, Suite^{Link} processing would provide the correct suite number:</p> <p data-bbox="548 793 924 913">Firm Name: Precisely Address Line 1: 4200 Parliament PI Address Line 2: STE 500 Postal Code: 20706-1844</p> <p data-bbox="548 928 1414 955">To perform Suite^{Link™} processing, you must have the Suite^{Link™} database installed.</p> <p data-bbox="548 976 1024 1003">This option takes one of the following values:</p> <p data-bbox="557 1018 1078 1098">N No, do not use Suite^{Link™}. Default. Y Yes, use Suite^{Link™} processing.</p> <p data-bbox="548 1123 1401 1150">For a listing of fields returned by this option, see SuiteLink Output on page 523.</p> |

| Parameter | Description |
|-----------------------------|---|
| Option.OutputPreferredAlias | <p data-bbox="548 380 1214 407">Specifies whether to use a street's preferred alias in the output.</p> <p data-bbox="548 426 1425 485">Street name aliases in the United States are alternative names given to sections of a street. There are four types of street name aliases:</p> <ul data-bbox="548 504 1425 840" style="list-style-type: none"> <li data-bbox="548 504 1425 562">• Preferred—A preferred alias is the street name preferred locally. It typically applies only to a specific range of addresses on the street. <li data-bbox="548 569 1425 682">• Abbreviated—An abbreviated alias is a variation of the street name that can be used in cases where the length of AddressLine1 is longer than 31 characters. For example, the street name 1234 BERKSHIRE VALLEY RD APT 312A could be abbreviated to 1234 BERKSHIRE VLLY RD APT 312A. <li data-bbox="548 688 1425 781">• Changed—There has been an official street name change and the alias reflects the new name. For example if SHINGLE BROOK RD is changed to CANNING DR, then CANNING DR would be a changed alias type. <li data-bbox="548 787 1425 840">• Other—The street alias is made up of other names for the street or common abbreviations of the street. <p data-bbox="548 863 1305 890">The non-alias version of the street name is called the base street name.</p> <p data-bbox="548 909 1401 968">If the preferred alias is used in the input then the preferred alias will be the street name in the output regardless of whether you enable this option.</p> <p data-bbox="548 987 1414 1066">This is one of three options that control how ValidateAddress handles street name aliases. The other two are Option.OutputStreetNameAlias and Option.OutputAbbreviatedAlias.</p> <p data-bbox="548 1085 1425 1230">In most cases, if you select both Option.OutputPreferredAlias and Option.OutputAbbreviatedAlias, and ValidateAddress finds both a preferred and an abbreviated alias in the postal database, the abbreviated alias will be used in the output. The exception to this rule is if the input street name is a preferred alias. In this case, the preferred alias will be used in the output.</p> <p data-bbox="548 1249 1114 1276">Y Yes, perform preferred alias processing.</p> <p data-bbox="548 1295 1268 1323">N No, do not perform preferred alias processing. Default.</p> <p data-bbox="631 1350 1414 1432">Note: If the input address contains a street name alias of type "changed" the output address will always contain the base street name regardless of the options you specify.</p> |

| Parameter | Description |
|-------------------------------|---|
| Option.OutputAbbreviatedAlias | <p>Specifies whether to use a street's abbreviated alias in the output if the output address line is longer than 31 characters.</p> <p>This is one of three options that control how ValidateAddress handles street name aliases. The other two are Option.OutputStreetNameAlias and Option.OutputPreferredAlias.</p> <p>Note: If a preferred alias is specified in the input, the output street name will always be the preferred alias, even if you enable abbreviated street name alias processing.</p> <p>Y Yes, perform abbreviated alias processing.</p> <p>N No, do not perform abbreviated alias processing. Default.</p> <p>Note: If the input address contains a street name alias of type "changed" the output address will always contain the base street name regardless of the options you specify.</p> |
| Option.DPVDetermineNoStat | <p>Determines the "no stat" status of an address. An address is considered "no stat" if it exists but cannot receive mail, and therefore is not counted as a delivery statistic on a carrier's route (hence the term "no stat"). Examples include buildings under construction or those that the letter carrier has identified as not likely to receive mail.</p> <p>N No, do not determine "no stat" status. Default.</p> <p>Y Yes, determine "no stat" status.</p> <p>Note: You must enable DPV processing to use this option.</p> <p>The result is returned in the DPVNoStat field. For more information see LACSLink Output on page 519</p> |
| Option.DPVDetermineVacancy | <p>Determines if the location has been unoccupied for at least 90 days.</p> <p>N No, do not determine vacancy. Default.</p> <p>Y Yes, determine vacancy.</p> <p>Note: You must enable DPV processing to use this option.</p> <p>The result is returned in the DPVVacant field. For more information see LACSLink Output on page 519</p> |
| Option.ReturnVerimove | <p>Returns VeriMove detail data in output.</p> <p>N No, do not return VeriMove detail data. Default.</p> <p>Y Yes, return VeriMove detail data.</p> |

| Parameter | Description |
|--|--|
| Option.SuppressZipPlusPhantomCarrierR777 | <p>Specifies whether to suppress addresses with Carrier Route R777. These addresses are phantom routes and are not eligible for street delivery. Since these addresses are assigned a ZIP + 4[®] code by the USPS[®], Validate Address marks these addresses as deliverable. Select this option if you do not want addresses with Carrier Route R777 marked as deliverable. This will cause the following actions:</p> <ul style="list-style-type: none"> • No ZIP + 4 code is assigned • Address is not counted on the USPS Form 3553 (CASS Summary Report) • DPV Footnote of R7 is returned <p>N No, do not suppress addresses with Carrier Route R777.</p> <p>Y Yes, suppress addresses with Carrier Route R777.</p> |
| Option.StreetMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input street name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| Option.FirmMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input firm name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| Option.DirectionalityMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input directionals, such as the "N" in 123 N Main St., must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium". Default.</p> <p>L The matching algorithm is "loose."</p> |

| Parameter | Description |
|-------------------------------------|---|
| Option.DPVSuccessfulStatusCondition | <p>Select the match condition where a DPV result does NOT cause a record to fail.</p> <p>F Full match</p> <p>P Partial match</p> <p>A Always. Default.</p> <p>Note: You must enable DPV processing to use this option.</p> |
| Option.FailOnCMRAMatch | <p>Treat Commercial Mail Receiving Agency (CMRA) matches as failures?</p> <p>N No, do not treat CMRA matches as failures. Default.</p> <p>Y Yes, treat CMRA matches as failures.</p> <p>Note: You must enable DPV processing to use this option.</p> |
| Option.StandardAddressPMBLine | <p>Specifies where Private Mailbox (PMB) information is placed.</p> <p>N Do not include the PMB information in Standard Address output (default).</p> <p>1 Place the PMB information in AddressLine1. If you specify 1, you must set Option.StandardAddressFormat to either C or D.</p> <p>2 Place the PMB information in AddressLine2.</p> |
| Option.PreferredCity | <p>Specifies whether the preferred last line city name should be stored.</p> <p>Z Store the Preferred Last Line City Name from the USPS ZIP+4 File (Override City Name).</p> <p>Note: If you select this option, Validate Address generates a CASS-certified configuration and the USPS 3553 Report.</p> <p>C Store the USPS-preferred City Name from USPS City/State File.</p> <p>Note: If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.</p> <p>P Store the Primary City Name from the USPS City/State File.</p> <p>Note: If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.</p> |

CASS Certified Processing

CASS Certified™ processing also generates the USPS CASS Detailed Report, which contains some of the same information as the 3553 report but provides much greater detail about DPV, LACS, and SuiteLink statistics. The USPS CASS Detailed Report is not required for postal discounts and does not need to be submitted with your mailing.

1. Validate Address must be in CASS Certified™ mode. If **(Not CASS Certified)** appears at the top of the window, click the **Enable CASS** button. The **Enforce CASS rules** check box will appear.
2. Click **Configure CASS 3553**. The **CASS Report Fields** dialog box appears.
3. Type the **List Processor** company name, **List Name or ID#**, and the **Number of Lists** being processed for this job.
4. Type the **Mailer Name, Address, and City, State, ZIP**.
5. Click **OK**.

The List information will appear in Section B and the Mailer information in Section D of the generated USPS® CASS Form 3553.

6. In Enterprise Designer, drag **CASS3553** from the Reports pallet to the canvas.
7. Double-click the **CASS3553** icon on the canvas.
8. On the **Stages** tab, check the **Validate Address** check box. Note that if you have renamed the Validate Address stage to something else, you should check the box with the name you have given the address validation stage.
9. On the **Parameters** tab, select the format for the report. You can create the report in PDF, HTML, or plain text format.
10. Click **OK**.
11. Repeat steps 6-10 for **CASSDetail** if you want to produce the CASS Detail Report.

Canadian Address Options

| Parameter | Description |
|----------------------------------|---|
| Option.PerformCanadianProcessing | <p>Specifies whether to process Canadian addresses. If you enable Canadian address processing ValidateAddress will attempt to validate Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they is returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>N No, do not process Canadian addresses.</p> <p>Y Yes, process Canadian addresses (default).</p> |
| Option.Database.Canada | <p>Specifies which database you want to use for validating Canadian addresses. To specify a database for Canadian address validation, select a database in the Database drop-down list. Only databases that have been defined in the CAN Database Resources panel in the Management Console are available.</p> |

| Parameter | Description | | | | | | | | |
|---------------------------------|--|------------|----------------------------------|------------------|-------------------------------|--------------|---------------------------|-------------|--------------------------|
| Option.CanFrenchFormat | <p>Specifies how to determine the language (English or French) to use to format the address and directional. The following example shows an address formatted in English and French:</p> <p>English: 123 Main St W French: 123 Rue Main O</p> <p>The parameter controls the formatting of the address. It also affects the spelling of the directional but not spelling of the suffix.</p> <ul style="list-style-type: none"> C Use the street suffix returned by the matching process to determine the language. The street suffix returned by the matching process, which is used internally by ValidateAddress during processing, may be different from that in the input address. Ambiguous records are formatted like the input. Default. All addresses in Quebec are formatted using French. S Use the Canadian database to determine the language. The Canadian database contains data from the Canada Post Corporation (CPC). All addresses in Quebec are formatted using French. T Use the CanLanguage input field to determine the language. If there is a non-blank value in this field the address are formatted using French. | | | | | | | | |
| Option.CanEnglishApartmentLabel | <p>For English addresses, specifies the default apartment label to use in the output if there is no apartment label in the input address. This setting is ignored if you specify <code>Option.CanStandardAddressFormat=F</code>.</p> <table border="0"> <tr> <td data-bbox="813 1377 854 1404">Apt</td> <td data-bbox="1016 1377 1341 1404">Use "Apt" as the label. Default.</td> </tr> <tr> <td data-bbox="813 1425 935 1453">Apartment</td> <td data-bbox="1016 1425 1325 1453">Use "Apartment" as the label.</td> </tr> <tr> <td data-bbox="813 1474 870 1501">Suite</td> <td data-bbox="1016 1474 1268 1501">Use "Suite" as the label.</td> </tr> <tr> <td data-bbox="813 1522 862 1549">Unit</td> <td data-bbox="1016 1522 1252 1549">Use "Unit" as the label.</td> </tr> </table> | Apt | Use "Apt" as the label. Default. | Apartment | Use "Apartment" as the label. | Suite | Use "Suite" as the label. | Unit | Use "Unit" as the label. |
| Apt | Use "Apt" as the label. Default. | | | | | | | | |
| Apartment | Use "Apartment" as the label. | | | | | | | | |
| Suite | Use "Suite" as the label. | | | | | | | | |
| Unit | Use "Unit" as the label. | | | | | | | | |

| Parameter | Description |
|--------------------------------|---|
| Option.CanFrenchApartmentLabel | <p>For French addresses, specifies the default apartment label to use in the output if there is no apartment label in the input address. This setting is ignored if you specify <code>Option.CanStandardAddressFormat=F</code>.</p> <p>App Use "App" as the label. Default.</p> <p>Appartement Use "Appartement" as the label.</p> <p>Bureau Use "Bureau" as the label.</p> <p>Suite Use "Suite" as the label.</p> <p>Unite Use "Unite" as the label.</p> |
| Option.ForceCorrectionLVR | <p>Changes the civic and/or suite information to match the Large Volume Receiver (LVR) or single-single record (used when there is only one record for that postal code/street name/street type).</p> <p>N Do not change the civic and/or suite information to match the LVR or single-single record. The LVR record will be marked as a valid but non-correctable record (VN). The single-single record will be corrected, if possible, or processed as a non-correctable record..</p> <p>Y Change the civic and/or suite information to match the LVR or single-single record.</p> <p>Note: If you check this box, the Statement of Address Accuracy will not be printed because this is not a SERP-recognized setting.</p> |
| Option.CanPreferHouseNum | <p>In cases where the house number and postal code are both valid but in conflict, you can force the postal code to be corrected based on the house number by specifying <code>Option.CanPreferHouseNum=Y</code>. If you do not select this option the house number is changed to match the postal code.</p> <p>N Change the house number to match the postal code. Default.</p> <p>Y Change the postal code to match the house number.</p> |

| Parameter | Description |
|---------------------------------|---|
| Option.CanOutputCityAlias | <p>Specifies whether or not to return the city alias when the alias is in the input address. This option is disabled when you specify <code>Option.CanOutputCityFormat=D</code>.</p> <p>Y Output the city alias when the city alias is in the input. Default.</p> <p>N Never output the city alias even if it is in the input.</p> |
| Option.CanNonCivicFormat | <p>Specifies whether or not non-civic keywords are abbreviated in the output. For example, Post Office Box vs. PO Box.</p> <p>A Abbreviate non-civic keywords. Default.</p> <p>F Do not abbreviate non-civic keywords. The full keyword is used.</p> |
| Option.EnableSERP | <p>Specifies whether or not to use SERP options.</p> <p>Y Enable SERP options.</p> <p>N Do not enable SERP options. Default.</p> |
| Option.CanStandardAddressFormat | <p>Specifies where to place secondary address information in the output address. Secondary address information refers to apartment numbers, suite numbers, and similar designators.</p> <p>D Place apartment information in the location specified in the Default.</p> <p>B Place apartment information at the at the end of the AddressLine1 field.</p> <p>F Place the apartment number only (no label) at the beginning of the AddressLine1 field. For example, 400-123 Rue Main</p> <p>E Place the apartment number and label at the beginning of the AddressLine1 field. For example, Apt 400 123 Rue Main</p> <p>S Place apartment information on a separate line.</p> <p>S Place apartment information in the same location as the input address.</p> |

| Parameter | Description |
|----------------------------|---|
| Option.CanOutputCityFormat | <p>Specifies whether to use the long, medium, or short version of the city if the city has a long name. For example,</p> <p>Long: BUFFALO HEAD PRAIRIE Medium: BUFFALO-HEAD-PR Short: BUFFALO-HD-PR</p> <p>D Use the default option specified by the <code>Option.OutputShortCityName</code> parameter. Default. If you specify <code>Option.OutputShortCityName=V</code>, the city is formatted as if you select for this option (see below) and <code>Y</code> for CanOutputCityAlias.</p> <p>S Output short city name.</p> <p>L Output the long city name.</p> <p>M Output the medium city name.</p> <p>I Use the same city format as used in the input address. Output is L, M, or S.</p> |
| Option.CanRuralRouteFormat | <p>Specifies where to place rural route delivery information. An example of an address with rural route delivery information is:</p> <p>36 GRANT RD RR 3 ANTIGONISH NS</p> <p>In this address, "RR 3" is the rural route delivery information.</p> <p>A Place rural route delivery information on the same line as the address, after the address information. Default. For example,</p> <p>36 GRANT RD RR 3</p> <p>S Place rural route delivery information on a separate address line. For example,</p> <p>36 GRANT RD RR 3</p> |

| Parameter | Description |
|--------------------------------|---|
| Option.CanDeliveryOfficeFormat | <p data-bbox="805 380 1425 436">Specifies where to place station information. An example of an address with station information is:</p> <p data-bbox="805 457 1032 514">PO BOX 8625 STN A ST. JOHN'S NL</p> <ul data-bbox="805 535 1425 898" style="list-style-type: none"><li data-bbox="805 535 1425 592">I Place station information in the same location as it is in the input address. Default.<li data-bbox="805 613 1425 724">A Place station information on the same line as the address, after the address information. For example, PO BOX 8625 STN A<li data-bbox="805 745 1425 898">S Place station information on a separate address line. For example, PO BOX 8625 STN A |

| Parameter | Description |
|---|--|
| Option.CanDualAddressLogic | <p>Specifies whether ValidateAddress should return a street match or a PO Box/non-civic match when the address contains both civic and non-civic information. One of the following:</p> <p>D Use DualAddressLogic Global Option. Default.</p> <p>P Match to PO Box or other non-street data.</p> <p>S Match to street.</p> <p>For example, given the following input address:</p> <p>AddressLine1: 36 GRANT RD AddressLine2: RR 4 City: ANTIGONISH StateProvince: NS</p> <p>ValidateAddress would return one of the following:</p> <ul style="list-style-type: none"> • If <code>Option.CanDualAddressLogic</code> is set to S, ValidateAddress returns the following: <p>AddressLine1: 36 GRANT RD AddressLine2: RR 3 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L1</p> • If <code>Option.CanDualAddressLogic</code> is set to P, ValidateAddress returns the following: <p>AddressLine1: RR 4 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L2</p> <p>The address data that is not used to standardize the address is returned in the AdditionalInputData field. For more information, see Output Data Options on page 452.</p> |
| Option.canSwitchManagedPostalCodeConfidence | <p>Select this check-box to lower the output Confidence score of the record. This is useful for the non-correctable (VN) records. You can identify such records by the DefaultValidPostalCode value. It is Y for the VN type of records and blank for others.</p> |

SERP Processing

1. Validate Address must be in SERP Certified™ mode. If **(Not SERP Certified)** appears at the top of the window, click the **Enable SERP settings** button. The **Configure SERP** box will appear.
2. Click **Configure SERP**. The **SERP Report Fields** dialog box appears.
3. Type your merchant **CPC number**.
4. Type the mailer **Name, Address, and City, State, ZIP**.
5. Click **OK**.
6. In Enterprise Designer, drag **SERPReport** from the Reports pallet to the canvas.
7. Double-click the **SERPReport** icon on the canvas.
8. On the **Stages** tab, ensure that the **Validate Address** check box is checked. Note that if you have renamed the Validate Address stage to something else, you should check the box with the name you have given the address validation stage.
9. On the **Parameters** tab, select the format for the report. You can create the report in PDF, HTML, or plain text format. PDF format is the default.
10. Click **OK**.

Obtaining SERP Return Codes

SERP return codes indicate the quality of the input address as determined by the Canada Post's Software Evaluation and Recognition Program regulations.

To obtain SERP return codes, specify `Option.OutputRecordType=P`. For more information on `Option.OutputRecordType`, see [Output Data Options](#) on page 452.

SERP return codes are provided in the following output field.

Table 44: SERP Return Code Output

| Response Element | Description |
|------------------|--|
| CanadianSERPCode | <p>Validation/correction return code (Canadian addresses only):</p> <p>V The input was valid. Canada Post defines a "valid" address as an address that meets all the following requirements:</p> <p style="padding-left: 40px;">Note: There are exceptions. For further information, contact the CPC.</p> <ul style="list-style-type: none"> • The address must contain all required components as found in CPC's Postal Code Data Files. • The address must provide an exact match on all components for only one address in CPC's Postal Code Data Files, allowing for acceptable alternate words and names listed in the CPC Postal Code Data Files. • Address components must be in a form that allows recognition without ambiguity. Certain components may require "qualifiers" to identify them. For instance, a Route Service address requires the key words "Rural Route" or "RR" for differentiation from a "Suburban Service" or "SS" address with the same number. <p>I The input was invalid. An "invalid" address is one that does not meet CPC requirements for a valid address (see above). Examples of this include address components that are missing, invalid, or inconsistent.</p> <p>C The input was correctable. A "correctable" address is one that can be corrected to match one, and only one, address.</p> <p>N The input was non-correctable. A "non-correctable" address is one that could be corrected a number of different ways such that ValidateAddress cannot identify a single correct version.</p> <p>F The input address was foreign (outside of Canada).</p> |

International Address Options

Addresses outside of the U.S. and Canada are referred to as "international" addresses. The following options control international address processing:

| Parameter | Description |
|---------------------------------------|--|
| Option.PerformInternationalProcessing | <p data-bbox="805 380 1425 730">Specifies whether to process international addresses (addresses outside the U.S. and Canada). If you enable international address processing <code>ValidateAddress</code> will attempt to validate international addresses. If you disable international address processing, international addresses will fail, meaning they is returned with an "F" in the Status output field. The output field <code>Status.Code</code> will say "DisabledCoder." If you are not licensed for international address processing you must disable international address processing in order for your jobs to complete successfully, regardless of whether or not they contain international addresses.</p> <p data-bbox="889 751 1425 926">Note: You must have a valid license for international address processing to successfully process international addresses. If you enable international address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p data-bbox="813 957 1370 984">N No, do not process international addresses.</p> <p data-bbox="813 1010 1398 1037">Y Yes, process international addresses (default).</p> |
| Option.Database.International | <p data-bbox="805 1121 1425 1295">Specifies which database you want to use for validating international addresses. To specify a database for international address validation, select a database in the Database drop-down list. Only databases that have been defined in the INTL Database Resources panel in the Management Console are available.</p> |

| Parameter | Description |
|---|---|
| Option.InternationalCityStreetSearching | <p>By default, ValidateAddress provides a balance of good address matching accuracy with good performance. If you are willing to trade matching accuracy for faster performance, use the Option.InternationalCityStreetSearching parameter to increase processing speed. When you do this, some accuracy is lost. This option only controls performance for addresses outside the U.S. and Canada. This setting affects a small percentage of records, mostly addresses in the U.K. There is no performance control for U.S. and Canadian address processing.</p> <p>If you use GetCandidateAddresses, the candidate addresses returned by GetCandidateAddresses may differ from the multiple matches returned by ValidateAddress if you set the performance tuning option for international addresses to any value other than 100.</p> <p>To control performance, specify a value from 0 to 100. A setting of 100 maximizes accuracy while a setting of 0 maximizes speed. The default is 100.</p> |
| Option.AddressLineSearchOnFail | <p>This option enables ValidateAddress to search the AddressLine input fields for the city, state/province, postal code, and country when the address cannot be matched using the values in the City, StateProvince, and PostalCode input fields.</p> <p>Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields.</p> <p>Consider disabling this option if your input addresses use the City, State/Province and PostalCode fields. If you enable this option and these fields are used, there is an increased possibility that ValidateAddress will fail to correct values in these fields (for example a misspelled city name).</p> <p>N No, do not search the AddressLine fields.</p> <p>Y Yes, search the address line fields. Default.</p> |

Response

The output from ValidateAddress contains different information depending on the output categories you select.

Standard Address Output

Standard address output consists of four lines of the address which correspond to how the address would appear on an address label. City, state/province, postal code, and other data is also included in standard address output. Standard address output is returned for validated addresses if you set `Option.OutputRecordType=A`. Standard address fields are always returned for addresses that could not be validated. For non-validated addresses, the standard address output fields contain the address as it appeared in the input ("pass through" data). If you want addresses to be standardized according to postal authority standards when validation fails, specify `Option.OutputFormattedOnFail=Y` in your request.

Table 45: Standard Address Output

| Response Element | Description |
|---------------------|--|
| AdditionalInputData | Input data not used by the address validation process. For more information, see About Additional Input Data . |
| AddressLine1 | If the address was validated, the first line of the validated and standardized address. If the address could not be validated, the first line of the input address without any changes. |
| AddressLine2 | If the address was validated, the second line of the validated and standardized address. If the address could not be validated, the second line of the input address without any changes. |
| AddressLine3 | If the address was validated, the third line of the validated and standardized address. If the address could not be validated, the third line of the input address without any changes. |
| AddressLine4 | If the address was validated, the fourth line of the validated and standardized address. If the address could not be validated, the fourth line of the input address without any changes. |
| AddressLine5 | For U.K. addresses only. If the address was validated, the fifth line of the validated and standardized address. If the address could not be validated, the fifth line of the input address without any changes. |
| City | The validated city name. |

| Response Element | Description |
|------------------------|--|
| Country | <p>The country in the format determined by what you selected in Data.OutputCountryFormat:</p> <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| DepartmentName | For U.K. addresses only, a subdivision of a firm. For example, Engineering Department. |
| FirmName | The validated firm or company name. |
| PostalCode | The validated ZIP Code™ or postal code. |
| PostalCode.AddOn | The 4-digit add-on part of the ZIP Code™. For example, in the ZIP Code™ 60655-1844, 1844 is the 4-digit add-on. (U.S. addresses only.) |
| PostalCode.Base | The 5-digit ZIP Code™; for example 20706 (U.S. addresses only). |
| StateProvince | The validated state/province or its abbreviated value. |
| USUrbanName | The validated urbanization name. (U.S. addresses only.) This is used primarily for Puerto Rico addresses. |
| DefaultValidPostalCode | <p>This field is generated only for Canadian addresses.</p> <p>A value of Y indicates that the record is non-correctable (VN) type. In such cases, you have the option of lowering the output Confidence score for the record. To do this, select the Switch default valid postal code confidence check-box in the Input Options. For more information, see section Canadian Address Options on page 480.</p> <p>Note: For all other records, the field value is blank.</p> |

Parsed Address Elements Output

Output addresses are formatted in the parsed address format if you set `Option.OutputRecordType=E`. If you want formatted data in the Parsed Address format to be returned when validation fails (that is, a normalized address), specify `Option.OutputFormattedOnFail=Y`.

Note: If you always want parsed input data returned regardless of whether or not validation is successful, specify `Option.OutputRecordType=I`. For more information, see [Parsed Input](#) on page 569.

Table 46: Parsed Address Output

| Response Element | Description |
|--|--|
| <code>AdditionalInputData</code> | Input data not used by <code>ValidateAddress</code> . For more information, see Additional Input Data on page 524. |
| <code>AdditionalInputData.Base</code> | Input data that was not output to the standardized address by <code>ValidateAddress</code> . For more information, see Additional Input Data on page 524. |
| <code>AdditionalInputData.Unmatched</code> | Input data passed to the matcher but not used by <code>ValidateAddress</code> for validation. For more information, see Additional Input Data on page 524. |
| <code>ApartmentLabel</code> | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| <code>ApartmentLabel2</code> | Secondary apartment designator, for example: 123 E Main St APT 3, 4th Floor Note: In this release, this field will always be blank. |
| <code>ApartmentNumber</code> | Apartment number. For example: 123 E Main St APT 3 |

| Response Element | Description |
|---|--|
| ApartmentNumber2 | <p>Secondary apartment number. For example: 123 E Main St APT 3, 4th Floor</p> <p>Note: In this release, this field will always be blank.</p> |
| CanadianDeliveryInstallationAreaName | Delivery installation name (Canadian addresses only) |
| CanadianDeliveryInstallationQualifierName | Delivery installation qualifier (Canadian addresses only) |
| CanadianDeliveryInstallationType | Delivery installation type (Canadian addresses only) |
| City | Validated city name |
| Country | <p>Country. Format is determined by what you selected in <code>Data.OutputCountryFormat</code>:</p> <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| DepartmentName | For U.K. addresses only, a subdivision of a firm. For example, Engineering Department. |
| FirmName | The validated firm or company name |
| HouseNumber | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional | Leading directional, for example: 123 E Main St Apt 3 |

| Response Element | Description |
|---------------------|--|
| POBox | Post office box number. If the address is a rural route address, the rural route box number will appear here. |
| PostalCode | Validated postal code. For U.S. addresses, this is the ZIP Code. |
| PrivateMailbox | Private mailbox indicator. |
| PrivateMailbox.Type | The type of private mailbox. Possible values include: <ul style="list-style-type: none"> • Standard • Non-Standard <p>Note: This replaces PrivateMailboxType (no period in field name). Please modify your API calls accordingly.</p> |
| RRHC | Rural Route/Highway Contract indicator |
| StateProvince | Validated state or province name |
| StreetName | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix | Street suffix, for example: 123 E Main St Apt 3 |
| TrailingDirectional | Trailing directional, for example: 123 Pennsylvania Ave NW |
| USUrbanName | USPS® urbanization name. Puerto Rican addresses only. |

Parsed Input

The output can include the input address in parsed form. This type of output is referred to as "parsed input." Parsed input fields contain the address data that was used as input regardless of whether or not ValidateAddress validated the address. Parsed input is different from the "parsed address

elements" output in that parsed address elements contain the validated address if the address could be validated, and, optionally, the input address if the address could not be validated. Parsed input always contains the input address regardless of whether or not `ValidateAddress` validated the address.

To include parsed input fields in the output, set `Option.OutputRecordType=I`.

Table 47: Parsed Input

| Response Element | Description |
|--|--|
| <code>ApartmentLabel.Input</code> | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| <code>ApartmentNumber.Input</code> | Apartment number, for example: 123 E Main St APT 3 |
| <code>CanadianDeliveryInstallationAreaName.Input</code> | Delivery installation name (Canadian addresses only) |
| <code>CanadianDeliveryInstallationQualifierName.Input</code> | Delivery installation qualifier (Canadian addresses only) |
| <code>CanadianDeliveryInstallationType.Input</code> | Delivery installation type (Canadian addresses only) |
| <code>City.Input</code> | Validated city name |
| <code>Country.Input</code> | Country. Format is determined by what you selected in <code>Data.OutputCountryFormat</code> : <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| <code>FirmName.Input</code> | The validated firm or company name |
| <code>HouseNumber.Input</code> | House number, for example: 123 E Main St Apt 3 |

| Response Element | Description |
|---------------------------|--|
| LeadingDirectional.Input | Leading directional, for example: 123 E Main St Apt 3 |
| POBox.Input | Post office box number. If the address is a rural route address, the rural route box number will appear here. |
| PostalCode.Input | Validated postal code. For U.S. addresses, this is the ZIP Code. |
| PrivateMailbox.Input | Private mailbox indicator |
| PrivateMailbox.Type.Input | The type of private mailbox. Possible values include: <ul style="list-style-type: none"> • Standard • Non-Standard |
| RRHC.Input | Rural Route/Highway Contract indicator |
| StateProvince.Input | Validated state or province name |
| StreetName.Input | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix.Input | Street suffix, for example: 123 E Main St Apt 3 |
| TrailingDirectional.Input | Trailing directional, for example: 123 Pennsylvania Ave NW |
| USUrbanName.Input | USPS® urbanization name |

Postal Data Output

If `Option.OutputRecordType` contains **P** then the following fields are returned in the output.

Table 48: Postal Data Output

| Response Element | Description |
|-------------------------|---|
| CanadianSERPCode | Validation/correction return code (Canadian addresses only). For more information, see Obtaining SERP Return Codes on page 487. |
| IntHexaviaCode | For addresses in France only, a numeric code that represents the street. For information about Hexavia codes, see www.laposte.fr . |
| IntINSEECode | For addresses in France only, a numeric code that represents the city. For a listing of INSEE codes, see www.insee.fr . |
| PostalBarCode | The two-digit delivery point portion of the delivery point barcode (U.S. addresses only). For more information, see Creating Delivery Point Barcodes on page 460. |
| USAltAddr | Indicates whether or not alternate address matching logic was used, and if so which logic was used (U.S. addresses only). One of the following: <ul style="list-style-type: none"> null No alternate address scheme used. D Delivery point alternate logic was used. E Enhanced highrise alternate match logic was used. S Small town default logic was used. U Unique ZIP Code logic was used. |
| USBCCheckDigit | Check-digit portion of the 11-digit delivery point barcode (U.S. addresses only). For more information, see Creating Delivery Point Barcodes on page 460. |
| USCarrierRouteCode | Carrier route code (U.S. addresses only). For more information, see Obtaining Carrier Route Codes on page 459. |
| USCongressionalDistrict | Congressional district (U.S. addresses only). For more information, see Obtaining Congressional Districts on page 458. |
| USCountyName | County name (U.S. addresses only). For more information, see Obtaining County Names on page 459. |

| Response Element | Description |
|--------------------|--|
| USFinanceNumber | The finance number in which the address resides (U.S. addresses only). The finance number is a number assigned by the USPS to an area that covers multiple ZIP Codes. An address is validated only if its finance number matches the finance number of the candidate address in the U.S. Database. |
| USFIPSCountyNumber | FIPS (Federal Information Processing Standards) county number (U.S. addresses only). For more information, see Obtaining FIPS County Numbers on page 459. |
| USLACS | <p>Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following:</p> <p>Y Yes, the address is a candidate for LACS^{Link} processing. If LACS^{Link} is enabled, an attempt is made to convert the address using the LACS^{Link} database. If the conversion attempt is successful, the output address is the new address obtained from the LACS^{Link} database. If the attempt is not successful, the address will not be converted.</p> <p>N No, the address is not a candidate for LACS^{Link} processing. LACS^{Link} processing may still be attempted if LACS^{Link} processing is requested, the LACS^{Link} database is installed, and one of the following is true:</p> <ul style="list-style-type: none"> • The address matches to a Rural Route address and the RecordType.Default field returns a Y. • The input address could not be matched to any address in the U.S. Postal Database (Failures due to multiple matches are not LACS^{Link} candidates.) |
| USLastLineNumber | <p>A six-character alphanumeric value that groups together ZIP Codes that share the same primary city. For example, addresses with the following two last lines would have the same last line number:</p> <p>Chantilly VA 20151 Chantilly VA 20152</p> |

Result Indicators

Result indicators provide information about the kinds of processing performed on an address. There are two types of result indicators:

Record-Level Result Indicators

Record-level result indicators provide data about the results of ValidateAddress processing for each record, such as the success or failure of the match attempt, which coder processed the address,

and other details. The following table lists the record-level result indicators returned by `ValidateAddress`.

Table 49: Record Level Indicators

| Response Element | Description |
|------------------|--|
| AddressFormat | <p>The type of address data being returned:</p> <p>F French format (for example: 123 Rue Main)</p> <p>E English format (for example: 123 Main St)</p> |
| Confidence | <p>The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct. For multiple matches, the confidence level is 0. For details about how this number is calculated, see Introduction to the Validate Address Confidence Algorithm on page 1036.</p> |
| CouldNotValidate | <p>If no match was found, which address component could not be validated:</p> <ul style="list-style-type: none"> • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber • RuralRoute <p>Note: More than one component may be returned, in a comma-separated list.</p> |

| Response Element | Description |
|------------------|--|
| CountryLevel | <p>The category of address matching available. This is always "A" for U.S. and Canadian addresses. One of the following:</p> <p>A The address is in a country for which there is highly detailed postal data available. Addresses in this match level can have the following address elements validated and corrected, and added if missing from the input:</p> <ul style="list-style-type: none">• Postal code• City name• State/county name• Street address elements• Country name <p>B The address is in a country for which there is a medium level of postal data available. Addresses in this match level can have the following address elements validated and corrected, and added if missing from the input:</p> <ul style="list-style-type: none">• Postal code• City name• State/county name• Country name <p>C The address is in a country for which the postal data is least detailed. Addresses in this match level can have the following actions performed on them:</p> <ul style="list-style-type: none">• Validate and correct country name (cannot supply missing country name)• Validate the format of the postal code (cannot supply missing postal code or validate the code) |

| Response Element | Description |
|------------------|---|
| MatchScore | <p>MatchScore provides an indication of the degree to which the output address is correct. It is significantly different from Confidence in that Confidence indicates how much the input address changed to obtain a match, whereas the meaning of Match Score varies between U.S. and non-U.S. addresses.</p> <p>For U.S. addresses, MatchScore is a one-digit score on a scale of 0 to 9 that reflects the closeness of the street-name match (after transformations by ValidateAddress, if any). Zero indicates an exact match and 9 indicates the least likely match. If no match was found, this field is blank.</p> <p>For non-U.S. and non-Canadian addresses, MatchScore is a five-digit score, with a maximum value of 00999. Higher numbers indicates a closer match.</p> <p>This field does not apply to Canadian addresses.</p> <p>Note that you cannot equate match scores from U.S. addresses with those of non-U.S. addresses. For example, a match score of 4 for a U.S address does not indicate the same level of match as a 00004 for a non-U.S. address.</p> <p>Note: The Validate Address and Advanced Matching Module components both use the MatchScore field. The MatchScore field value in the output of a dataflow is determined by the last stage to modify the value before it is sent to an output stage. If you have a dataflow that contains Validate Address and Advanced Matching Module components and you want to see the MatchScore field output for each stage, use a Transformer stage to copy the MatchScore value to another field. For example, Validate Address produces an output field called MatchScore and then a Transformer stage copies the MatchScore field from Validate Address to a field called AddressMatchScore. When the matcher stage runs it populates the MatchScore field with the value from the matcher and passes through the AddressMatchScore value from Validate Address.</p> |
| MultimatchCount | If multiple matches were found, indicates the number of records that are possible matches. |
| MultipleMatches | <p>Indicates which address component had multiple matches, if multiple matches were found:</p> <ul style="list-style-type: none"> • Firm • LeadingDirectional • PostalCode • StreetName • StreetSuffix • TrailingDirectional • Urbanization <p>Note: More than one component may be returned, in a comma-separated list.</p> |

| Response Element | Description | | | | | | |
|---------------------|--|-------------|---------------------------------------|-------------|--|------------|-----------------------------|
| ProbableCorrectness | <p>Indicates the accuracy of a match on the scale of 0 to 9. The result can be:</p> <ul style="list-style-type: none"> • <i>Blank</i> - No Match Found • 0 - Most likely to be correct (Exact Match) • 1-8 - Intermediate probability of being correct • 9 - Match least likely to be correct | | | | | | |
| ProcessedBy | <p>Which address coder processed the address:</p> <table> <tr> <td>USA</td> <td>U.S. address coder</td> </tr> <tr> <td>CAN</td> <td>Canadian address coder</td> </tr> <tr> <td>INT</td> <td>International address coder</td> </tr> </table> | USA | U.S. address coder | CAN | Canadian address coder | INT | International address coder |
| USA | U.S. address coder | | | | | | |
| CAN | Canadian address coder | | | | | | |
| INT | International address coder | | | | | | |
| RecordType | <p>Type of address record, as defined by U.S. and Canadian postal authorities (supported for U.S. and Canadian addresses only):</p> <ul style="list-style-type: none"> • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal | | | | | | |
| RecordType.Default | <p>Code indicating the "default" match:</p> <table> <tr> <td>Y</td> <td>The address matches a default record.</td> </tr> <tr> <td>null</td> <td>The address does not match a default record.</td> </tr> </table> | Y | The address matches a default record. | null | The address does not match a default record. | | |
| Y | The address matches a default record. | | | | | | |
| null | The address does not match a default record. | | | | | | |
| Status | <p>Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches.</p> <table> <tr> <td>null</td> <td>Success</td> </tr> <tr> <td>F</td> <td>Failure</td> </tr> </table> | null | Success | F | Failure | | |
| null | Success | | | | | | |
| F | Failure | | | | | | |

| Response Element | Description |
|--------------------|---|
| Status.Code | Reason for failure, if there is one. For multiple matches, all possible matches is "MultipleMatchesFound." <ul style="list-style-type: none"> • DisabledCoder • InsufficientInputData • MultipleMatchesFound • UnableToValidate |
| Status.Description | Description of the problem, if there is one. Possible Multiple Addresses Found This value will appear if Status.Code=MultipleMatchesFound. Address Not Found This value will appear if Status.Code=UnableToValidate. PerformUSProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformCanadianProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformInternationalProcessing disabled This value will appear if Status.Code=DisabledCoder. |

Field-Level Result Indicators

Field-level result indicators describe how ValidateAddress handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in **HouseNumber.Result**.

To enable field-level result indicators, specify `Option.OutputFieldLevelReturnCodes=Y`. For more information, see [Output Data Options](#) on page 452.

The following table lists the field-level result indicators. If a particular field does not apply to an address, the result indicator may be blank.

Table 50: Field-Level Result Indicators

| Response Element | Description |
|-----------------------|--|
| AddressRecord.Result | <p>These result codes apply to international addresses only.</p> <ul style="list-style-type: none"> <li data-bbox="813 480 1421 569">M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. <li data-bbox="813 594 1421 646">S Standardized. This option includes any standard abbreviations. <li data-bbox="813 672 1019 697">U Unmatched. <li data-bbox="813 722 1421 779">V Validated. The data was confirmed correct and remained unchanged from input. |
| ApartmentLabel.Result | <ul style="list-style-type: none"> <li data-bbox="813 867 1421 924">A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. <li data-bbox="813 949 1377 974">C Corrected. U.S. and Canadian addresses only. <li data-bbox="813 999 1421 1087">D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data. <li data-bbox="813 1113 1421 1192">F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. <li data-bbox="813 1218 1421 1306">P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. <li data-bbox="813 1331 1421 1388">R The apartment label is required but is missing from the input address. U.S. addresses only. <li data-bbox="813 1413 1421 1465">S Standardized. This option includes any standard abbreviations. <li data-bbox="813 1491 1421 1516">U Unmatched. Does not apply to Canadian addresses. <li data-bbox="813 1541 1421 1591">V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|------------------------|--|
| ApartmentNumber.Result | <ul style="list-style-type: none">A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.C Corrected. Canadian addresses only.D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. addresses that are an EWS match will have a value of P. U.S. and Canadian addresses only.R The apartment number is required but is missing from the input address. U.S. addresses only.S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.U Unmatched.V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|------------------|--|
| City.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>F Hyphens missing or punctuation errors. Canadian addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R The city is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| Country.Result | <p>These result codes do not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|--------------------|--|
| FirmName.Result | <p>C Corrected. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only.</p> <p>U Unmatched. U.S. and Canadian addresses only.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. U.S. addresses only.</p> |
| HouseNumber.Result | <p>A Appended. The field was added to a blank input field. Canadian addresses only.</p> <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>O Out of range. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The house number is required but is missing from the input address. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|---------------------------|--|
| LeadingDirectional.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. Non-blank input was corrected to a non-blank value. U.S. addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. Does not apply to Canadian addresses.</p> |

| Response Element | Description |
|------------------|---|
| POBox.Result | <p>A Appended. The field was added to a blank input field. Canadian addresses only.</p> <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The P.O. Box number is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> <p>Blank Not applicable.</p> |

| Response Element | Description |
|-----------------------|--|
| PostalCode.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses.</p> <p>R The postal code is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses.</p> <p>U Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| PostalCodeCity.Result | <p>These result codes apply to international addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|-------------------|---|
| PostalCode.Source | <p>These result codes apply to U.S. addresses only.</p> <p>FinanceNumber The ZIP Code™ in the input was verified by using USPS® Finance Number groupings.</p> <p>ZIPMOVE The ZIP Code™ in the input address was corrected because the USPS® redrew ZIP Code™ boundaries and the address is now in a different ZIP Code™.</p> |
| PostalCode.Type | <p>P The ZIP Code™ contains only PO Box addresses. U.S. addresses only.</p> <p>U The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.</p> <p>M The ZIP Code™ is for military addresses. U.S. addresses only.</p> <p>null The ZIP Code™ is a standard ZIP Code™.</p> |
| RRHC.Result | <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The rural route/highway contract is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only.</p> <p>U Unmatched. U.S. and Canadian addresses only.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. U.S. and Canadian addresses only.</p> |

| Response Element | Description |
|----------------------|--|
| RRHC.Type | <p>These result codes apply to U.S. addresses only.</p> <p>HC The address is a Highway Contract address.</p> <p>RR The address is a Rural Route address.</p> |
| StateProvince.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R The state is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| Street.Result | <p>These result codes apply to international addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R Street corrected. House number is out of range. Applies to French, UK, and Japanese records only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|------------------------------------|---|
| StreetName.AbbreviatedAlias.Result | <p>Indicates the result of abbreviated alias processing. One of the following:</p> <ul style="list-style-type: none"> null No abbreviated alias processing attempted. B The StreetName field contains the base street name. L The standardized address length is less than 31 characters so the StreetName field contains the base name. N No abbreviated alias found. Y An abbreviated alias was found for input address. The StreetName field contains the abbreviated alias. |
| StreetName.Alias.Type | <p>This result code applies to U.S. addresses only.</p> <p>Note: In previous releases this field was named StreetName.AliasType with no "." between "Alias" and "Type." This old name is obsolete. Please update your processes to use the new name StreetName.Alias.Type.</p> <ul style="list-style-type: none"> Abbreviated The alias is an abbreviation of the street name. For example, HARTS-NM RD is an abbreviated alias for HARTSVILLE NEW MARLBORO RD. Changed There has been an official street name change and the alias reflects the new name. For example if SHINGLE BROOK RD is changed to CANNING DR, then CANNING DR would be a changed alias type. Other The street alias is made up of other names for the street or common abbreviations of the street. Preferred The street alias is the locally preferred alias. For example, a street is named "South Shore Dr." because it runs along the southern shore of a lake, not because it is south of a municipal demarcation line. So, "South" is not a predirectional in this case and should not be shorted to "S". So, "South Shore Dr." would be the preferred alias. |

| Response Element | Description |
|----------------------------------|--|
| StreetName.PreferredAlias.Result | <p>Indicates the result of preferred alias processing. One of the following:</p> <ul style="list-style-type: none"> null No preferred alias processing attempted. A Preferred alias processing was not attempted because the input address matched to an alias. Preferred alias processing is only attempted for base addresses. N No preferred alias found. Y A preferred alias was found for the input address. The StreetName field contains the preferred alias. |
| StreetName.Result | <ul style="list-style-type: none"> A Appended. The field was added to a blank input field. Canadian addresses only. C Corrected. U.S. and Canadian addresses only. D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data. F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only. U Unmatched. V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|---------------------|---|
| StreetSuffix.Result | <ul style="list-style-type: none">A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.C Corrected. U.S. and Canadian addresses only.D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data.F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.S Standardized. This option includes any standard abbreviations.U Unmatched. Does not apply to U.S. addresses.V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|----------------------------|--|
| TrailingDirectional.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| USUrbanName.Result | <p>These result codes apply to U.S. addresses only.</p> <p>A Appended. The field was added to a blank input field.</p> <p>C Corrected.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

Output from Options

ValidateAddress returns additional data depending on the options you select. For information on the output generated by each option, see the options listed in the following sections:

Enhanced Line of Travel Output

Enhanced Line of Travel processing produces the following output.

| Response Element | Description |
|------------------|--|
| USLOTCode | <p>Line of Travel sequence code and an indicator denoting USPS® LOT sequence. This field is in the format nnnnY where:</p> <p>nnnn The four-digit LOT code.</p> <p>Y One of the following:</p> <ul style="list-style-type: none"> • A—Ascending LOT sequence • D—Descending LOT sequence |
| USLOTHex | A hexadecimal value that allows you to sort your file in ascending order only. The hexadecimal values range from 0 to FF ascending, then FF through 0 descending. |
| USLOTSquence | A two-byte value used for final sortation in place of the DPC add-on. It consists of an uppercase letter followed by a digit 0 through 9. Values range from A0 (99 descending) through J9 (00 descending), and K0 (00 ascending) through T9 (99 ascending). |

LACS^{Link} Output

| Response Element | Description |
|-------------------|--|
| USLACS | <p>Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following:</p> <p>Y Yes, the address is a candidate for LACS^{Link} processing. If LACS^{Link} is enabled, ValidateAddress will attempt to convert the address using the LACS^{Link} database. If the conversion attempt is successful, the output address is the new address obtained from the LACS^{Link} database. If the attempt is not successful, the address will not be converted.</p> <p>N No, the address is not a candidate for LACS^{Link} processing. LACS^{Link} processing may still be attempted if LACS^{Link} processing is requested, the LACS^{Link} database is installed, and one of the following is true:</p> <ul style="list-style-type: none"> • The address matches to a Rural Route address and the RecordType.Default field returns a Y. • The input address could not be matched to any address in the U.S. Postal Database (Failures due to multiple matches are not LACS^{Link} candidates.) |
| USLACS.ReturnCode | <p>Indicates the success or failure of LACS^{Link} processing. (U.S. addresses only.)</p> <p>A LACS^{Link} processing successful. Record matched through LACS^{Link} processing.</p> <p>00 LACS^{Link} processing failed. No matching record found during LACS^{Link} processing.</p> <p>09 LACS^{Link} processing matched the input address to an older highrise default address. The address has been converted. Rather than provide an imprecise address, LACS^{Link} processing does not provide a new address.</p> <p>14 LACS^{Link} processing failed. Match found during LACS^{Link} processing but conversion did not occur due to other USPS[®] regulations.</p> <p>92 LACS^{Link} processing successful. Record matched through LACS^{Link} processing. Unit number dropped on input.</p> <p>null LACS^{Link} did not process the record, or LACS^{Link} processing was not attempted.</p> |

RDI Output

| Response Element | Description |
|------------------|---|
| RDI | Return values indicating address type. B The address is a business address. R The address is a residential address. M The address is both a residential and a business address. null Not checked because the address did not code at a ZIP + 4 [®] level, or RDI™ was not performed. |

DPV and CMRA Output

| Response Element | Description |
|------------------|--|
| DPV | <p>Indicates the results of Delivery Point Validation (DPV) processing.</p> <p>Y DPV confirmed. Mail can be delivered to the address.</p> <p>N Mail cannot be delivered to the address.</p> <p>S The building number was validated but the unit number could not be confirmed. A building number is the primary address number for a building. A unit number is a number of a distinct mailing address within a building such as an apartment, suite, floor, and so on. For example, in this address 424 is the building number and 12 is the unit number:</p> <p>424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA</p> <p>D The building number was validated but the unit number was missing from input. A building number is the primary address number for a building. A unit number is a number of a distinct mailing address within a building such as an apartment, suite, floor, and so on. For example, in this address 424 is the building number and 12 is the unit number:</p> <p>424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA</p> <p>M The address matches multiple valid delivery points.</p> <p>U The address could not be confirmed because the address did not code at the ZIP + 4[®] level.</p> <p>V The address caused a false-positive violation.</p> |
| CMRA | <p>Indicates if the address is a Commercial Mail Receiving Agency (CMRA)</p> <p>Y Yes, the address is a CMRA.</p> <p>N No, the address is not a CMRA.</p> <p>U Unconfirmed.</p> |

| Response Element | Description |
|------------------|--|
| DPVFootnote | <p>DPV footnote codes.</p> <p>AA Input address matched to the ZIP + 4[®] file.</p> <p>A1 Input address not matched to the ZIP + 4[®] file.</p> <p>BB Input address matched to DPV (all components).</p> <p>CC Input address primary number matched to DPV but secondary number not match (present but not valid).</p> <p>F1 Input address is military; DPV bypassed.</p> <p>G1 Input address is general delivery; DPV bypassed.</p> <p>M1 Input address primary number missing.</p> <p>M3 Input address primary number invalid.</p> <p>N1 Input address primary number matched to DPV but high rise address missing secondary number.</p> <p>P1 Input address missing RR or HC Box number.</p> <p>P3 Input address missing PO, RR, or HC Box number</p> <p>RR Input address matched to CMRA.</p> <p>R1 Input address matched to CMRA but secondary number not present.</p> <p>R7 Input address matched to phantom carrier route R777 (not eligible for street delivery).</p> <p>U1 Input address is unique ZIP; DPV bypassed.</p> |
| DPVVacant | <p>Indicates whether the building is vacant (unoccupied for 90 days). One of the following:</p> <p>Y Yes, the building is vacant.</p> <p>N No, the building is not vacant.</p> <p>null The <code>Option.DPVDetermineVacancy</code> option was not turned on.</p> |
| DPVNoStat | <p>Indicates whether the building is a "no stat" building and therefore unable to receive mail. One of the following:</p> <p>Y Yes, the building is a "no stat" building, which means the building is not receiving mail.</p> <p>N No, the building is not a "no stat" building, which means the building does receive mail.</p> <p>null The option was not turned on.</p> |

Suite^{Link} Output

| Response Element | Description |
|---------------------|---|
| SuiteLinkReturnCode | <p>Indicates whether or not ValidateAddress corrected the secondary address information (U.S. addresses only). One of the following:</p> <p>A ValidateAddress corrected the secondary address information.</p> <p>00 ValidateAddress did not correct the secondary address information.</p> <p>null Suite^{Link} was not performed.</p> <p>XX Suite^{Link} processing encountered an error. For example, an error would occur if the Suite^{Link} database is expired.</p> |
| SuiteLinkMatchCode | <p>Provides additional information on the Suite^{Link} match attempt. (U.S. addresses only)</p> <p>A ValidateAddress corrected the secondary address information.</p> <p>B ValidateAddress did not correct the secondary address information. No additional detail about the match attempt is available.</p> <p>C The words in the FirmName field are all "noise" words. Noise words are defined by the USPS[®] and are ignored when attempting to match the firm name. Examples of noise words are "company" and "corporation". ValidateAddress is not able to correct secondary address information for firm names that consist entirely of noise words. For example "Company and Corporation" is all noise words.</p> <p>D The address is not a high-rise default address. Suite^{Link} matching is only done for high-rise default addresses. A high-rise default is a default to use when the address does not contain valid secondary information (the apartment number or apartment type is missing).</p> <p>E Suite^{Link} processing failed because the Suite^{Link} database is expired.</p> <p>null Suite^{Link} was not performed or there was an error.</p> |

| Response Element | Description |
|-------------------|--|
| SuiteLinkFidelity | <p>Indicates how well ValidateAddress matched the firm name to the firm names in the Suite^{Link} database.</p> <p>1 The firm name matches the Suite^{Link} database exactly.</p> <p>2 Good match. All words in the firm name except one matched the firm name in the Suite^{Link} database.</p> <p>3 Poor match. More than one word in the firm name did not match the firm name in the Suite^{Link} database.</p> <p>null Suite^{Link} could not match the firm name, or was not performed, or there was an error.</p> |

VeriMove Output

| Response Element | Description |
|-------------------|---|
| VeriMoveDataBlock | <p>Indicates whether or not ValidateAddress should return a 250-byte field containing input data to pass to VeriMove Express. This field contains the Detail Results Indicator data required by VeriMove. For more information about the contents of this field, see the VeriMove User's Guide. One of the following:</p> <p>Y Yes, return the field VeriMoveDataBlock.</p> <p>N No, do not return the field VeriMoveDataBlock.</p> |

Additional Input Data

Some input data is ignored during the address standardization process. This extraneous data (sometimes referred to as "dropped data") is returned in the AdditionalInputData field. Some examples of dropped data include:

- Delivery instructions (for example, "Leave at back door")
- Phone numbers (for example, "555-135-8792")
- Attention lines (for example, "Attn: John Smith")

Data such as this is generally not embedded in an address. If it is embedded, the extraneous data can usually be identified and returned in the AdditionalInputData field.

Note: Dropped data from split indicia addresses is not returned. A split indicia address is one where a primary address is split between multiple address lines. For example, if the primary

address is "1 Green River Valley Rd" then the following would be a split indicia version of this address:

1 Green River
Valley Rd
01230

If there is more than one piece of dropped data in an address, each piece of data is separated by a semicolon and a space ("; ") for U.S. addresses and a space for addresses outside the U.S. The order of dropped data in `AdditionalInputData` is:

1. Care of, mail stop (U.S. addresses only)
2. Other extraneous data found on address lines
3. Entire unused data lines

For example, if this is the input address:

123 Main St C/O John Smith
Apt 5 Drop at back dock
jsmith@example.com
555-123-4567
05674

Then `AdditionalInputData` would contain:

C/O John Smith; Apt 5 Drop At Back Dock; 555-123-4567; Jsmith@example.com; 555-123-4567

Care of Data

For U.S. addresses only, "care of" data is returned in `AdditionalInputData`. The following addresses contain examples of "care of" data:

123 Main St C/O John Smith
Apt 5
05674

123 Main St
Apt 5 ATTN John Smith
05674

123 Main St Apt 5
MailStop 2
05674

Extraneous Data on Its Own Address Line

`ValidateAddress` returns extraneous data on its own address line for U.S. and Canadian addresses.

For U.S. addresses, `ValidateAddress` uses the first two non-blank address lines to perform address standardization, unless either the firm name extraction or urbanization code extraction options are enabled (see [Address Line Processing for U.S. Addresses](#) on page 452 for more information).

Data on other address lines is returned in `AdditionalInputData`. In the following address, "John Smith" would be returned in `AdditionalInputData` because it is in the third non-blank address line and `ValidateAddress` only uses the first two non-blank address lines for U.S. addresses.

```
123 Main St
Apt 5
John Smith
05674
```

If one of either of the first two non-blank address lines contains extraneous data, that data is returned in `AdditionalInputData`. For example, in the following addresses "John Smith" would be returned in `AdditionalAddressData`.

```
123 Main St
John Smith
05674
```

```
John Smith
123 Main St
05674
```

In the following address both "John Smith" and "Apt 5" would both be returned in `AdditionalInputData`. "John Smith" would be returned because it is extraneous data in one of the first two address lines and "Apt 5" would be returned because U.S. address data must be in the first two non-blank address lines.

```
John Smith
123 Main St
Apt 5
05674
```

Extraneous Data Within an Address Line

Extraneous data that is within an address line is returned in `AdditionalInputData`. For example, in the following addresses "John Smith" would be returned in `AdditionalInputData`.

```
123 Main St John Smith
05674
```

```
123 Main St Apt 5 John Smith
05674
```

```
123 Main St John Smith
Apt 5
05674
```

```
123 Main St
Apt 5 John Smith
05674
```

For U.S. addresses, only extraneous data at the end of the address line is returned in `AdditionalInputData`. Extraneous data that is not at the end of an address line is not returned for U.S. addresses. For example, in the following addresses "John Smith" is not returned.

John Smith 123 Main St
05674

123 Main John Smith St
05674

The `AdditionalInputData` will sometimes contain the original street name or suffix if the street name was changed to obtain a match and the street name or suffix was at the end of a line. For example this address:

Precisely
4200 Parliament
Lanham MD

`ValidateAddress` would correct the spelling of the street name and add the suffix, returning "4200 Parliament PI" as the corrected street address and "Parlament" in `AdditionalInputData`.

Dual Addresses

A dual address is an address that contains both street and PO Box/Rural Route/Highway Contract information. Depending on the processing options you select, the portion of the dual address that is not used for address standardization may be returned in `AdditionalInputData`. For more information, see [About Dual Address Logic](#) on page 465.

ValidateAddressGlobal

`ValidateAddressGlobal` provides enhanced address standardization and validation for addresses outside the U.S. and Canada. `ValidateAddressGlobal` can also validate addresses in the U.S. and Canada but its strength is validation of addresses in other countries. If you process a significant number of addresses outside the U.S. and Canada, you should consider using `ValidateAddressGlobal`.

`ValidateAddressGlobal` is part of the Universal Addressing Module.

`ValidateAddressGlobal` performs several steps to achieve a quality address, including transliteration, parsing, validation, and formatting.

Character Set Mapping and Transliteration

`ValidateAddressGlobal` handles international strings and their complexities. It uses fully Unicode enabled string processing which enables the transliteration of non-roman characters into the Latin character set and mapping between different character sets.

Character set mapping and transliteration features include:

- Support for over 30 different character sets including UTF-8, ISO 8859-1, GBK, BIG5, JIS, EBCDIC
- Proper "elimination" of diacritics according to language rules
- Transliteration for various alphabets into Latin Script
- Greek (BGN/PCGN 1962, ISO 843 - 1997)
- Cyrillic (BGN/PCGN 1947, ISO 9 - 1995)

- Hebrew
- Japanese Katakana, Hiragana and Kanji
- Chinese Pinyin (Mandarin, Cantonese)
- Korean Hangul

Address Parsing, Formatting, and Standardization

Restructuring incorrectly fielded address data is a complex and difficult task especially when done for international addresses. People introduce many ambiguities as they enter address data into computer systems. Among the problems are misplaced elements (such as company or personal names in street address fields) or varying abbreviations that are not only language, but also country specific. `ValidateAddressGlobal` identifies address elements in address lines and assigns them to the proper fields. This is an important precursor to the actual validation. Without restructuring, "no match" situations might result.

Properly identified address elements are also important when addresses have to be truncated or shortened to fit specific field length requirements. With the proper information in the right fields, specific truncation rules can be applied.

- Parses and analyzes address lines and identifies individual address elements
- Processes over 30 different character sets
- Formats addresses according to the postal rules of the country of destination
- Standardizes address elements (such as changing AVENUE to AVE)

Global Address Validation

Address validation is the correction process where properly parsed address data is compared against reference databases supplied by postal organizations or other data providers. `ValidateAddressGlobal` validates individual address elements to check for correctness using sophisticated fuzzy matching technology and produces standardized and formatted output based on postal standards and user preferences. `FastCompletion` validation type can be used in quick address entry applications. It allows input of truncated data in several address fields and generates suggestions based on this input.

In some cases, it is not possible to fully validate an address. Here `ValidateAddressGlobal` has a unique deliverability assessment feature that classifies addresses according to their probable deliverability.

Resource URL

JSON endpoint:

```
http://server:port/rest/ValidateAddressGlobal/results.json
```

XML endpoint:

```
http://server:port/rest/ValidateAddressGlobal/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/ValidateAddressGlobal/results.json?
City.StateProvince=NY&Data.AddressLine1=1+Global+View&
Data.City=Troy&Data.Country=USA
```

The JSON returned by this request would be:

```
{
  "output": [
    {
      "Country.Input": "USA",
      "AddressLine1.Input": "1 Global View",
      "City.Input": "Troy",
      "Country": "UNITED STATES",
      "AddressLine1": "1 GLOBAL VW",
      "HouseNumber": "1",
      "StreetName": "GLOBAL",
      "StreetSuffix": "VW",
      "City": "TROY",
      "PostalCode": "12180-8371",
      "PostalCode.Base": "12180",
      "PostalCode.AddOn": "8371",
      "StateProvince": "NEW YORK",
      "County": "RENSSELAER",
      "LastLine": "TROY NY 12180-8371",
      "AddressBlock1": "1 GLOBAL VW",
      "AddressBlock2": "TROY NY 12180-8371",
      "ProcessStatus": "C4",
      "ProcessStatus.Description": "Corrected - all elements have been
checked",
      "ModeUsed": "BATCH",
      "CountOverflow": "NO",
      "MailabilityScore": "5",
      "Confidence": "82.09",
      "ElementResultStatus": "88F088E0F000000000E0",
      "ElementInputStatus": "00600050600000000060",
      "ElementRelevance": "11101010100000000010",
      "AddressType": "S",
      "AMAS.Status": "EAM0",
      "user_fields": []
    }
  ]
}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/ValidateAddressGlobal/results.xml?
Data.AddressLine1=1+Global+VW&Data.City=Troy&
Data.Country=USA&Data.StateProvince=NY
```

The XML returned by this request would be:

```
<ns2:xml.ValidateAddressGlobalResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/ValidateAddressGlobal">

  <ns2:output>
    <ns2:Address>
      <ns2:Country>UNITED STATES</ns2:Country>
      <ns2:AddressLine1>1 GLOBAL VW</ns2:AddressLine1>
      <ns2:HouseNumber>1</ns2:HouseNumber>
      <ns2:StreetName>GLOBAL</ns2:StreetName>
      <ns2:StreetSuffix>VW</ns2:StreetSuffix>
      <ns2:City>TROY</ns2:City>
      <ns2:PostalCode>12180-8371</ns2:PostalCode>
      <ns2:PostalCode.Base>12180</ns2:PostalCode.Base>
      <ns2:PostalCode.AddOn>8371</ns2:PostalCode.AddOn>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:County>RENSSELAER</ns2:County>
      <ns2>LastLine>TROY NY 12180-8371</ns2>LastLine>
      <ns2:AddressBlock1>1 GLOBAL VW</ns2:AddressBlock1>
      <ns2:AddressBlock2>TROY NY 12180-8371</ns2:AddressBlock2>
      <ns2:ProcessStatus>C4</ns2:ProcessStatus>
      <ns2:ProcessStatus.Description>
        Corrected - all elements have been checked
      </ns2:ProcessStatus.Description>
      <ns2:ModeUsed>BATCH</ns2:ModeUsed>
      <ns2:CountOverflow>NO</ns2:CountOverflow>
      <ns2:MailabilityScore>5</ns2:MailabilityScore>
      <ns2:Confidence>85.09</ns2:Confidence>
      <ns2:ElementResultStatus>
        88F0F8E0F000000000E0
      </ns2:ElementResultStatus>
      <ns2:ElementInputStatus>
        00606050600000000060
      </ns2:ElementInputStatus>
      <ns2:ElementRelevance>
        11101010100000000010
      </ns2:ElementRelevance>
      <ns2:AddressType>S</ns2:AddressType>
      <ns2:AMAS.Status>EAM0</ns2:AMAS.Status>
      <ns2:user_fields/>
    </ns2:Address>
  </ns2:output>
</ns2:xml.ValidateAddressGlobalResponse>
```

Request

Parameters for Input Data

ValidateAddressGlobal takes a standard address as input. All addresses use this format no matter what country the address is from.

Table 51: ValidateAddressGlobal Input

| Parameter | Format | Description |
|---|--------------|--|
| Data.AddressLine1 through Data.AddressLine6 | String [79] | <p>These fields contain address line data. AddressLine1 contains the first address line, AddressLine2 contains the second address line, and so forth. Note that the city, state/province, and postal code information should be placed in their respective fields, not address line fields. For example:</p> <p>AddressLine1: 17413 Blodgett Road AddressLine2: PO Box 123 City: Mount Vernon StateProvince: WA PostalCode: 97273 Country: USA</p> <p>If the input address is not already parsed into the appropriate address line and City, StateProvince, and PostalCode fields, use the UnformattedLine fields instead of the address line fields.</p> |
| Data.City | String [79] | City name |
| Data.StateProvince | String [79] | State or province. |
| Data.PostalCode | String [79]: | <p>The postal code for the address. In the U.S. this is the ZIP Code®.</p> <p>99999 99999-9999 A9A9A9 A9A 9A9 9999 999</p> |
| Data.Contact | String [79] | The name of the addressee. For example, "Mr. Jones". |

| Parameter | Format | Description |
|---|------------------|---|
| Data.Country | String [79] | The name of the country. If no value is specified in the or option, you must specify a country. |
| Data.FirmName | String [79] | The company or firm name. |
| Data.Street | String [79] | Street |
| Data.Number | Building [79] | Number |
| Data.Building | String [79] | Building |
| Data.SubBuilding | String [79] | SubBuilding |
| Data.DeliveryService | String [79] | DeliveryService |
| Data.UnformattedLine1 through Data.UnformattedLine10 | String [79] | <p>Use these fields if the input address is completely unparsed and you want ValidateAddressGlobal to attempt to parse the address into the appropriate fields. For example:</p> <p>UnformattedLine1: 17413 Blodgett Road UnformattedLine2: PO Box 123 UnformattedLine3: Mount Vernon WA 97273 UnformattedLine4: USA</p> <p>This address would be parsed into these output fields:</p> <p>AddressLine1: 17413 Blodgett Road AddressLine2: PO Box 123 City: Mount Vernon StateProvince: WA PostalCode: 97273 Country: USA</p> <p>Note: If you specify input in the unformatted line fields you must specify the entire address using only unformatted line fields. Do not use other fields such as City or StateProvince in combination with unformatted line fields.</p> |

Parameters for Options

Input Options

Table 52: ValidateAddressGlobal Input Options

| Parameter | Description/Valid Values |
|---------------------------------|---|
| Option.Database.AddressGlobal | Specifies the database resource containing the postal data to use for address validation. Only databases that have been defined in the Global Database Resources panel in the Management Console are available. For more information, see the <i>Spectrum Technology Platform Administration Guide</i> . |
| Option.Input.DefaultCountryISO3 | Specifies a default country to use when the input record does not contain explicit country information. Specify the country using the ISO3 country code. If you do not specify a default country each input record must have the country specified in the Country input field. For a list of ISO codes see ISO Country Codes and Module Support on page 1011. |
| Option.Input.ForceCountryISO3 | Causes address records to be always treated as originating from the country specified here, overriding the country in the address record and the default country. Specify the country using the ISO3 country code. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| Option.Input.FormatDelimiter | <p>Enables you to use non-standard formatting for multiline addresses in input files. Acceptable values for this field include the following:</p> <ul style="list-style-type: none"> • CRLF (default) • LF • CR • SEMICOLON (2101 MASSACHUSETTS AVE NW ; WASHINGTON DC 20008) • COMMA (2101 MASSACHUSETTS AVE NW , WASHINGTON DC 20008) • TAB (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • PIPE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • SPACE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) <p>Note: The same value must be selected for both the input option and output option.</p> |

Output Options

Table 53: ValidateAddressGlobal Output Options

| Parameter | Description |
|---------------------------------|---|
| Option.Result.MaximumResults | This option specifies the maximum number of candidate addresses to return. This field is disabled for batch processing; for all other processing modes the default is 1 and the maximum is 99. If you are using FastCompletion mode, you may want to enter a number greater than 1 to ensure you are provided with multiple options for completing a field. |
| Option.Result.IncludeInputs | <p>Specifies whether to include the input data in the output. If enabled, the output will contain fields that end with .Input containing the corresponding input field. For example, the output field AddressLine1.Input would contain the data specified in the input field AddressLine1.</p> <p>TRUE Include the input data in the output.</p> <p>FALSE Do not include the input data in the output (default).</p> |
| Option.Result.StateProvinceType | <p>Specifies the format for the StateProvince field. One of the following.</p> <p>ABBREVIATION Return the abbreviation for the state or province. For example, North Carolina would be returned as "NC".</p> <p>COUNTRY_STANDARD Return either the abbreviation or the full name depending on the format used by the country's postal authority. (Default)</p> <p>EXTENDED Return the full name of the state or province, not the abbreviation. For example "North Carolina".</p> |

| Parameter | Description |
|---------------------------|---|
| Option.Result.CountryType | Specifies the language or code to use for the country name returned by ValidateAddressGlobal. |
| ISO2 | The two-character ISO code for the country |
| ISO3 | The three-character ISO code for the country |
| ISO_NUMBER | The ISO country number |
| NAME_CN | Chinese |
| NAME_DA | Danish |
| NAME_DE | German |
| NAME_EN | English (default) |
| NAME_ES | Spanish |
| NAME_FI | Finnish |
| NAME_FR | French |
| NAME_GR | Greek |
| NAME_HU | Hungarian |
| NAME_IT | Italian |
| NAME_JP | Japanese |
| NAME_KR | Korean |
| NAME_NL | Dutch |
| NAME_PL | Polish |
| NAME_PT | Portuguese |
| NAME_RU | Russian |
| NAME_SA | Sanskrit |
| NAME_SE | Swedish |

| Parameter | Description |
|---------------------------------|---|
| Option.Result.PreferredScript | <p>Specifies the alphabet in which the output should be returned. The alphabet in which the data is returned differs from country to country. For most countries the output will be Latin I regardless of the selected preferred language.</p> <p>ASCII_Extended ASCII characters with expansion of special characters (for example, Ã– = OE)</p> <p>ASCII_Simplified ASCII characters</p> <p>Database (default) Latin I or ASCII characters (as per reference database standard)</p> <p>Latin Latin I characters</p> <p>Latin_Alt Latin I characters (alternative transliteration)</p> <p>Postal_Admin_Alt Latin I or ASCII characters (local postal administration alternative)</p> <p>Postal_Admin_Pref Latin I or ASCII characters (as preferred by local postal administration)</p> <p>For countries that use an alphabet other than Latin I, the returned alphabet differs from country to country. For more information, see Alphabets for Non-Latin 1 Countries on page 537.</p> |
| Option.Result.PreferredLanguage | <p>Specifies the language in which the output should be returned. The alphabet in which the data is returned differs from country to country, but for most countries the output will be Latin, regardless of the selected preferred language.</p> <p>DATABASE Language derived from reference data for each address. Default.</p> <p>ENGLISH English locality and state/province names output, if available.</p> |
| Option.Result.FormatDelimiter | <p>Enables you to use non-standard formatting for multiline addresses in the output. Acceptable values for this field include the following:</p> <ul style="list-style-type: none"> • CRLF (default) • LF • CR • SEMICOLON (2101 MASSACHUSETTS AVE NW ; WASHINGTON DC 20008) • COMMA (2101 MASSACHUSETTS AVE NW , WASHINGTON DC 20008) • TAB (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • PIPE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • SPACE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) <p>Note: The same value must be selected for both the input option and output option.</p> |

| Parameter | Description |
|----------------------|--|
| Option.Result.Casing | Specifies the casing of the output. |
| | NATIVE Output will be based on the reference database standard. |
| | UPPER Output will be in upper case for all countries. |
| | LOWER Output will be in lower case for all countries. |
| | MIXED Casing determined by country-specific rules. |
| | NOCHANGE For parse mode, returns the data the way it was entered. For validation mode, uses the casing found in the reference data and according to postal rules. Values that could not be checked against the reference data will retain their input casing. |

Alphabets for Non-Latin 1 Countries

For countries that use an alphabet other than Latin I, the returned alphabet differs from country to country. The following table shows how the output is returned for specific countries. All countries that are not listed use the value specified in the field option.

| Country | Database | | | Latin | | | |
|---------|----------|----------|----------|-----------------------|-----------------------|---|----------------------------------|
| RUS | Cyrillic | Cyrillic | Cyrillic | CYRILLIC_ISO | CYRILLIC_BGN | CYRILLIC_ISO + LATIN_SIMPLE | CYRILLIC_ISO + LATIN |
| JPN | Kanji | Kanji | Kana | JAPANESE | JAPANESE | JAPANESE + LATIN_SIMPLE | JAPANESE + LATIN |
| CHN | Hanzi | Hanzi | Hanzi | CHINESE_ MANDARIN | CHINESE_ CANTONESE | CHINESE_ MANDARIN + LATIN_SIMPLE | CHINESE_ MANDARIN + LATIN |
| HKG | Hanzi | Hanzi | Hanzi | CHINESE_ CANTONESE | CHINESE_ MANDARIN | CHINESE_ CANTONESE + LATIN_SIMPLE | CHINESE_ CANTONESE + LATIN |
| TWN | Hanzi | Hanzi | Hanzi | CHINESE_ CANTONESE | CHINESE_ MANDARIN | CHINESE_ CANTONESE + LATIN_SIMPLE | CHINESE_ CANTONESE + LATIN |

| Country Database | | | | Latin | | | |
|------------------|---------|---------|---------|-----------|-----------|-----------------------------|----------------------|
| GRC | Greek | Greek | Greek | GREEK_ISO | GREEK_BGN | GREEK_ISO + LATIN_SIMPLE | GREEK_ISO + LATIN |
| KOR | Latin | Hangul | Hanja | KOREAN | KOREAN | KOREAN + LATIN_SIMPLE | KOREAN + LATIN |
| ISR | Latin | Hebrew | Hebrew | HEBREW | HEBREW | HEBREW + LATIN_SIMPLE | HEBREW + LATIN |
| ROM | Latin-3 | Latin-3 | Latin-3 | Latin-3 | Latin-3 | LATIN_SIMPLE | LATIN |
| POL | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| CZE | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| CRI | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| HUN | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| MDA | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| SVK | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| LAT | Latin-7 | Latin-7 | Latin-7 | Latin-7 | Latin-7 | LATIN_SIMPLE | LATIN |

Process Options

Table 54: ValidateAddressGlobal Process Options

| Parameter | Description |
|----------------------------------|---|
| Option.Process.OptimizationLevel | <p>Use this option to set the appropriate balance between processing speed and quality. One of the following:</p> <p>NARROW The parser will honor input assignment strictly, with the exception of separation of House Number from Street information.</p> <p>STANDARD The parser will separate address element more actively as follows:</p> <ul style="list-style-type: none"> • Province will be separated from Locality information • PostalCode will be separated from Locality information • House Number will be separated from Street information • SubBuilding will be separated from Street information • DeliveryService will be separated from Street information • SubBuilding will be separated from Building information • Locality will be separated from PostalCode information <p>WIDE Parser separation will happen similarly to Standard, but additionally up to 10 parsing candidates will be passed to validation for processing. Validation will widen its search tree and take additional reference data entries into account for matching.</p> <p>Please note that adjusting the optimization level might have no effect for countries that lack the postal reference data information required for the kind of separation described above.</p> <p>Increasing separation granularity from Narrow to Standard consumes some processing power, but the major impact on processing speed is from validation processing a larger search tree, thus increasing the number of data accesses and comparisons for the optimization level Wide, in an attempt to make the most out of the input data given.</p> |

| Parameter | Description |
|---------------------|--|
| Option.Process.Mode | <p>Specifies the type of processing to perform on the addresses. One of the following:</p> <p>BATCH Use this mode in batch processing environments when no human input or selection is possible. It is optimized for speed and will terminate its attempts to correct an address when ambiguous data is encountered that cannot be corrected automatically. The Batch processing mode will fall back to Parse mode when the database is missing for a specific country.</p> <p style="padding-left: 40px;">Note: When the Process Status returns a value of I3, the attempt is considered a failure and the Status will return a value of F.</p> <p>CERTIFIED Use this mode in batch processing environments for Australian mail. Validate Address Global is certified by Australia Post's Address Matching Approval System (AMAS). It will standardize and validate your mail against the Postal Address File, providing postal discounts and allowing for the least amount of undeliverable pieces.</p> <p>FASTCOMPLETION Use this mode if you want to use FastCompletion mode to enter truncated data in address fields and have Validate Address Global generate suggestions. For example, if you work in a call center or point-of-sale environment, you can enter just part of an address element and the FastCompletion feature will provide valid options for the complete element.</p> <p>INTERACTIVE Use this mode when working in interactive environments to generate suggestions when an address input is ambiguous. This validation type is especially useful in data entry environments when capturing data from customers or prospects. It requires the input of an almost-complete address and will attempt to validate or correct the data provided. If ambiguities are detected, this validation type will generate up to 20 suggestions that can be used for pick lists. The Interactive processing mode will fall back to Parse mode when the respective database is missing for a specific country.</p> <p>PARSE Use this mode for separating address input into tokens for subsequent processing in other systems, bypassing validation. For example, you could use this mode when address data of already high quality simply needs to be tokenized quickly for export to an external system or for use by a downstream stage.</p> |

| Parameter | Description |
|------------------------------|---|
| Option.Process.MatchingScope | <p>Specifies how closely an address must match the reference data in order for the address to be validated. One of the following:</p> <p>Note: These settings may not have an effect for countries lacking the necessary level of detail in the postal reference data.</p> <p>ALL All address elements must match.</p> <p>DELIVERYPOINT_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, street, house number, and sub building.</p> <p>STREET_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, and street.</p> <p>LOCALITY_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, and City/Locality/Suburb.</p> |

Response

Address Data

Table 55: Parsed Address Elements

| Response Element | Description |
|------------------|---|
| AddressBlock1-9 | <p>The AddressBlock output fields contain a formatted version of the standardized or normalized address as it would be printed on a physical mailpiece. Validate Address Global formats the address into address blocks using postal authority standards. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9. For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882</p> |

| Response Element | Description |
|------------------------|---|
| AddressLine1-6 | <p>If the address was validated, the address line fields contain the validated and standardized address lines. If the address could not be validated, the address line fields contain the input address without any changes. Note that the last line of the address is contained in the LastLine field. For example:</p> <p>AddressLine1: 4200 PARLIAMENT PL STE 600 LastLine: LANHAM MD 20706-1882</p> |
| AdministrativeDistrict | An area smaller than a state/province but larger than a city. |
| ApartmentLabel | The flat or unit type (such as STE or APT), for example: 123 E Main St Apt 3 |
| ApartmentNumber | The flat or unit number, for example: 123 E Main St Apt 3 |
| BlockName | An estate or block name. |
| BuildingName | The name of a building, for example Sears Tower. |
| City | The name of the town or city. For example, Vancouver , BC. |
| City.AddInfo | Additional information about the city. |
| City.SortingCode | A code used by the postal authority to speed up delivery in certain countries for large localities, for example Prague or Dublin. |
| Contact | The name of the addressee. For example, Mr. Jones . |
| Country | The country in the language or code specified in the option. |
| County | Dependent state or province information that further subdivides a state or province. An example would be a U.S. county. |
| FirmName | The name of a company. |
| Floor | Information that further subdivides a building, for example, the suite or apartment number. For example: 123 E Main St Apt 3, 4th Floor |

| Response Element | Description |
|--------------------|---|
| HouseNumber | The house number 1, for example: 298A-1B New South Head Rd |
| LastLine | Complete last address line (city, state/province, and postal code). |
| LeadingDirectional | Street directional that precedes the street name. For example, the N in 138 N Main Street. |
| Locality | Dependent place name that further subdivides a Locality. Examples are colonias in Mexico, Urbanisaciones in Spain. |
| POBox | Post Box descriptor (POBox, Postfach, Case Postale etc.) and number. |
| PostalCode | The postal code for the address. The format of the postcode varies by country. |
| PostalCode.AddOn | The second part of a postcode. For example, for Canadian addresses this will be the LDU. For U.S. addresses this is the ZIP + 4 add on. This field is not used by most countries. |
| PostalCode.Base | The base portion of the postcode. |
| Room | A room number in a building. |
| SecondaryStreet | The name of a secondary street or rural route. |
| StateProvince | The name of the state or province. |
| StreetName | The name of street where property is located, for example: 123 E Main St Apt 3 |
| StreetSuffix | The street suffix, for example: 123 E Main St Apt 3 |
| SubBuilding | A portion of a building, such as a suite. For example, Suite 102. |
| Suburb | Dependent place name that further subdivides a Locality. An example would be Mahalle in Turkey. |
| Territory | The name of a territory. Territories are larger than a state/province. |

| Response Element | Description |
|---------------------|---|
| TrailingDirectional | The trailing directional, for example: 123 Pennsylvania Ave NW |

Original Input Data

This option outputs the original input data in <FieldName>.Input fields.

Table 56: Original Input Data

| Response Element | Format | Description |
|---------------------|-------------|---------------------|
| AddressLine1.Input | String [79] | First address line |
| AddressLine2.Input | String [79] | Second address line |
| AddressLine3.Input | String [79] | Third address line |
| AddressLine4.Input | String [79] | Fourth address line |
| AddressLine5.Input | String [79] | Fifth address line |
| AddressLine6.Input | String [79] | Sixth address line |
| City.Input | String [79] | City name |
| StateProvince.Input | String [79] | State or province |

| Response Element | Format | Description |
|-----------------------|---------------|---|
| PostalCode.Input | String [79] | The postal code for the address. In the U.S. this is the ZIP Code. One of these formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 |
| Contact.Input | String [79] | The name of the addressee. For example, "Mr. Jones". |
| Country.Input | String [79] | Specify the country using the format you chose for input country format (English name, ISO code, or UPU code). For a list of valid values, see ISO Country Codes and Module Support on page 1011. |
| FirmName.Input | String [79] | The company or firm name. |
| Street.Input | String [79] | Street |
| Number.Input | Building [79] | Number |
| Building.Input | String [79] | Building |
| SubBuilding.Input | String [79] | SubBuilding |
| DeliveryService.Input | String [79] | DeliveryService |

Result Codes

These output fields contain information about the result of the validation processing.

Table 57: Result Codes

| Response Element | Result Code |
|--------------------|---|
| AddressType | <p>For United States and Canada addresses only, the AddressType field indicates the type of address. One of the following:</p> <p>F The address was validated/corrected to the firm name.</p> <p>B The address was validated/corrected to the building name.</p> <p>G The address is a general delivery address.</p> <p>H The address was validated/corrected to the high-rise default.</p> <p>L The address is a large volume receiver.</p> <p>M The address is a military address.</p> <p>P The address was validated/corrected to PO box.</p> <p>R The address was validated/corrected to a rural route.</p> <p>S The address was validated/corrected to a street address.</p> <p>U The address could not be validated/corrected so the type is unknown.</p> |
| Confidence | <p>The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct.</p> |
| CountOverflow | <p>Indicates whether the number of candidate addresses exceeds the number returned. One of the following:</p> <p>Yes Yes, there are additional candidate addresses. To obtain the additional candidates, increase the value.</p> <p>No No, there are no additional candidates.</p> |
| ElementInputStatus | <p>ElementInputStatus provides per element information on the matching of input elements to reference data. The values in this field vary depending on whether you are using batch mode or parse mode. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 551.</p> |
| ElementRelevance | <p>Indicates which address elements are actually relevant from the local postal authority's point of view. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 551 .</p> |

| Response Element | Result Code | | | | | | | | | | | | |
|---------------------|---|----------|--|----------|------------------------------|----------|-----------------------|----------|-------------|----------|-------|----------|-----------|
| ElementResultStatus | ElementResultStatus categorizes the result in more detail than the ProcessStatus field by indicating if and how the output fields have been changed from the input fields. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 551. | | | | | | | | | | | | |
| MailabilityScore | <p>An estimate of how likely it is that mail sent to the address would be successful delivered. One of the following:</p> <table border="0"> <tr> <td>5</td> <td>Completely confident of deliverability</td> </tr> <tr> <td>4</td> <td>Almost certainly deliverable</td> </tr> <tr> <td>3</td> <td>Should be deliverable</td> </tr> <tr> <td>2</td> <td>Fair chance</td> </tr> <tr> <td>1</td> <td>Risky</td> </tr> <tr> <td>0</td> <td>No chance</td> </tr> </table> | 5 | Completely confident of deliverability | 4 | Almost certainly deliverable | 3 | Should be deliverable | 2 | Fair chance | 1 | Risky | 0 | No chance |
| 5 | Completely confident of deliverability | | | | | | | | | | | | |
| 4 | Almost certainly deliverable | | | | | | | | | | | | |
| 3 | Should be deliverable | | | | | | | | | | | | |
| 2 | Fair chance | | | | | | | | | | | | |
| 1 | Risky | | | | | | | | | | | | |
| 0 | No chance | | | | | | | | | | | | |
| ModeUsed | Indicates the processing mode used. The processing mode is specified in the option. For a description of the modes, see Process Options on page 539. | | | | | | | | | | | | |
| MultimatchCount | If the address was matched to multiple candidate addresses in the reference data, this field contains the number of candidate matches found. | | | | | | | | | | | | |

| Response Element | Result Code |
|------------------|-------------|
|------------------|-------------|

| | |
|---------------|--|
| ProcessStatus | |
|---------------|--|

Response Element

Result Code

Provides a general description of the output quality. For a more detailed description of the output quality, see the ElementResultStatus field.

One of the following:

| | |
|-----------|--|
| V4 | Verified. The input data is correct. All elements were checked and input matched perfectly. |
| V3 | Verified. The input data is correct on input but some or all elements were standardized or the input contains outdated names or exonyms. |
| V2 | Verified. The input data is correct but some elements could not be verified because of incomplete reference data. |
| V1 | Verified. The input data is correct but the user standardization has deteriorated deliverability (wrong element user standardization - for example, postcode length chosen is too short). Not set by validation. |
| C4 | Corrected. All elements have been checked. |
| C3 | Corrected, but some elements could not be checked. |
| C2 | Corrected, but delivery status unclear (lack of reference data). |
| C1 | Corrected, but delivery status unclear because user standardization was wrong. Not set by validation. |
| I4 | Data could not be corrected completely, but is very likely to be deliverable. Single match (for example, HNO is wrong but only 1 HNO is found in reference data). |
| I3 | Data could not be corrected completely, but is very likely to be deliverable. Multiple matches (for example, HNO is wrong but more than 1 HNO is found in reference data). |
| I2 | Data could not be corrected, but there is a slim chance that the address is deliverable. |
| I1 | Data could not be corrected and is unlikely to be delivered. |
| RA | Country recognized from the Force country Setting |
| R9 | Country recognized from DefaultCountryISO3 Setting |
| R8 | Country recognized from name without errors |
| R7 | Country recognized from name with errors |
| R6 | Country recognized from territory |
| R5 | Country recognized from province |
| R4 | Country recognized from major town |
| R3 | Country recognized from format |
| R2 | Country recognized from script |
| R1 | Country not recognized - multiple matches |

| Response Element | Result Code |
|---------------------------|--|
| | R0 Country not recognized |
| | S4 Parsed perfectly |
| | S3 Parsed with multiple results |
| | S2 Parsed with errors. Elements change position. |
| | S1 Parse Error. Input Format Mismatch. |
| | N1 Validation Error: No validation performed because country was not recognized. |
| | N2 Validation Error: No validation performed because required reference database is not available. |
| | N3 Validation Error: No validation performed because country could not be unlocked. |
| | N4 Validation Error: No validation performed because reference database is corrupt or in wrong format. |
| | N5 Validation Error: No validation performed because reference database is too old. |
| | N6 Validation Error: No validation performed because input data was insufficient. |
| | Q3 FastCompletion Status: Suggestions are available - complete address. |
| | Q2 FastCompletion Status: Suggested address is complete but combined with elements from the input (added or deleted). |
| | Q1 FastCompletion Status: Suggested address is not complete (enter more information). |
| | Q0 FastCompletion Status: Insufficient information provided to generate suggestions. |
| Status | Reports the success or failure of the processing attempt. null Success F Failure |
| Status.Code | The reason for the failure, if there was one. |
| Status.Description | A description of the reason for the failure, if there was one. |

Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance

The ElementInputStatus, ElementResultStatus, and ElementRelevance output fields contain a series of digits that describe the outcome of the validation operation in detail. ElementInputStatus contains some information for parsing operations.

This is what an ElementInputStatus value looks like:

44606040600000000060

This is what an ElementResultStatus value looks like:

88F0F870F00000000040

This is what an ElementRelevance value looks like:

11101010100000000000

To understand the values in these fields you need to know which element each position represents, and the meaning of the values in each position. For example, the first digit indicates the result from the PostalCode.Base output field. The position meanings are listed below.

- Position 1—PostalCode.Base
- Position 2—PostalCode.AddOn
- Position 3—City
- Position 4—Locality and Suburb
- Position 5—StateProvince
- Position 6—County
- Position 7—StreetName
- Position 8—SecondaryStreet
- Position 9—HouseNumber
- Position 10—Number level 1
- Position 11—POBox
- Position 12—Delivery service level 1
- Position 13—Building level 0
- Position 14—BuildingName
- Position 15—Sub building level 0
- Position 16—Floor and Room
- Position 17—FirmName
- Position 18—Organization level 1
- Position 19—Country
- Position 20—Territory

For ElementInputStatus, the possible values for validation are:

- 0—Empty
- 1—Not found
- 2—Not checked (no reference data)

- 3—Wrong - Set by validation only: The reference database suggests that either Number or DeliveryService is out of valid number range. Input is copied, not corrected for batch mode, for interactive mode and FastCompletion suggestions are provided.
- 4—Matched with errors in this element
- 5—Matched with changes (inserts and deletes) For example:
 - Parsing: Splitting of house number for "MainSt 1"
 - Validation: Replacing input that is an exonym or dropping superfluous fielded input that is invalid according to the country reference database
- 6—Matched without errors

For ElementInputStatus, the possible values for parsing are:

- 0—Empty
- 1—Element had to be relocated
- 2—Matched but needed to be normalized
- 3—Matched

For ElementRelevance, the possible values for parsing are:

- 0—Empty
- 1—Element had to be relocated
- 2—Matched but needed to be normalized
- 3—Matched

For ElementResultStatus, the possible values are (for all address elements apart from country):

- 0—Empty
- 1—Not validated and not changed. Original is copied.
- 2—Not validated but standardized.
- 3—Validated but not changed due to invalid input, database suggests that number is out of valid ranges. Input is copied, not corrected - this status value is only set in batch mode.
- 4—Validated but not changed due to lack of reference data.
- 5—Validated but not changed due to multiple matches. Only set in batch mode, otherwise multiple suggestions that replace the input are marked as corrected (status value 7).
- 6—Validated and changed by eliminating the input value
- 7—Validated and changed due to correction based on reference data
- 8—Validated and changed by adding value based on reference data
- 9—Validated, not changed, but delivery status not clear (for example, DPV value wrong; given number ranges that only partially match reference data).
- C—Validated, verified but changed due to outdated name
- D—Validated, verified but changed from exonym to official name
- E—Validated, verified but changed due to standardization based on casing or language. Validation only sets this status if input fully matches a language alternative.
- F—Validated, verified and not changed due to perfect match

For Country (position 19 & 20), the following values are possible:

- 0—Empty
- 1—Country not recognized
- 4—Country recognized from DefaultCountryISO3 setting
- 5—Country not recognized - multiple matches
- 6—Country recognized from script
- 7—Country recognized from format
- 8—Country recognized from major town
- 9—Country recognized from province
- C—Country recognized from territory
- D—Country recognized from name with errors
- E—Country recognized from name without errors
- F—Country recognized from ForceCountryISO3 setting

ValidateAddressLoqate

ValidateAddressLoqate standardizes and validates addresses using postal authority address data. ValidateAddress Loqate can correct information and format the address using the format preferred by the applicable postal authority. It also adds missing postal information, such as postal codes, city names, state/province names, and so on.

ValidateAddressLoqate also returns result indicators about validation attempts, such as whether or not ValidateAddressLoqate validated the address, the level of confidence in the returned address, the reason for failure if the address could not be validated, and more.

During address matching and standardization, ValidateAddressLoqate separates address lines into components and compares them to the contents of the Spectrum Universal Address databases. If a match is found, the input address is *standardized* to the database information. If no database match is found, ValidateAddressLoqate optionally *formats* the input addresses. The formatting process attempts to structure the address lines according to the conventions of the appropriate postal authority.

ValidateAddressLoqate is part of Spectrum Universal Address.

Resource URL

JSON endpoint:

```
http://server:port/rest/ValidateAddressLoqate/results.json
```

XML endpoint:

```
http://server:port/rest/ValidateAddressLoqate/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/ValidateAddressLoqate/results.json?
Data.AddressLine1=1+Global+View&Data.City=Troy&Data.StateProvince=NY
```

The JSON returned by this request would be:

```
{
  "output_port": [
    {
      "Confidence": "95",
      "CouldNotValidate": "",
      "ProcessedBy": "LOQATE",
      "MatchScore": "100.0",
      "AddressLine1": "1 Global Vw",
      "AddressLine2": "",
      "City": "Troy",
      "StateProvince": "NY",
      "PostalCode": "12180-8371",
      "Country": "United States",
      "FirmName": "",
      "PostalCode.Base": "12180",
      "PostalCode.AddOn": "8371",
      "user_fields": []
    }
  ]
}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/ValidateAddressLoqate/results.xml?
Data.AddressLine1=1+Global+View&Data.City=Troy&Data.StateProvince=NY
```

The XML returned by this request would be:

```
<ns2:xml.ValidateAddressLoqateResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/ValidateAddressLoqate">

  <ns2:output_port>
    <ns2:Address>
      <ns2:Confidence>95</ns2:Confidence>
      <ns2:CouldNotValidate/>
      <ns2:ProcessedBy>LOQATE</ns2:ProcessedBy>
      <ns2:MatchScore>100.0</ns2:MatchScore>
      <ns2:AddressLine1>1 Global Vw</ns2:AddressLine1>
      <ns2:AddressLine2/>
      <ns2:City>Troy</ns2:City>
      <ns2:StateProvince>NY</ns2:StateProvince>
      <ns2:PostalCode>12180-8371</ns2:PostalCode>
      <ns2:PostalCode.Base>12180</ns2:PostalCode.Base>
    
```

```

    <ns2:PostalCode.AddOn>8371</ns2:PostalCode.AddOn>
    <ns2:Country>United States</ns2:Country>
    <ns2:FirmName/>
    <ns2:user_fields/>
  </ns2:Address>
</ns2:output_port>
</ns2:xml.ValidateAddressLoqateResponse>

```

Request

Parameters for Input Data

Table 58: Input Format

| Parameter | Format | Description |
|-------------------|--------|--|
| Data.AddressLine1 | String | The first address line. |
| Data.AddressLine2 | String | The second address line. |
| Data.AddressLine3 | String | The third address line. |
| Data.AddressLine4 | String | The fourth address line. |
| Data.City | String | The city name. |
| Data.Country | String | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • Two-character ISO 3166-1 Alpha 2 country code • Three-character ISO 3166-1 Alpha 3 country code • English country name <p>See ISO Country Codes and Module Support on page 1011 for a list of ISO codes.</p> |
| Data.FirmName | String | The company or firm name. |

| Parameter | Format | Description |
|--------------------|--------|--|
| Data.PostalCode | String | The postal code for the address in one of these formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 |
| Data.StateProvince | String | The state or province. |

Address Line Processing for U.S. Addresses

The input fields AddressLine1 through AddressLine4 are handled differently for U.S. addresses depending on whether the firm name extraction or urbanization code extraction options are enabled. If either of these options is enabled, ValidateAddressLoqate will look at the data in all four fields to validate the address and extract the requested data (firm name and/or urbanization code). If neither of these options is enabled, ValidateAddressLoqate uses only the first two non-blank address line fields in its validation attempt. The data in the other address line fields is returned in the output field AdditionalInputData. For example,

AddressLine1: A1 Calle A

AddressLine2:

AddressLine3: URB Alamar

AddressLine4: Precisely

In this address, if either firm name extraction or urbanization code extraction were enabled, ValidateAddressLoqate would examine all four address lines. If neither firm name extraction nor urbanization code extraction were enabled, ValidateAddressLoqate would examine AddressLine1 and AddressLine3 (the first two non-blank address lines) and attempt to validate the address using that data; the data in AddressLine4 would be returned in the output field AdditionalInputData.

Options

The following table lists the options that control the type of information returned by ValidateAddressLoqate.

Table 59: Output Data Options

| Parameter | Description |
|------------------------------------|---|
| Option.Database.Loqate | Specifies which database you want to use for validating international addresses. To specify a database for international address validation, select a database in the Database drop-down list. |
| Option.OutputFieldLevelReturnCodes | Specifies whether to include field-level result indicators. Field-level result indicators describe how ValidateAddressLoqate handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in HouseNumber.Result . For a complete listing of result indicator output fields, see Result Indicators on page 573. N No, do not output field-level return codes (default). Y Yes, output field-level return codes. |

| Parameter | Description |
|------------------------------|---|
| Option.OutputFormattedOnFail | <p data-bbox="685 380 1425 520">Specifies whether to return a formatted address when an address cannot be validated. The address is formatted using the preferred address format for the address's country. If this option is not selected, the output address fields are blank when ValidateAddressLoqate cannot validate the address.</p> <p data-bbox="685 541 1284 617"> N No, do not format failed addresses (default). Y Yes, format failed addresses. </p> <p data-bbox="685 646 1425 940">Formatted addresses are returned using the format specified by the Include a standard address, Include address line elements, and Include postal information check boxes. Note that if you select Include address line elements, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, select Include standardized input address elements.</p> <p data-bbox="685 961 1425 1014">If you check this option, you must select Include a standard address and/or Include address line elements.</p> <p data-bbox="685 1035 1425 1297">Formatted addresses are returned using the format specified by the OutputRecordType option. Note that if you specify OutputRecordType=E, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, specify OutputRecordType=I.</p> <p data-bbox="685 1318 1425 1350">If you specify Y, you must specify "A" and/or "E" for OutputRecordType.</p> <p data-bbox="685 1371 1425 1633">Formatted addresses are returned using the format specified by the Option.OutputRecordType option. Note that if you specify Option.OutputRecordType=E, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, specify Option.OutputRecordType=I.</p> <p data-bbox="685 1654 1425 1694">If you specify Y, you must specify "A" and/or "E" for Option.OutputRecordType.</p> |

| Parameter | Description | | | | |
|----------------------------|--|----------|--|----------|-----------------------------|
| Option.OutputAddressBlocks | <p data-bbox="685 380 1427 495">Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p data-bbox="685 514 1024 541">For example, this input address:</p> <p data-bbox="685 560 1084 705">AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p data-bbox="685 730 1065 758">Results in this address block output:</p> <p data-bbox="685 777 1203 863">AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p data-bbox="685 888 1427 1033">ValidateAddressLoqate formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p data-bbox="685 1056 902 1083">One of the following:</p> <table data-bbox="685 1094 1263 1167"><tr><td data-bbox="685 1094 711 1121">N</td><td data-bbox="824 1094 1263 1121">No, do not return address blocks. Default.</td></tr><tr><td data-bbox="685 1140 711 1167">Y</td><td data-bbox="824 1140 1109 1167">Yes, return address blocks.</td></tr></table> | N | No, do not return address blocks. Default. | Y | Yes, return address blocks. |
| N | No, do not return address blocks. Default. | | | | |
| Y | Yes, return address blocks. | | | | |

| Parameter | Description |
|-----------------------|--|
| Option.AmasFormatting | <p>Specifies that output address data is to be formatted using Address Matching Approval System (AMAS) conventions.</p> <p>This option causes Validate Address Loqate to use AMAS rules when standardizing an address. AMAS is an Australia Post program for enforcing addressing standards. For more information on the AMAS formatting conventions, refer to the Address Matching Approval System (AMAS) Handbook.</p> <p>This option modifies the output data as follows.</p> <ul style="list-style-type: none"> • Numeric fields are padded with zeros. This affects the following output fields: HouseNumber, HouseNumber2, PostalDeliveryNumber, and DPID. For example, if the input address is 298 New South Head Rd Double Bay NSW 2028, then the format of the HouseNumber field is changed from 298 to 00298. • If a match is not made, then all digits in the DPID field will be zero. For example, 00000000. • If a match is not made, then all return fields (parsed address elements) will be blank, except numeric fields which will contain all zeros. • The CCD field is not output. <p>Valid values are:</p> <p>N No, do not format the output data using AMAS conventions (default).</p> <p>Y Yes, format the output data using AMAS conventions.</p> <p>Note: When this option is selected, results will be returned with AMAS formatting regardless of selections made in the Acceptance level and Minimum match score fields.</p> |
| Option.OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p style="margin-left: 40px;">123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p style="margin-left: 40px;">123 MAIN ST MYTOWN FL 12345</p> |

| Parameter | Description |
|--------------------|--|
| Option.HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. ValidateAddressLoqate uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Somoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equitorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent And The Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Surivalue, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |

| Parameter | Description |
|----------------------------|--|
| Option.OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |
| Option.OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |

| Parameter | Description |
|--|--|
| Acceptance level Option.AcceptanceLevel | <p data-bbox="685 380 1425 491">Specifies the minimum verification level a record must reach to be considered successfully processed. The value in this field corresponds to the second character of the Address Verification Code, which is called "Post-Processed Verification Match Level":</p> <ul data-bbox="685 520 1425 1633" style="list-style-type: none"> <li data-bbox="685 520 1425 779">• 5—Delivery point (building or post box). The record will be passed or will have high confidence if ApartmentNumber, HouseNumber, Street, City, and StateProvince supplied in the input record match to the Loqate reference dataset. Will have moderate confidence if ApartmentNumber is correct but other remaining fields are incorrect, but in this case the Loqate engine should be able to identify the ApartmentNumber as ApartmentNumber is at a more granular level. It will have zero confidence if ApartmentNumber and other fields are unable to be parsed by the Loqate engine. <li data-bbox="685 789 1425 1024">• 4—Premise or building. The record will be passed or will have high confidence if House Number, Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if HouseNumber is correct but the other fields are not; however, in this case the Loqate engine should be able to identify the HouseNumber because HouseNumber is at a more granular level. It will have zero confidence if the HouseNumber and other fields are unable to be parsed by the Loqate engine. <li data-bbox="685 1035 1425 1270">• 3—Thoroughfare, road, or street. The record will be passed or will have high confidence if Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and State Province) are unable to be parsed by the Loqate engine. <li data-bbox="685 1281 1425 1516">• 2—Locality (city or town). The record will be passed or will have high confidence if both City and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate Engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and StateProvince) are unable to be parsed by the Loqate engine. <li data-bbox="685 1526 1425 1602">• 1—Administrative area (state or region). The record will be passed or will have high confidence if the StateProvince supplied in the input record matches the Loqate reference dataset. <li data-bbox="685 1612 1425 1633">• 0—None. This is equivalent to loosest match option. |

| Parameter | Description |
|--------------------------------------|---|
| Option.IsDuplicateHandlingMaskEnable | <p>Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options:</p> <p>S Selected by default. Preprocess the input and remove duplicates that occur in a single field.</p> <p>C Selected by default. Pre-process the input and remove duplicates across all fields.</p> <p>T Pre-process the input and remove duplicates in fields that are not standard address fields.</p> <p>F Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.</p> |
| Option.MinimumMatchScore | <p>Specifies a numeric value between 0 and 100 that indicates the degree to which Validate Address Loqate will change an address in order to obtain a match in the Loqate reference database. The lower the number, the greater amount of change is allowed. A value of 100 means that after parsing the input address is nearly identical to the validated address. A value of 0 means that the parsed input address may be completely changed in order to obtain a validated address.</p> |
| Option.KeepMultimatch | <p>Specifies whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> <p>For more information, see Returning Multiple Matches on page 564.</p> |
| Option.FailMultipleMatches | <p>Fails multiple addresses for those input addresses that have more than one possible match.</p> |

Returning Multiple Matches

If ValidateAddressLoqate finds multiple address in the postal database that are possible matches for the input address, you can have ValidateAddressLoqate return the possible matches. For example, the following address matches multiple addresses in the U.S. postal database:

PO BOX 1 New York, NY

Options

To return multiple matches, use the options described in the following table.

Table 60: Multiple Match Option

| Description/Valid Values |
|---|
| Indicates whether or not to return multiple address for those input addresses that have more than one possible match. |
| number between 1 and 10 that indicates the maximum number of addresses to return. The default value is 1. Note: The difference between <code>multipleMatch</code> and <code>multipleMatchLimit</code> is that a multiple match will return a failure if <code>multipleMatch</code> is set to true, whereas a multiple match will return one record if <code>multipleMatchLimit</code> is set to a value greater than 1. |
| To identify which output addresses are candidate addresses, you must. When you do this, records that are candidate addresses will have one or more "M" values in the field-level result indicators. |

Output

When you choose to return multiple matches, the addresses are returned in the address format you specify. For information on specifying address format, see [Options](#) on page 556. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information, see [Result Indicators](#) on page 573.

Match Score Threshold Options

There are two options for setting match score thresholds.

Note: These options are not available in the Validate Address Loqate user interface; they are located in the following file:

```
SpectrumDirectory/server/modules/loqate/env.properties
```

The **MatchScoreAbsoluteThreshold** option is used to specify the minimum match score a record must reach to be considered a candidate for matching. The default value is 60, and the maximum value is 100.

The **MatchScoreThresholdFactor** is a value that represents a factor of the highest matching result. This value is used as a cutoff for considering result candidates. The higher the value of the factor, the higher the chance of getting a good verification result. The default value is 95 and the maximum value is 100.

Response

The output from `ValidateAddressLoqate` contains various information depending on the output categories you select.

Standard Address Output

Standard address output consists of four lines of the address which correspond to how the address would appear on an address label. City, state/province, postal code, and other data is also included in standard address output. `ValidateAddressLoqate` returns standard address output for validated addresses if you. Standard address fields are always returned for addresses that could not be validated regardless of whether or not you. For non-validated addresses, the standard address output fields contain the address as it appeared in the input ("pass through" data). If you want `ValidateAddressLoqate` to standardize address according to postal authority standards when validation fails,.

Table 61: Standard Address Output

| Response Element | Description |
|----------------------------------|--|
| <code>AdditionalInputData</code> | Input data that could not be matched to a particular address component. For more information, see About Additional Input Data . |
| <code>AddressLine1-4</code> | If the address was validated, the first line of the validated and standardized address. If the address could not be validated, the first line of the input address without any changes. There can be up to four address block output fields: <code>AddressLine1</code> through <code>AddressLine4</code> . |
| <code>City</code> | The validated city name. |
| <code>Country</code> | The country in the format determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| <code>FirmName</code> | The validated firm or company name. |
| <code>PostalCode</code> | The validated ZIP Code™ or postal code. |

| Response Element | Description |
|------------------|---|
| PostalCode.AddOn | The 4-digit add-on part of the ZIP Code™. For example, in the ZIP Code™ 60655-1844, 1844 is the 4-digit add-on. |
| PostalCode.Base | The 5-digit ZIP Code™; for example 20706. |
| StateProvince | The validated state/province or its abbreviated value. |

Parsed Address Elements Output

Output addresses are formatted in the parsed address format if you. If you want `ValidateAddressLoqate` to return formatted data in the Parsed Address format when validation fails (that is, a normalized address),.

Note: If you want `ValidateAddressLoqate` to always return parsed input data regardless of whether or not validation is successful,. For more information, see [Parsed Input](#) on page 569.

Table 62: Parsed Address Output

| Response Element | Description |
|------------------|---|
| AddressBlock1-9 | <p>The AddressBlock output fields contain a formatted version of the standardized or normalized address as it would be printed on a physical mailpiece. Validate Address Global formats the address into address blocks using postal authority standards. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9. For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882</p> |

| Response Element | Description |
|--------------------|--|
| ApartmentLabel | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| ApartmentNumber | Apartment number, for example: 123 E Main St APT 3 |
| ApartmentNumber2 | Secondary apartment number, for example: 123 E Main St APT 3, 4th Floor Note: In this release, this field will always be blank. |
| Building | Descriptive name identifying an individual location. |
| City | Validated city name |
| Country | Country. Format is determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| County* | The smallest geographic data element within a country, for instance, USA County |
| FirmName | The validated firm or company name |
| HouseNumber | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional | Leading directional, for example: 123 E Main St Apt 3 |
| POBox | Post office box number. If the address is a rural route address, the rural route box number will appear here. |

| Response Element | Description |
|---------------------|---|
| PostalCode | Validated postal code. For U.S. addresses, this is the ZIP Code. |
| Principality * | The largest geographic data element within a country |
| StateProvince | Validated state or province name |
| StreetAlias | Alternate street name; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. for example: 123 E Main St Apt 3 |
| StreetName | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix | Street suffix, for example: 123 E Main St Apt 3 |
| Subcity* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet* | The dependent street or block data element within a country. For instance, UK Dependent Street . |
| TrailingDirectional | Trailing directional, for example: 123 Pennsylvania Ave NW |

*This is a subfield and may not contain data.

Parsed Input

The output can include the input address in parsed form. This type of output is referred to as "parsed input." Parsed input fields contain the address data that was used as input regardless of whether or not ValidateAddress validated the address. Parsed input is different from the "parsed address elements" output in that parsed address elements contain the validated address if the address could

be validated, and, optionally, the input address if the address could not be validated. Parsed input always contains the input address regardless of whether or not `ValidateAddress` validated the address.

To include parsed input fields in the output,.

Table 63: Parsed Input

| Response Element | Description |
|---------------------------------------|---|
| <code>ApartmentLabel.Input</code> | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| <code>ApartmentNumber.Input</code> | Apartment number, for example: 123 E Main St APT 3 |
| <code>City.Input</code> | Validated city name |
| <code>Country.Input</code> | Country. Format is determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| <code>County.Input*</code> | The smallest geographic data element within a country, for instance, USA County |
| <code>FirmName.Input</code> | The validated firm or company name |
| <code>HouseNumber.Input</code> | House number, for example: 123 E Main St Apt 3 |
| <code>LeadingDirectional.Input</code> | Leading directional, for example: 123 E Main St Apt 3 |
| <code>POBox.Input</code> | Post office box number. If the address is a rural route address, the rural route box number will appear here. |
| <code>PostalCode.Input</code> | Validated postal code. For U.S. addresses, this is the ZIP Code. |

| Response Element | Description |
|---------------------------|---|
| Principality.Input * | The largest geographic data element within a country |
| StateProvince.Input | Validated state or province name |
| StreetAlias.Input | Alternate street name; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. The base name is the name that applies to the entire street. For example: If StreetName is "N MAIN ST" the StreetAlias field would contain "MAIN" and the thoroughfare type, "ST", would be returned in the StreetSuffix field. |
| StreetName.Input | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix.Input | Street suffix, for example: 123 E Main St Apt 3 |
| Subcity.Input* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet.Input* | The dependent street or block data element within a country. For instance, UK Dependent Street . |
| TrailingDirectional.Input | Trailing directional, for example: 123 Pennsylvania Ave NW |

*This is a subfield and may not contain data.

Geocode Output

ValidateAddressLoqate returns the latitude/longitude, geocoding match code, dependent and double dependent localities, dependent thoroughfare, subadministrative and superadministrative areas, and the search distance as output. Match codes describe how well the geocoder matched the input address to a known address; they also describe the overall status of a match attempt. Search distance codes represent how close the geocode is to the actual physical location of an address.

Table 64: Geocode Address Output

| Response Element | Description |
|-------------------|--|
| Geocode.MatchCode | <p>This two-byte code reflects the status and level of geocode matching for an address. The first byte represents the geocoding status and is one of the following:</p> <p>A Multiple candidate geocodes were found to match the input address, and an average of these was returned</p> <p>I A geocode was able to be interpolated from the input addresses location in a range</p> <p>P A single geocode was found matching the input address</p> <p>U A geocode was not able to be generated for the input address</p> <p>The second byte represents the level of geocoding matching and is one of the following:</p> <p>5 Delivery point (post box or subbuilding)</p> <p>4 Premise or building</p> <p>3 Thoroughfare</p> <p>2 Locality</p> <p>1 Administrative area</p> <p>0 None</p> |
| Latitude | Eight-digit number in degrees and calculated to five decimal places (in the format specified). |
| Longitude | Eight-digit number in degrees and calculated to five decimal places (in the format specified). |
| SearchDistance | The radius of accuracy in meters, providing an indication of the probable maximum distance between the given geocode and the actual physical location. This field is derived from and dependent upon the accuracy and coverage of the underlying reference data. |

Table 65: City/Street/Postal Code Centroid Match Codes

| Element | Match Code |
|----------------------------|------------|
| Address Point | P4 |
| Address Point Interpolated | I4 |
| Street Centroid | A4/P3 |
| Postal Code/City Centroid | A3/P2/A2 |

Note: Geocode.Match.Code does not return two coordinates for a street segment (such as the beginning and ending of a portion of a street). Instead, with input resulting in return codes of I3 (interpolated to thoroughfare or street level, where no input premise number was provided), the complete street is used in the computation.

Result Indicators

Result indicators provide information about the kinds of processing performed on an address. There are two types of result indicators:

Record-Level Result Indicators

Record-level result indicators provide data about the results of ValidateAddressLoqate processing for each record, such as the success or failure of the match attempt, which coder processed the address, and other details. The following table lists the record-level result indicators returned by ValidateAddressLoqate.

Table 66: Record Level Indicators

| Response Element | Description |
|------------------|--|
| Confidence | The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct. For multiple matches, the confidence level is 0. For details about how this number is calculated, see Introduction to the Validate Address Confidence Algorithm on page 1036. |

| Response Element | Description |
|------------------|--|
| CouldNotValidate | <p>If no match was found, which address component could not be validated:</p> <ul style="list-style-type: none"> • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber <p>Note: More than one component may be returned, in a comma-separated list.</p> |
| MatchScore | <p>MatchScore provides an indication of the similarity between the input data and the closest reference data match. It is significantly different from Confidence in that Confidence indicates how much the input address changed to obtain a match, whereas the meaning of Match Score varies between U.S. and non-U.S. addresses.</p> <p>The int getFieldMatchscore (unit record, const char*) field is a decimal value between 0 and 100 that reflects the similarity between the identified input data and the closest reference data match. A result of 100 indicates that no changes other than alias, casing, or diacritic changes have been made to the input data. A result of 0 indicates that there is no similarity between the input data and closest reference data match.</p> <p>Note: The Validate Address Loqate and Advanced Matching Module components both use the MatchScore field. The MatchScore field value in the output of a dataflow is determined by the last stage to modify the value before it is sent to an output stage. If you have a dataflow that contains Validate Address Loqate and Advanced Matching Module components and you want to see the MatchScore field output for each stage, use a Transformer stage to copy the MatchScore value to another field. For example, Validate Address Loqate produces an output field called MatchScore and then a Transformer stage copies the MatchScore field from Validate Address Loqate to a field called AddressMatchScore. When the matcher stage runs it populates the MatchScore field with the value from the matcher and passes through the AddressMatchScore value from Validate Address Loqate.</p> |
| ProcessedBy | <p>Which address coder processed the address:</p> <p>LOQATE The Loqate coder processed the address.</p> |

| Response Element | Description | | | | |
|--------------------------|---|--------------------------|---|----------|---------|
| Status | <p>Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches.</p> <table> <tr> <td>null</td> <td>Success</td> </tr> <tr> <td>F</td> <td>Failure</td> </tr> </table> | null | Success | F | Failure |
| null | Success | | | | |
| F | Failure | | | | |
| Status.Code | <p>Reason for failure, if there is one.</p> <ul style="list-style-type: none"> • UnableToValidate | | | | |
| Status.Description | <p>Description of the problem, if there is one.</p> <table> <tr> <td>Address Not Found</td> <td>This value will appear if Status.Code=UnableToValidate.</td> </tr> </table> | Address Not Found | This value will appear if Status.Code=UnableToValidate. | | |
| Address Not Found | This value will appear if Status.Code=UnableToValidate. | | | | |

Field-Level Result Indicators

Field-level result indicators describe how ValidateAddressLoqate handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in **HouseNumber.Result**.

To enable field-level result indicators, .

The following table lists the field-level result indicators. If a particular field does not apply to an address, the result indicator may be blank.

Table 67: Field-Level Result Indicators

| Response Element | Description |
|------------------------|--|
| ApartmentLabel.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. |
| | R The apartment label is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| ApartmentNumber.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. addresses that are an EWS match will have a value of P. U.S. and Canadian addresses only. |
| | R The apartment number is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|------------------|---|
| City.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Hyphens missing or punctuation errors. Canadian addresses only. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. |
| | R The city is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| Country.Result | These result codes do not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| County.Result* | The smallest geographic data element within a country, for instance, USA County |
| FirmName.Result | C Corrected. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. |
| | U Unmatched. U.S. and Canadian addresses only. |
| | V Validated. The data was confirmed correct and remained unchanged from input. U.S. addresses only. |

| Response Element | Description |
|---------------------------|--|
| HouseNumber.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | O Out of range. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | R The house number is required but is missing from the input address. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| LeadingDirectional.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. Non-blank input was corrected to a non-blank value. U.S. addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. Does not apply to Canadian addresses. |

| Response Element | Description |
|-------------------|---|
| POBox.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | R The P.O. Box number is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| PostalCode.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. |
| | R The postal code is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|-----------------------|--|
| PostalCode.Type | <p>P The ZIP Code™ contains only PO Box addresses. U.S. addresses only.</p> <p>U The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.</p> <p>M The ZIP Code™ is for military addresses. U.S. addresses only.</p> <p>null The ZIP Code™ is a standard ZIP Code™.</p> |
| Principality.Result * | The largest geographic data element within a country |
| StateProvince.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only.</p> <p>R The state is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| StreetAlias.Result | An alternate name for a street; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. The base name is the name that applies to the entire street. For example: If StreetName is "N MAIN ST" the StreetAlias field would contain "MAIN" and the thoroughfare type,"ST", would be returned in the StreetSuffix field. |

| Response Element | Description |
|---------------------|--|
| StreetName.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. |
| | S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| StreetSuffix.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to U.S. addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| Subcity.Result* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet.Result* | The dependent street or block data element within a country. For instance, UK Dependent Street . |

| Response Element | Description |
|----------------------------|--|
| TrailingDirectional.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

*This is a subfield and may not contain data.

The AVC Code

The *Address Verification Code (AVC)* is an 11-byte code that is made up of accuracy indicators for addresses; the codes tell you the quality of the processing results and provide guidelines on how to correct the input data if necessary. Each individual address receives its own code. This code is automatically returned within your dataflow output. An example of an AVC is:

V44-I44-P6-100

An AVC has eight parts:

- Verification Status
- Post-Process Verification Match Level
- Pre-Process Verification Match Level
- Parsing Status
- Lexicon Identification Match Level
- Context Identification Match Level
- Postcode Status
- Matchscore

Verification Status

The level to which an address was verified.

- **V**—Verified. A complete match was made between the input data and a single record from the available reference data. For simple address validation, this is considered the best code to return.
- **P**—Partially verified. A partial match was made between the input data and a single record from the available reference data. This could mean that there is granular data for the address information that was provided, but additional information is required to return a full validation.
- **A**—Ambiguous. There are multiple addresses that could match the input.
- **U**—Unable to verify. This gets returned when there is not enough information to verify an address or when the input query is unreadable. The output fields will contain the input data.
- **R**—Reverted. The record could not be verified to the specified minimum acceptable level. This occurs when advanced options such as minimum reversion levels are set on a process. The output fields will contain the input data.
- **C**—Conflict. There is more than one close reference data match with conflicting values.

Post-Process Verification Match Level

The level to which the input data matches the available reference data after processing.

- **5**—Delivery point (building or post box). The record will be passed or will have high confidence if ApartmentNumber, HouseNumber, Street, City, and StateProvince supplied in the input record match to the Loqate reference dataset. Will have moderate confidence if ApartmentNumber is correct but other remaining fields are incorrect, but in this case the Loqate engine should be able to identify the ApartmentNumber as ApartmentNumber is at a more granular level. It will have zero confidence if ApartmentNumber and other fields are unable to be parsed by the Loqate engine.
- **4**—Premise or building. The record will be passed or will have high confidence if House Number, Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if HouseNumber is correct but the other fields are not; however, in this case the Loqate engine should be able to identify the HouseNumber because HouseNumber is at a more granular level. It will have zero confidence if the HouseNumber and other fields are unable to be parsed by the Loqate engine.
- **3**—Thoroughfare, road, or street. The record will be passed or will have high confidence if Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and State Province) are unable to be parsed by the Loqate engine.
- **2**—Locality (city or town). The record will be passed or will have high confidence if both City and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate Engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and StateProvince) are unable to be parsed by the Loqate engine.
- **1**—Administrative area (state or region). The record will be passed or will have high confidence if the StateProvince supplied in the input record matches the Loqate reference dataset.
- **0**—None. This is equivalent to loosest match option.

Pre-Process Verification Match Level

The level to which the input data matches the available reference data before processing.

- **5**—Delivery point (building or post box)
- **4**—Premise or building.
- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Parsing Status

The level to which an address was parsed.

- **I**—Identified and parsed. The input data has been identified and placed into components. For example, with "123 Kingston Av" Validate Address Loqate would be able to determine that "123" was a Premise Number, "Kingston" was the Thoroughfare Name, and "Av" or "Avenue" would be the Thoroughfare Type.
- **U**—Unable to parse. Validate Address Loqate was unable to identify and parse the input data. As with the "Unverified" verification status, the input data was incomplete or vague.

Lexicon Identification Match Level

The level to which the input data has some recognized form through the use of pattern matching (for instance, a numeric value could be a premise number) and lexicon matching (for example, "rd" could be Thoroughfare Type "road"; "London" could be a locality, and so on).

- **5**—Delivery point (building or post box)
- **4**—Premise or building.
- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Context Identification Match Level

The level to which the input data can be recognized based on the context in which it appears. This is the least accurate form of matching and is based on identifying a word as a particular address element. For example, input could be determined to be a thoroughfare because it was preceded by something that could be a premise and followed by something that could be a locality, the latter items being identified through a match against the reference data or the lexicon.

- **5**—Delivery point (building or post box)
- **4**—Premise or building.

- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Postcode Status

The level to which a postal code was verified.

- **P8**—PostalCodePrimary and PostalCodeSecondary verified.
- **P7**—PostalCodePrimary verified, PostalCodeSecondary added or changed.
- **P6**—PostalCodePrimary verified.
- **P5**—PostalCodePrimary verified with small change.
- **P4**—PostalCodePrimary verified with large change.
- **P3**—PostalCodePrimary added.
- **P2**—PostalCodePrimary identified by lexicon.
- **P1**—PostalCodePrimary identified by context.
- **P0**—PostalCodePrimary empty.

Match Score

A numeric value between 0 and 100 representing the similarity between the identified input data and the output data for the record. A result of 100 means that no changes other than additions, alias, casing, or diacritic changes have been made to the input data. A result of 0 means there is no similarity between the input data item and the output data provided.

AMAS Output

The following table lists the standard fields that are output by ValidateAddressLoqate.

Table 68: Output Fields

| Response Element | Description |
|------------------|-------------------------------------|
| Barcode | Standard barcode based on the DPID. |
| | F Failure (no barcode found) |
| | 20-digit number Success |

| Response Element | Description |
|------------------|---|
| DPID | The Delivery Point Identifier. An eight-digit number from the Australia Post Postal Address File that uniquely identifies a mail delivery point, such as a street address. Note: This field will contain "00000000" for Australian addresses that are not AMAS-verified and will be empty for non-Australian addresses. |
| FloorNumber | The floor/level number, for example: 123 E Main St Apt 3, 4th Floor |
| FloorType | The floor/level type, for example: 123 E Main St Apt 3, 4th Floor |
| PostalBoxNum | The postal delivery number, for example: PO Box 42 |

Spectrum Universal Name

OpenNameParser

OpenNameParser breaks down personal and business names and other terms in the name data field into their component parts. These parsed name elements are then subsequently available to other automated operations such as name matching, name standardization, or multiple-record name consolidation.

OpenNameParser does the following:

- Determines the type of a name in order to describe the function that the name performs. Name entity types are divided into two major groups: personal names and business names. Within each of these major groups are subgroups.
- Determines the form of a name in order to understand which syntax the parser should follow for parsing. Personal names usually take on a natural (signature) order or a reverse order. Business names are usually ordered hierarchically.
- Determines and labels the component parts of a name so that the syntactical relationship of each name part to the entire name is identified. The personal name syntax includes prefixes, first, middle, and last name parts, suffixes, and account description terms, among other personal name parts. The business name syntax includes the firm name and suffix terms.
- Parses conjoined personal and business names and either retains them as one record or splits them into multiple records. Examples of conjoined names include "Mr. and Mrs. John Smith" and "Baltimore Gas & Electric dba Constellation Energy".

- Parses output as records or as a list.
- Assigns a parsing score that reflects the degree of confidence that the parsing is correct.

Resource URL

JSON endpoint:

```
http://server:port/rest/OpenNameParser/results.json
```

XML endpoint:

```
http://server:port/rest/OpenNameParser/results.xml
```

Example with JSON Response

The following example requests a JSON response:

```
http://myserver:8080/rest/OpenNameParser/results.json?  
Data.Name=John+Williams+Smith
```

The JSON returned by this request would be:

```
{"output_port": [{  
  "Name": "John Williams Smith",  
  "CultureCodeUsedToParse": "",  
  "FirstName": "John",  
  "LastName": "Smith",  
  "MiddleName": "Williams",  
  "Names": [],  
  "IsParsed": true,  
  "IsPersonal": true,  
  "IsConjoined": false,  
  "IsReverseOrder": false,  
  "IsFirm": false,  
  "NameScore": 100,  
  "user_fields": []  
}]}
```

Example with XML Response

The following example requests an XML response:

```
http://myserver:8080/rest/OpenNameParser/results.xml?  
Data.Name=John+Williams+Smith
```

The XML returned by this request would be:

```
<ns2:xml.OpenNameParserResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/OpenNameParser">
  <ns2:output_port>
    <ns2:Result>
      <ns2:Name>John Williams Smith</ns2:Name>
      <ns2:CultureCodeUsedToParse/>
      <ns2:FirstName>John</ns2:FirstName>
      <ns2:LastName>Smith</ns2:LastName>
      <ns2:MiddleName>Williams</ns2:MiddleName>
      <ns2:Names/>
      <ns2:IsParsed>true</ns2:IsParsed>
      <ns2:IsPersonal>true</ns2:IsPersonal>
      <ns2:IsConjoined>false</ns2:IsConjoined>
      <ns2:IsReverseOrder>false</ns2:IsReverseOrder>
      <ns2:IsFirm>false</ns2:IsFirm>
      <ns2:NameScore>100</ns2:NameScore>
      <ns2:user_fields/>
    </ns2:Result>
  </ns2:output_port>
</ns2:xml.OpenNameParserResponse>
```

Request

Parameters for Input Data

Table 69: Open Name Parser Input

| Parameter | Description | | | | | | | | |
|---------------------|--|---------------------|---------------------------|-----------|---------|-----------|----------|-----------|-----------|
| Data.CultureCode | <p>The culture of the input name data. The options are listed below.</p> <table border="0"> <tr> <td>Null (empty)</td> <td>Global culture (default).</td> </tr> <tr> <td>de</td> <td>German.</td> </tr> <tr> <td>es</td> <td>Spanish.</td> </tr> <tr> <td>ja</td> <td>Japanese.</td> </tr> </table> <p>Note: If you added your own domain using the Open Parser Domain Editor, the cultures and culture codes for that domain are also valid.</p> | Null (empty) | Global culture (default). | de | German. | es | Spanish. | ja | Japanese. |
| Null (empty) | Global culture (default). | | | | | | | | |
| de | German. | | | | | | | | |
| es | Spanish. | | | | | | | | |
| ja | Japanese. | | | | | | | | |
| Data.Name | The name you want to parse. This field is required. | | | | | | | | |

Options

Parameters for Parsing Options

The following table lists the options that control the parsing of names.

Table 70: Open Name Parser Parsing Options

| Parameter | Description |
|---------------------------------------|---|
| Option.ParseNaturalOrderPersonalNames | <p>Specifies whether to parse names where the is in the order Title, First Name, Middle Name, Last Name, and Suffix.</p> <p>true Parse personal names that are in natural order.</p> <p>false Do not parse names that are in natural order.</p> |
| Option.ParseReverseOrderPersonalNames | <p>Specifies whether to parse names where the last name is specified first.</p> <p>true Parse personal names that are in reverse order.</p> <p>false Do not parse names that are in reverse order.</p> |
| Option.ParseConjoinedNames | <p>Specifies whether to parse conjoined names.</p> <p>true Parse conjoined names.</p> <p>false Do not parse conjoined names.</p> |
| Option.SplitConjoinedNames | <p>Specifies whether to separate names containing more than one individual into multiple records, for example, <i>Bill & Sally Smith</i>.</p> <p>true Split conjoined names.</p> <p>false Do not split conjoined names.</p> |
| Option.ParseBusinessNames | <p>Specifies whether to parse business names.</p> <p>true Parse business names.</p> <p>false Do not parse business names.</p> |

| Parameter | Description |
|--------------------------|--|
| Option.OutputAsList | <p>Specifies whether to return the parsed name elements in a list form.</p> <p>true Return the parsed elements in a list form.</p> <p>false Do not return the parsed elements in a list form.</p> |
| Option.ShortcutThreshold | <p>Specifies how to balance performance versus quality. A faster performance will result in lower quality output; likewise, higher quality will result in slower performance. When this threshold is met, no other processing will be performed on the record.</p> <p>Specify a value from 0 to 100. The default is 100.</p> |

Parameters for Culture Options

The following table lists the options that control name cultures.

Table 71: Open Name Parser Cultures Options

| Parameter | Description |
|-----------------------|--|
| Option.DefaultCulture | <p>Specifies which culture(s) you want to include in the parsing grammar. Global Culture is the default selection.</p> <p>Specify cultures by specifying the two-character culture code in a comma-separated list in priority order. For example, to attempt to parse the name using the Spanish culture first then Japanese, you would specify:</p> <p><code>es, ja, ,</code></p> |

Parameters for Advanced Options

The following table lists the advanced options for name parsing.

Table 72: Open Name Parser Advanced Options

| Option | Description |
|---|--|
| Option.NaturalOrderPersonalNamesDomain | Specifies the domain to use when parsing natural order personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |
| Option.NaturalOrderPersonalNamesPriority | Specify a number between 1 and 5 that indicates the priority of the natural order personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run. Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned. |
| Option.ReverseOrderPersonalNamesDomain | Specifies the domain to use when parsing reverse order personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |
| Option.ReverseOrderPersonalNamesPriority | Specify a number between 1 and 5 that indicates the priority of the reverse order personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run. Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned. |
| Option.NaturalOrderConjoinedPersonalNamesDomain | Specifies the domain to use when parsing natural order conjoined personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |

| Option | Description |
|---|---|
| Option.NaturalOrderConjoinedPersonalNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the natural order conjoined personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |
| Option.ReverseOrderConjoinedPersonalNamesDomain | <p>Specifies the domain to use when parsing reverse order conjoined personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer.</p> |
| Option.ReverseOrderConjoinedPersonalNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the reverse order conjoined personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |
| Option.BusinessNamesDomain | <p>Specifies the domain to use when parsing business names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer.</p> |

| Option | Description |
|------------------------------|---|
| Option.BusinessNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the business names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |

Response

Table 73: Open Name Parser Output

| Response Element | Format | Description |
|---|--------|--|
| AccountDescription | String | An account description that is part of the name. For example, in "Mary Jones Account # 12345", the account description is "Account#12345". |
| Names | String | A hierarchical field that contains a list of parsed elements. This field is returned when you check the Output results as list box under Parsing Options. |
| Fields Related to Names of Companies | | |
| FirmConjunction | String | Indicates that the name of a firm contains a conjunction such as "d/b/a" (doing business as), "o/a" (operating as), and "t/a" (trading as). |
| FirmName | String | The name of a company. For example, <code>Precisely</code> . |
| FirmSuffix | String | The corporate suffix. For example, "Co." and "Inc." |

| Response Element | Format | Description |
|---|--------|---|
| IsFirm | String | Indicates that the name is a firm rather than an individual. |
| Fields Related to Names of Individual People | | |
| Conjunction | String | Indicates that the name contains a conjunction such as "and", "or", or "&". |
| CultureCode | String | The culture codes contained in the input data. |
| CultureCodeUsedToParse | String | Identifies the culture-specific grammar that was used to parse the data. Null (empty) Global culture (default). de German. es Spanish. ja Japanese. Note: If you added your own domain using the Open Parser Domain Editor, the cultures and culture codes for that domain will appear in this field as well. |
| FirstName | String | The first name of a person. |
| GeneralSuffix | String | A person's general/professional suffix. For example, MD or PhD. |
| IsParsed | String | Indicates whether an output record was parsed. Values are true or false. |
| IsPersonal | String | Indicates whether the name is an individual rather than a firm. Values are true or false. |
| IsReverseOrder | String | Indicates whether the input name is in reverse order. Values are true or false. |

| Response Element | Format | Description |
|--|--------|---|
| LastName | String | The last name of a person. Includes the paternal last name. |
| LeadingData | String | Non-name information that appears before a name. |
| MaturitySuffix | String | A person's maturity/generational suffix. For example, Jr. or Sr. |
| MiddleName | String | The middle name of a person. |
| Name. | String | The personal or firm name that was provided in the input. |
| NameScore | String | Indicates the average score of known and unknown tokens for each name. The value of NameScore will be between 0 and 100, as defined in the parsing grammar. 0 is returned when no matches are returned. |
| SecondaryLastName | String | In Spanish parsing grammar, the surname of a person's mother. |
| TitleOfRespect | String | Information that appears before a name, such as "Mr.", "Mrs.", or "Dr." |
| TrailingData | String | Non-name information that appears after a name. |
| Fields Related to Conjoined Names | | |
| Conjunction2 | String | Indicates that a second, conjoined name contains a conjunction such as "and", "or", or "&". |
| Conjunction3 | String | Indicates that a third, conjoined name contains a conjunction such as "and", "or", or "&". |
| FirmName2 | String | The name of a second, conjoined company. For example, Baltimore Gas & Electric dba Constellation Energy. |

| Response Element | Format | Description |
|------------------|--------|--|
| FirmSuffix2 | String | The suffix of a second, conjoined company. |
| FirstName2 | String | The first name of a second, conjoined name. |
| FirstName3 | String | The first name of a third, conjoined name. |
| GeneralSuffix2 | String | The general/professional suffix for a second, conjoined name. For example, MD or PhD. |
| GeneralSuffix3 | String | The general/professional suffix for a third, conjoined name. For example, MD or PhD. |
| IsConjoined | String | Indicates that the input name is conjoined. An example of a conjoined name is "John and Jane Smith." |
| LastName2 | String | The last name of a second, conjoined name. |
| LastName3 | String | The last name of a third, conjoined name. |
| MaturitySuffix2 | String | The maturity/generational suffix for a second, conjoined name. For example, Jr. or Sr. |
| MaturitySuffix3 | String | The maturity/generational suffix for a third, conjoined name. For example, Jr. or Sr. |
| MiddleName2 | String | The middle name of a second, conjoined name. |
| MiddleName3 | String | The middle name of a third, conjoined name. |
| TitleOfRespect2 | String | Information that appears before a second, conjoined name, such as "Mr.", "Mrs.", or "Dr." |

| Response Element | Format | Description |
|------------------|--------|--|
| TitleOfRespect3 | String | Information that appears before a third, conjoined name, such as "Mr.", "Mrs.", or "Dr." |

SOAP

Spectrum Enterprise Tax

AssignGeoTAXInfo

AssignGeoTAXInfo identifies the tax districts that apply to a given address. Specifically, AssignGeoTAXInfo returns the following information about an address:

- Latitude/longitude coordinates
- FIPS state codes and county codes
- County names
- MCD/CCD codes and names
- CBSA/CSA codes and names
- Place FIPS and GNIS codes and names
- Incorporated or unincorporated status codes
- Cross-reference tax keys
- Result indicators
- Optionally, the relationship of an address to user-defined polygons

AssignGeoTAXInfo optionally includes enhanced tax jurisdiction information for an address, including:

- **Insurance premium districts**—Areas designated for the collection of taxes imposed on insurance policy premiums based on the policy holder's address. Insurance premium districts are created by state governments.
- **Payroll tax districts**—Areas designated for the collection of taxes imposed on employers to support state or local government facilities and services based on the employee's and/or employer's address. Examples include taxes collected for districts to pay for schools, police, or other services. Payroll tax districts are created by state or local governments.

- **Payroll system tax codes**—Codes that represent specific jurisdictions that collect payroll tax. Using payroll system tax codes has advantages over using the payroll tax district information returned by Assign GeoTAX Info:
 - AssignGeoTAXInfo uses an additional database to determine payroll tax codes, resulting in more accurate payroll tax determination.
 - Many payroll systems use specific codes to determine withholding amounts. Since you can customize the payroll tax codes returned by AssignGeoTAXInfo, you can set up a process where AssignGeoTAXInfo returns the exact payroll tax codes required by your payroll system instead of returning jurisdictional IDs that must then be translated into the codes used by your system.
- **Special purpose tax districts**—Areas designated for the collection of taxes imposed on residents to support specialized services for residents of the district based on the resident's address. Examples include services such as sewer service, transit service, or water resources. Special purpose tax districts are created by legislative action, court action, or public referendums. This optional information requires the use of boundary files which require an additional license. Contact your Precisely sales representative for more information.
- **Sales and Use Tax Rates**—Using the optional Precisely Sales and Use Tax Rate file, AssignGeoTAXInfo can return sales and use tax rates for each of the assigned tax jurisdictions as well as the total tax rate for the assigned locations.

AssignGeoTAXInfo is part of Spectrum Enterprise Tax.

Resource URL

```
http://server:port/soap/AssignGeoTaxInfo
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ass="http://www.precisely.com/spectrum/services/AssignGeoTAXInfo"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <ass:AssignGeoTAXInfoRequest>
      <ass:input_port>
        <ass:Address>
          <ass:AddressLine1>1 Global View</ass:AddressLine1>
          <ass:City>Troy</ass:City>
          <ass:StateProvince>NY</ass:StateProvince>
          <ass:PostalCode>12180</ass:PostalCode>
        </ass:Address>
      </ass:input_port>
    </ass:AssignGeoTAXInfoRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

```

    </soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:AssignGeoTAXInfoResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/AssignGeoTAXInfo">

      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>100.0</ns3:Confidence>
          <ns3:ProcessedBy>GTX</ns3:ProcessedBy>
          <ns3:Census.MatchCode>S</ns3:Census.MatchCode>
          <ns3:Census.MatchLevel>Street</ns3:Census.MatchLevel>
          <ns3:County.Code>083</ns3:County.Code>
          <ns3:County.Name>Rensselaer</ns3:County.Name>
          <ns3:StateCode>36</ns3:StateCode>
          <ns3:LatLong>42.683028-073.702968</ns3:LatLong>
          <ns3:LatLong.MatchCode>R</ns3:LatLong.MatchCode>
          <ns3:LatLong.MatchLevel>Rooftop</ns3:LatLong.MatchLevel>

          <ns3:Latitude>42.683028</ns3:Latitude>
          <ns3:Longitude>-073.702969</ns3:Longitude>
          <ns3:State.Abbreviation>NY</ns3:State.Abbreviation>
          <ns3:Place.Code>00000</ns3:Place.Code>

          <ns3:Place.IncorporatedFlag>Uninc</ns3:Place.IncorporatedFlag>
          <ns3:AddressLine1>1 GLOBAL VW</ns3:AddressLine1>
          <ns3:City>TROY</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>121808371</ns3:PostalCode>

          <ns3:AddressMatch.MatchCode>S80</ns3:AddressMatch.MatchCode>

          <ns3:AddressMatch.LocationCode>AS0</ns3:AddressMatch.LocationCode>
          <ns3:AddressMatch.LastLine>TROY, NY
12180-8371</ns3:AddressMatch.LastLine>
          <ns3:AddressMatch.Zip>12180</ns3:AddressMatch.Zip>
          <ns3:AddressMatch.Zip4>8371</ns3:AddressMatch.Zip4>
          <ns3:AddressMatch.GenRC>S</ns3:AddressMatch.GenRC>

          <ns3:AddressMatch.DataTypename>TOMTOM</ns3:AddressMatch.DataTypename>

          <ns3:MCD.DistanceToBorder>000002938</ns3:MCD.DistanceToBorder>

          <ns3:Place.DistanceToBorder>00000000</ns3:Place.DistanceToBorder>
          <ns3:GNISCode>00000000</ns3:GNISCode>
        </ns3:Address>

```

```

    </ns3:output_port>
  </ns3:AssignGeoTAXInfoResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

The following table provides information about the format of AssignGeoTAXInfo input.

| Parameter | Format | Description |
|--------------|-----------------|---|
| AddressLine1 | String [100] | First address line |
| AddressLine2 | String [100] | Second address line |
| AddressLine3 | String [100] | Third address line |
| AddressLine4 | String [100] | Fourth address line |
| BufferWidth | String [10] | <p>Specifies the width of the polygon buffers to use for Boundary File processing. The buffer width is used to determine if a point is close to the edge of a polygon. The output field BufferRelation indicates whether or not the point is within the polygon's buffer area. For more information, see Buffering on page 1008.</p> <p>This field overrides the value specified in the DefaultBufferWidth option. Specify the border width in the units specified by the DistanceUnits option.</p> <p>If you do not specify a buffer width in this input field, the default is used.</p> |
| City | String [50] | City name |

| Parameter | Format | Description |
|-----------------|--------------|---|
| Country | String [var] | The country where the address resides. The data you enter in this field has no impact on processing. It is simply passed through to output. Note: Only US addresses are supported. |
| FirmName | String [var] | The company or firm name. |
| PostalCode | String [9] | Nine-digit ZIP Code |
| StateProvince | String [50] | The state where the address resides. The data you enter in this field has no impact on processing. It is simply passed through to output. |
| UserBufferWidth | Long [10] | Specifies the width of the polygon buffers to use for User-Defined Boundary File processing. The buffer width is used to determine if a point is close to the edge of a polygon. The output field BufferRelation indicates whether or not the point is within the polygon's buffer area. For more information, see Buffering on page 1008. This field overrides the value specified in the DefaultBufferWidth option. Specify the border width in the units specified by the DistanceUnits option. If you do not specify a buffer width in this input field, the default is used. |

Matching Options

Matching options control the address search methodology and match results handling returned by AssignGeoTAXInfo.

| Parameter | Description | | | | |
|--|---|----------|---|----------|--|
| Optional files: The following options enable the database resource(s) to use in the search process. | | | | | |
| UseGeoTaxAuxiliaryFile | Specifies whether or not AssignGeoTAXInfo should attempt a match to the GeoTAX Auxiliary file. The GeoTAX Auxiliary file contains new addresses that have not yet been added to the Master File. <table> <tr> <td>Y</td> <td>Use the GeoTAX Auxiliary file for matching. (default)</td> </tr> <tr> <td>N</td> <td>Do not use the GeoTAX Auxiliary file for matching.</td> </tr> </table> | Y | Use the GeoTAX Auxiliary file for matching. (default) | N | Do not use the GeoTAX Auxiliary file for matching. |
| Y | Use the GeoTAX Auxiliary file for matching. (default) | | | | |
| N | Do not use the GeoTAX Auxiliary file for matching. | | | | |

| Parameter | Description |
|---|---|
| UseAuxiliaryFile | <p>Specifies whether to attempt a match to a User Auxiliary file. User Auxiliary files are user-defined files that override results from the master files in street-level matching.</p> <p>Y Use the User Auxiliary file for matching.</p> <p>N Do not use the User Auxiliary file for matching. (default)</p> |
| UseStateProvidedFile | <p>Specifies whether to attempt a match to the state-supplied file. Use this option in combination with <code>FileSearchOrder</code> to specify a state-supplied file to use.</p> <p>State-supplied files are provided by individual state governments. By matching to the state-supplied files, you can remain compliant with tax jurisdiction assignment requirements mandated by new federal and state laws, such as the Mobile Telecommunications Sourcing Act and the Florida state Communications Services Tax Simplification Law.</p> <p>There are two supported file formats: the Florida-native format and the national TS-158 format (ANSI Transaction Set No. 158). The state of Florida provides address files in both the TS-158 and its own native format.</p> <p>If this option is enabled, the address is first matched to the state supplied file. If a state match cannot be found, the master files are used to attempt a match.</p> <p>Y Use the State-supplied file for matching.</p> <p>N Do not use the State-supplied file for matching. (default)</p> <p>Note: You must install the appropriate State-supplied file to use these options. For instructions, see the <i>Spectrum Technology Platform Installation Guide</i>.</p> |
| FileSearchOrder | <p>Specifies which state-supplied file to use. This option only takes effect if you specify <code>UseStateProvidedFile=Y</code>. One of the following:</p> <p>FLOnly Use only the Florida-native formatted file. (default)</p> <p>TSOnly Use only the TS-158 formatted file.</p> |
| UseRelaxedSecondaryMatching | <p>Specifies whether input addresses with secondary information are matched to records without secondary information. This option applies only to Florida-native files.</p> <p>Y Use relaxed secondary matching.</p> <p>N Do not use relaxed secondary matching. (default)</p> |
| <p>Address Searching and Matching: These options can be enabled for use in the address search and match processes.</p> | |

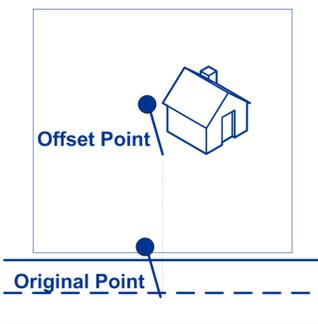
| Parameter | Description |
|-----------------------------|--|
| GsMatchMode | <p>Match modes determine the leniency used to make a match between your input and the reference database. Select a match mode based on the quality of your input and your desired output. For example, if you have an input database that is prone to errors, you may want to select the relaxed match mode.</p> <p>0 - Exact Requires a very tight match. This restrictive mode generates the fewest match candidates, which decreases the processing time. When using this mode, ensure that your input is very clean; free of misspellings and incomplete addresses.</p> <p>1 - Close Requires a close match and generates a moderate number of match candidates. (default)</p> <p>2 - Relaxed Allows a loose match and generates the most match candidates, which increases the processing time and results in more multiple matches. Use this mode if you are not confident that your input is clean; free of misspellings and incomplete addresses. This is the only mode that does not respect the street parity when making an address match.</p> |
| GsSearchArea | <p>The search area options allow for searching the address' finance area or an expanded area specified by distance.</p> <p>1 Searches the entire finance area for a match. A finance area is a region defined by the U.S. Postal Service and typically consists of a set of contiguous ZIP Codes.(default)</p> <p>2 Searches the area specified by the radius in miles. The search area can be extended up to a 99-mile radius from the centroid of the input ZIP Code to assist in finding a match when the input address contains limited or inaccurate city or ZIP Code information. The expanded area is confined to within the state's borders.</p> |
| GsSearchRadius | <p>Radius for search area.</p> <p>1-99 miles Search radius. (default = 0 miles)</p> |
| GsEnableFirstLetterExpanded | <p>Looks for the correct first letter of a street address if the first letter is missing or incorrect. Spectrum Enterprise Tax searches through the alphabet looking for possible correct first letters to complete the street address.</p> <p>Note: This feature is disabled by default and cannot be enabled in Exact mode.</p> <p>Y Enable first letter change matches.</p> <p>N Do not allow first letter change matches. (default)</p> |

| Parameter | Description |
|------------------------|--|
| GsEnableRangedAddress | <p>Matches to a house range input. 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. When this feature is enabled, the address range is geocoded to the interpolated mid-point of the range.</p> <p>Note: This feature is disabled by default and cannot be enabled in Exact mode.</p> <p>Y Allow address range matches.</p> <p>N Do not allow address range matches. (default)</p> |
| GsAlternateLookup | <p>This option allows specifying the preferred way to match when both an address and firm name are provided. The matching method can be set to match to the address rather than the firm name or vice versa. If neither are specified, the default matching method is to match to the address line only.</p> <p>1 Searches for street name, but if there isn't a match, will use the firm name.</p> <p>2 Looks up the firm name, but if there isn't a match, will use the street name.</p> <p>3 Searches only street records. (default)</p> |
| GsMultiMatchResolution | <p>A multi-match occurs when multiple equally-scored matches are found in either the Points or Streets files and cannot be resolved to a single best candidate. There are several choices for handling a multi-match outcome:</p> <p>N No matches are returned. (default)</p> <p>R Return the first match candidate in the list.</p> <p>A The information for all the match candidates is returned.</p> |

Geocoding Options

Geocoding is the process of determining the latitude/longitude coordinates of a given address. Address coordinates are used as the basis for determining the tax jurisdictions for an address. Geocoding options control how AssignGeoTAXInfo determines address latitude/longitude coordinates.

| Parameter | Description |
|---|-------------|
| Latitude/Longitude placement: These options can be set for the geocode result. | |

| Parameter | Description |
|---------------|---|
| LatLongOffset | <p>Indicates the offset distance in feet from the street center line.</p> <p>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. Since the building represented by an address is not on the street itself, you do not want the geocode for an address to be a point on the street. Instead, you want the geocode to represent the location of the building which sits next to the street. For example, an offset of 40 feet means that the geocode will represent a point 40 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. The diagram below shows an offset point in relation to the original point.</p>  <p>0 No offset. (default)</p> <p>20 Twenty feet offset from street centerline.</p> <p>40 Forty feet offset from street centerline. (recommended)</p> <p>60 Sixty feet offset from street centerline.</p> |

| Parameter | Description |
|-----------|--|
| Squeeze | <p>Specifies if the street end points should be "squeezed" when determining the geocode of an address in street-level matching. When <code>Squeeze</code> is enabled, both street and end points are moved closer to the center of the segment by 50 feet. The diagram below compares the end points of a street segment to the squeezed end points of a street segment.</p>  <p>Y Apply squeeze.</p> <p>N Do not apply squeeze. (default)</p> |

| | |
|---------------|---|
| LatLongFormat | <p>Indicates the desired format for the returned latitude/longitude. One of the following:</p> <p>PreZero Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W.</p> <p>PreZeroDecimal Decimal degrees using directional indicator. For example, 090.000000N180.000000W. (default)</p> <p>Decimal Decimal degrees using signed latitude/longitude. For example, 90.000000-180.000000.</p> <p>DecimalAssumed Decimal degrees using signed latitude/longitude (no decimal point). For example, 90000000-180000000.</p> <p>DegMinSec Degrees, minutes, seconds. For example, 90 00 00N180 00 00W.</p> |
|---------------|---|

Expanded Geocoding options: These options enable additional geocoding functionality.

| Parameter | Description |
|-----------------------------------|---|
| GsEnableAddressPointInterpolation | <p>Address point interpolation uses a patented process that improves upon regular street segment interpolation by inserting point data into the interpolation process.</p> <p>Note: This feature is only for use with point-level geocoding.</p> <p>A match is first attempted using the loaded points data. If an exact point match is found in the points data, then searching ceases and the point match is returned. If an exact point match was not found, Spectrum Enterprise Tax attempts to find high and low boundary address points to use for address point interpolation.</p> <p>Y Enable address point interpolation.</p> <p>N Disable address point interpolation. (default)</p> |
| Minimum geocode quality | |
| GsEnableGeographicFallback | <p>The default search does not perform a search of geographic centroids. When enabled, the Geographic Fallback feature locates the first city, county and/or state centroid, and then matches from the set of possible matches found.</p> <p>Y If a definitive match cannot be made, then return the next higher level geographic centroid.</p> <p>N Disable geographic fallback feature. (default)</p> |

| Parameter | Description |
|--|--|
| GsEnableStreetCentroid | <p>If an input street address cannot be found using the street number and name, Spectrum Enterprise Tax then searches the input ZIP Code or city/state for the closest match. If Spectrum Enterprise Tax is able to locate the street, it returns a geocode along the matched street segment rather than the geocode for the entered ZIP Code or ZIP + 4.</p> <p>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) returns. For example, if you enter Main St and there are both an E Main St and a W Main St within the input ZIP Code, then an E030 returns and the location code returned is reflective of the input ZIP Code. The location code returned begins with a "C" when matched to a single street segment, indicated by E029. The Spectrum Enterprise Tax does not change the street name on the output address.</p> <p>Y If a street or point match cannot be made, then return a street level centroid.</p> <p>N Do not return a street level centroid if a match cannot be made. (default)</p> <p>Note: This feature should only be used for exception processing or research. It should not be used in a production process.</p> |
| <hr/> <p>Boundary matching: These options can be set when matching to a boundary file such as SPD, IPD, PAY, Place and MCD or user-defined.</p> <hr/> | |
| DistanceUnits | <p>Specifies the units in which to measure distance. One of the following:</p> <p>Feet Distances are measured in feet. (default)</p> <p>Meters Distances are measured in meters.</p> |
| DefaultBufferWidth | <p>Specifies the buffer width to use for tax district boundary files. The tax district boundary files are the Special Purpose District (SPD) file, the Insurance Premium District (IPD) file, the Payroll Tax District (PAY) file, and Place and MCD files.</p> <p>The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>For more information about buffers, see Buffering on page 1008.</p> |

| Parameter | Description |
|------------------------|---|
| DefaultUserBufferWidth | <p>Specifies the buffer width to use for user-defined boundary files. Specify the distance in the units of measurement specified in the Distance units option. For information about buffers, see Buffering on page 1008. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>Note: To use buffers, the user-defined boundary file must support buffers.</p> |

Output Data Options

Data options control the data returned by AssignGeoTAXInfo.

| Parameter | Description |
|------------------------|--|
| GeoTAXOutputRecordType | <p>Select one or more of the following to obtain the type of data you want returned. If you do not want all of the fields in a record type returned, you can specify the individual fields to return by specifying them in the <code>OutputFields</code> option.</p> <ul style="list-style-type: none"> • C—Census • L—Latitude/Longitude • T—Tax Jurisdiction • U—User-defined boundary file • W—Payroll System Tax Codes • X—Auxiliary File • B—PB Software Sales and Use Tax Rate file <p>You can also specify one, and only one, of the following:</p> <p>I Insurance Premium Tax District (IPD)</p> <p>R Payroll Tax District (PAY)</p> <p>S Special Purpose Tax District (SPD)</p> <p>For a description of the fields in each output group, see Response on page 611.</p> <p>Note: If you specify <code>w</code>, to obtain the best payroll system tax code match possible.</p> |

| Parameter | Description |
|--------------|---|
| TaxKey | <p>If you use third-party tax compliance software from Vertex or Sovos, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following:</p> <p>N Do not return either the Sovos or Vertex jurisdiction codes (default).</p> <p>T Return the Sovos jurisdiction code for the address.</p> <p>V Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Precisely.</p> <p>Note: The Special Purpose District data is needed to achieve the best results from this option.</p> |
| TaxRate | <p>Indicates the sales and use tax rate type to return or none:</p> <p>N Do not return sales and use tax rates. (default)</p> <p>G Return the General sales and use tax rates.</p> <p>A Return the Automotive sales and use tax rates.</p> <p>C Return the Construction sales and use tax rates.</p> <p>M Return the Medical sales and use tax rates.</p> <p>Note: The Special Purpose District data is needed to achieve the best results from this option.</p> |
| OutputFields | <p>Indicates the individual output fields you want returned. You can use this field instead of the Output Record Type to limit the output to those fields that are important to your current data needs.</p> <p>For a list of the fields included in each data type, see Response on page 611.</p> |

Output Format

Output format options control how AssignGeoTAXInfo formats output data.

| Parameter | Description |
|--------------|--|
| OutputCasing | <p>Specifies the casing of these output fields: County.Name, CBSA.Name, MCD.Name, Place.Name, IPDn.DistrictName, PAYn.DistrictName, SPDn.DistrictName, and PTCn.PayrollDescription.</p> <p>One of the following:</p> <p>M Returns the output in mixed case (default). For example: Rensselaer.</p> <p>U Returns the output in upper case. For example: RENSSELAER.</p> |

Response

Address Match Results

The table below lists the fields returned from the address matching and geocoding process.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose the associated output data options (for example, census or tax jurisdiction data output options). Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------|--|---|
| AddressMatch.DataTypeName* | 20 | <p>Indicates the file from which the match was obtained. One of the following:</p> <ul style="list-style-type: none"> • USPS • TIGER • TOMTOM - Streets • NAVTEQ - Streets • TOMTOM_POINT • CENTRUS_POINT • NAVTEQ_POINT • MASTER LOCATION - Master Location Data • STATE_FILE • USER_AUXILIARY • LANDMARK_AUXILIARY |
| AddressMatch.Firm* | 41 | The name of the business if the address is a business address. |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AddressMatch.GenRC* | 2 | <p>General Return Code indicating the type of match.</p> <p>5 ZIP Code match</p> <p>9 ZIP+4 Code match</p> <p>A User Auxiliary file match</p> <p>C Street Centroid match</p> <p>F Geographic Fallback match</p> <p>G State-supplied file match</p> <p>I Intersection match</p> <p>L Landmark Auxiliary file match</p> <p>M Multiple match (multi-match)</p> <p>O Input Latitude/Longitude coordinates match</p> <p>P Address point match</p> <p>S Street address match</p> <p>U GeoTAX Auxiliary file match</p> <p>X Aborted processing or expired database</p> <p>Blank Did not match</p> |
| AddressMatch.Lastline* | 61 | The complete matched last address line (city, state, and postal code). |
| AddressMatch.LocationCode* | 5 | <p>The Location Code indicates the methodology used to complete the geocode and may also provide some information about the quality of the geocode.</p> <p>For the list of location codes, see Location Codes.</p> |
| AddressMatch.MatchCode* | 5 | <p>The Match Code indicates the portions of the address that matched or did not match to the reference file.</p> <p>For the list of match codes, see Match Codes.</p> |
| AddressMatch.NumCandidates* | 2 | When there are multiple equally-scored matches, returns the number of multiple match candidates found. |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------------|--|--|
| AddressMatch.PBKey | 14 | <p>A unique address identifier that is returned when an address match is made using the Master Location Database. The pbKey™ unique identifier is used as a lookup key to a GeoEnrichment database, in order to return attribute data for the match.</p> <p>The AddressMatch.PBKey field has "P" as the leading character, for example: P00001XSF11F.</p> |
| AddressMatch.Urbanization* | 31 | Urbanization name. Used for addresses in Puerto Rico. |
| AddressMatch.Zip* | 6 | The matched address five-digit ZIP Code. |
| AddressMatch.Zip4* | 5 | The matched address four-digit ZIP Code extension. |
| Census.MatchCode* | 2 | <p>The level of match obtained against the databases.</p> <p>5 ZIP Code level match</p> <p>9 ZIP + 4 Code level match</p> <p>A User Auxiliary file match</p> <p>C Street centroid match</p> <p>F Geographic fallback match</p> <p>G State-supplied file match</p> <p>I Intersection match</p> <p>L Landmark Auxiliary file match</p> <p>M Multiple match (multi-match)</p> <p>O Input latitude/longitude coordinates match</p> <p>P Address point match</p> <p>S Street address match</p> <p>U GeoTAX Auxiliary file match</p> <p>X Aborted processing or expired database</p> <p>Blank Did not match</p> |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Census.MatchLevel* | 19 | The level of match obtained against the databases. |
| | | AbortedExpiredData Aborted processing or expired database |
| | | Aux2 GeoTAX Auxiliary file match |
| | | Auxiliary Auxiliary street match |
| | | FallbackGeographic Geographic fallback match |
| | | Gov State file address match |
| | | Intersection Intersection match |
| | | LatLonInput Input latitude/longitude coordinates match |
| | | LandmarkAux Landmark Auxiliary file match |
| | | MultiMatch Multiple match |
| | | Point Address point match |
| | | Street Street address match |
| | | StreetCentroid Street centroid match |
| | | ZIP ZIP Code level match |
| | | ZIP+4 ZIP + 4 Code level match |
| | | NoMatch Did not match |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| Confidence* | 4 | <p>Indicates the confidence in the output provided; from 0 to 100. The higher the score, the higher the confidence in the match. Calculated based on the match results for individual output fields, using the following algorithm:</p> $\text{Census.MatchCode} + \text{LatLong.StreetMatchCode} + \text{LatLong.MatchCode}$ <p>The maximum confidence score is 100, so if this calculation results in a value greater than 100, the Confidence score is returned as 100.</p> <p>The following values are used:</p> <hr/> <ul style="list-style-type: none"> • Census.MatchCode <ul style="list-style-type: none"> • 5 = 45 • 9 = 75 • A = 85 • C = 55 • F = 45 • G = 85 • I = 85 • L = 85 • M = 0 • O = 85 • P = 100 • S = 85 • U = 85 • X = 0 • null = 0 <hr/> <ul style="list-style-type: none"> • LatLong.StreetMatchCode <ul style="list-style-type: none"> • H = 5 • L = 15 • S = -10 • Z = -5 • null = 0 <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|---|--|--|
| | | <ul style="list-style-type: none"> • LatLong.MatchCode <ul style="list-style-type: none"> • 2 = 0 • 4 = 10 • B = 0 • C = 0 • I = 10 • L = 15 • O = 15 • R = 15 • S = -10 • T = -2 • U = 15 • Z = -5 • null = -100 |
| County.Code * | 4 | Extracted from the Census.BlockCode. |
| County.Name * | 26 | Name of the county. |
| GNISCode * | 10 | Unique nine-digit Geographic Names Information System (GNIS) code. |
| Standardized input address fields - for field information, see Input Address on page 628. | | |
| MCD.DistanceToBorder * | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb file. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|---|
| MCD.PointStatus* | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched address point to the polygon defined by the Cousub.txt file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.Code* | 6 | Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.ClassCode* | 3 | Place class code. Place class codes are used to determine the proper taxing jurisdictions |
| Place.DistanceToBorder* | 10 | Returns the distance between the matched address point to the polygon defined by the Place.txt file. |
| Place.IncorporatedFlag* | 8 | <p>Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Inc Incorporated place code.</p> <p>Uninc Unincorporated place code.</p> <p>Unknown Incorporation status unknown.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|---|
| Place.LastAnnexedDate* | 8 | Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information. |
| Place.LastUpdatedDate* | 8 | Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place. |
| Place.LastVerifiedDate* | 8 | Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. |
| Place.Name* | 41 | The name of the "place" where the address is located. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.PointStatus* | 2 | Returns the result for a comparison between the matched address point to the polygon defined by the Place.txb file. For more information on buffers, see Buffering on page 1008. Note: The buffer distance for Place.txb is internally set to zero and cannot be modified. P The point is in the polygon. I The point is in the buffer area inside the polygon. B The point is in the buffer area and outside of the polygon. blank Polygon not found. |
| ProcessedBy* | 4 | Always returns GTX. |
| State.Abbreviation* | 3 | Two-character state abbreviation. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| StateCode* | 3 | Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode. |

Auxiliary File

The table below lists the output fields that contain Auxiliary file data. To include Auxiliary file data in the output, set `GeoTAXOutputRecordType = X`. The following table lists the output fields that contain tax jurisdiction data.

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AuxiliaryData.AuxiliaryFile | 301 | Data retrieved as a result of an auxiliary match from the user-defined area of the auxiliary file. |
| AuxiliaryData.StateFile | 201 | Data retrieved as a result of a state match. Data content and format vary depending on the state file used. |

Census

The census output fields contains census information from the U.S. Census, including Minor Civil Divisions (MCDs) and Census County Division (CCD) names and codes. MCDs are the primary political or administrative divisions of a county, representing many kinds of legal entities with a variety of governmental and administrative functions. CCDs are established in states where there are no legally established MCDs. The Census Bureau recognizes MCDs in 28 states and has established CCDs in 21 states. The District of Columbia has no primary divisions, and the city of Washington, DC is considered equivalent to an MCD for data presentation purposes.

Census data also contains the Federal Information Processing Standards (FIPS) codes for each state and county. The FIPS State Code and the FIPS County Code are both used by the Census Bureau to identify these geographic units.

The table below lists the output fields that contain census data. To include census data in the output, set `GeoTAXOutputRecordType = C`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include census data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|---|
| Census.Block | 4 | Census Block ID. |
| Census.BlockGroup | 2 | Census Block Group code. |
| Census.MatchCode* | 2 | The level of match obtained against the databases. 5 ZIP Code level match 9 ZIP + 4 Code level match A User Auxiliary file match C Street centroid match F Geographic fallback match G State-supplied file match I Intersection match L Landmark Auxiliary file match M Multiple match (multi-match) O Input latitude/longitude coordinates match P Address point match S Street address match U GeoTAX Auxiliary file match X Aborted processing or expired database Blank Did not match |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Census.MatchLevel* | 19 | The level of match obtained against the databases. AbortedExpiredData Aborted processing or expired database Aux2 GeoTAX Auxiliary file match Auxiliary Auxiliary street match FallbackGeographic Geographic fallback match Gov State file address match Intersection Intersection match LatLonInput Input latitude/longitude coordinates match LandmarkAux Landmark Auxiliary file match MultiMatch Multiple match Point Address point match Street Street address match StreetCentroid Street centroid match ZIP ZIP Code level match ZIP+4 ZIP + 4 Code level match NoMatch Did not match |
| Census.Tract | 7 | Six-digit tract number extracted from the Census.BlockCode. |
| County.Code* | 4 | Extracted from the Census.BlockCode. |
| County.Name* | 26 | Name of the county. |
| MCD.Code | 6 | Minor Civil Division/Census County Division (MCD/CCD) Code. |
| MCD.Name | 41 | Minor Civil Division/Census County Division (MCD/CCD) name. |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------|--|---|
| MCD.PointStatus* | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched address point to the polygon defined by the Cousub.txt file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| MCD.DistanceToBorder* | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txt file. |
| MCD.Confidence | 4 | Provides the percentage overlap of the geocode to the MCD polygon layer. The returned percentage value describes the probability that the point falls in the MCD. |
| CBSA.Code | 6 | Core Based Statistical Area (CBSA) code. |
| CBSA.Name | 76 | Core Based Statistical Area (CBSA) name. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| CBSA.MetroFlag | 2 | Indicates if the CBSA is a "Metropolitan Statistical Area" or a "Micropolitan Statistical Area". Y Metropolitan Statistical Area - A Core Based Statistical Area associated with at least one urbanized area that has a population of at least 50,000. The Metropolitan Statistical Area comprises the central county or counties containing the core, plus adjacent outlying counties having a high degree of social and economic integration with the central county as measured through commuting. N Micropolitan Statistical Area - A Core Based Statistical Area associated with at least one urban cluster that has a population of at least 10,000, but less than 50,000. The Micropolitan Statistical Area comprises the central county or counties containing the core, plus adjacent outlying counties having a high degree of social and economic integration with the central county as measured through commuting. |
| CBSAD.Code | 6 | Core Based Statistical Area Division (CBSAD) code. |
| CBSAD.Name | 73 | Core Based Statistical Area Division (CBSAD) name. |
| CSA.Code | 4 | Combined Statistical Area (CSA) code. |
| CSA.Name | 78 | Combined Statistical Area (CSA) name. |
| State.Abbreviation* | 3 | Two-character state abbreviation. |
| StateCode* | 3 | Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode. |

Latitude/Longitude

The table below lists the output fields that contain latitude and longitude data. Latitude/Longitude data contains the coordinates for the address and additional information about how the latitude and

longitude for the address was determined. To include latitude/longitude data in the output, set `GeoTAXOutputRecordType = L`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include latitude/longitude data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| Latitude | 8 | Seven-digit number in degrees and calculated to four decimal places (in the format you specified). |
| Latitude.Directional | 2 | Latitude directional. N North S South |
| LatLong | 23 | Returned latitude/longitude, in the format you specified (up to 22 alphanumeric characters). |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|---|
| LatLong.MatchCode | 2 | Latitude/Longitude General Return Code. Denotes the level for which the geocode was determined. |
| | 2 | ZIP + 2 centroid |
| | 4 | ZIP + 4 Code centroid |
| | B | Block group centroid |
| | C | City centroid |
| | I | Intersection |
| | L | Match using the Landmark Auxiliary file |
| | O | Latitude/longitude was input |
| | R | Address-level based on street address |
| | S | State centroid |
| | T | Census tract centroid |
| | U | Address-level match using the GeoTAX Auxiliary file |
| | Z | ZIP Code centroid based on a five-digit ZIP code |
| | null | No latitude/longitude determined |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | If <code>AddressMatch.GenRC</code> is "P" (point match), then the following are possible values: |
| | 0 | Latitude/Longitude coordinates from User Dictionary. |
| | 2 | Latitude/Longitude coordinates from Parcel Centroid. |
| | 4 | Latitude/Longitude coordinates from Address Point. |
| | 5 | Latitude/Longitude coordinates from Structure Centroid. |
| | 7 | Latitude/Longitude coordinates from manually-placed Point. |
| | 8 | Latitude/Longitude coordinates from Front Door Point. |
| | 9 | Latitude/Longitude coordinates from Driveway Offset Point. |
| | A | Latitude/Longitude coordinates from Street Access Point. |
| | B | Latitude/Longitude coordinates from Base Parcel Point. |
| | C | Latitude/longitude coordinates from Backfill Address Point. |
| | D | Latitude/longitude coordinates from Virtual Address Point. |
| | E | Latitude/longitude coordinates from Interpolated Address Point. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|--|
| LatLong.MatchLevel* | 14 | A description of the value returned in the <code>LatLong.MatchCode</code> field. |
| | | ZIP+2 ZIP + 2 centroid |
| | | ZIP+4 ZIP + 4 centroid |
| | | Block Block group centroid |
| | | CityCentroid City centroid |
| | | Intersection Intersection match |
| | | LandmarkAux Match using the Landmark Auxiliary file |
| | | LatLonInput Input Latitude/Longitude coordinates was used |
| | | Rooftop Exact address match |
| | | StateCentroid State centroid |
| | | Tract Census tract centroid |
| | | Auxiliary Address-level match using the GeoTAX Auxiliary file |
| | | ZIP ZIP Code centroid |
| | | Point Point-level match. One of the following: <ul style="list-style-type: none"> • User Dictionary • Parcel Centroid • Address Point • Structure Centroid • Manually-placed Point • Front Door Point • Driveway Offset Point • Street Access Point • Base Parcel Point • Backfill Address Point • Virtual Address Point • Interpolated Address Point |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------|--|--|
| LatLong.StreetMatchCode* | 2 | Output street address return code. H House number not found on street L Latitude/longitude not determined on auxiliary match S Street not found in ZIP Code Z ZIP Code not found in street address database N Street-level matching option not selected null The street was successfully matched |
| LatLong.StreetMatchLevel* | 16 | Street level match used to determine the latitude/longitude FullMatch Successful match HouseNotFound House number not found on street LatLongNotFound Latitude/longitude not determined on auxiliary match StreetNotFound Street not found in ZIP Code ZipNotFound ZIP Code not found in street address database NotUsed Street-level matching option not selected |
| Longitude | 8 | Seven-digit number in degrees and calculated to four decimal places (in the format specified). |
| Longitude.Directional | 2 | Longitude directional. E East W West |

Input Address

AssignGeoTAXInfo always returns the input address as part of the output. Any changes to the address information resulting from the address cleansing process will be returned to these fields.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---------------------------|
| AddressLine1 | 101 | Input address line 1. |
| AddressLine2 | 101 | Input address line 2. |
| AddressLine3 | 101 | Input address line 3. |
| AddressLine4 | 101 | Input address line 4. |
| City | 51 | Input address city. |
| Country | 25 | Input address country. |
| FirmName | 101 | Input address firm name. |
| PostalCode | 10 | Input address postal code |
| StateProvince | 51 | Input address state. |

Parsed Elements

The parsed elements output fields contain standard street address line information as individual units, such as street suffixes (for example AVE, ST, or RD) and leading directionals (for example N and SE).

To include parsed elements in the output, assign the desired output fields to the `OutputFields` parameter.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|--|
| AddressMatch.UnitType | 5 | The type of unit, such as apartment, suite, or lot. |
| AddressMatch.UnitNumber | 12 | Apartment number. For example, 123 E Main St APT 3 |
| AddressMatch.HouseNumber | 12 | Building number for the address. |
| AddressMatch.PreDirectional | 3 | Leading directional. For example, 123 E Main St Apt 3 |
| AddressMatch.Street | 41 | The name of the street, not including any directionals or suffixes. For example, the word "Main" in this address: 123 E Main St Apt 3 |
| AddressMatch.StreetType | 5 | The street type of the matched location. For example, AVE for Avenue. |
| AddressMatch.PostDirectional | 3 | Street directional that follows the street name. For example, the "N" in this address: 456 Washington N . |

Payroll System Tax Code

The table below lists the output fields that contain Payroll System Tax Code (PTC) data. For more information about payroll tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set `GeoTAXOutputRecordType = W`.

Note: AssignGeoTAXInfo returns up to six payroll tax codes per address.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| NumberPTCsFound | 2 | The number of payroll tax codes found for this address. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|--|
| PTCn.MatchCode | 2 per PTC | <p>Indicates the level of match obtained for the address. In order from most specific match to least, the possible match codes are:</p> <p>P The address was matched to a specific Payroll District ID. This is the most specific match.</p> <p>G The address was matched to a GNIS Code.</p> <p>F The address was matched to a county's FIPS code.</p> <p>S The address was matched to a state's FIPS code. This is the least specific match.</p> |
| PTCn.PayrollCode | 16 per PTC | A code that represents a taxing authority in a payroll application. This is a user-defined code. The specific codes are determined by the payroll application that utilizes the data returned by AssignGeoTAXInfo. |
| PTCn.PayrollDescription | 41 per PTC | A description of the purpose of this payroll code. |
| PTCn.PayrollFlag | 7 per PTC | A user-defined flag from the PTC database. |
| StateCounty | 33 | The state abbreviation and county name. |

Tax Jurisdiction

Tax jurisdiction data contains information about the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state; or, an area recognized as significant because it is located in an incorporated municipality. Places are used to determine tax jurisdiction.

The table below lists the output fields that contain tax jurisdiction data. To include tax jurisdiction data in the output, set `GeoTAXOutputRecordType = T`.

Note: Fields denoted by an asterisk "*" are always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| Confidence.SurfaceType | 3 | <p>Indicates the confidence surface type. Setting a non-zero buffer width enables confidence generation. To determine a confidence level, a confidence surface is first generated. The confidence surface provides the smallest possible area wherein an address is likely to be located.</p> <p>0 Undefined</p> <p>1 The search failed - the address was not found.</p> <p>2 Intersection confidence surface generated.</p> <p>3 Interpolated street segment.</p> <p>4 Point-level match.</p> <p>5 State confidence surface generated.</p> <p>6 County confidence surface generated.</p> <p>7 City confidence surface generated.</p> <p>8 Reserved</p> <p>9 A ZIP Code confidence surface generated.</p> <p>10 A ZIP+2 confidence surface generated.</p> <p>11 A ZIP+4 confidence surface generated.</p> <p>12 Reserved</p> <p>13 A street centroid confidence surface generated.</p> |
| GeoTAXKey | 10 | <p>The value in this field varies depending on the option you specified in the <code>TaxKey</code> option:</p> <p>If you specified <code>T</code>, <code>GeoTAXKey</code> contains the proprietary codes used in Sovos tax compliance software. You can use this code in your Sovos application to find out the tax rate for the jurisdiction. The Sovos jurisdiction code formats are as follows:</p> <ul style="list-style-type: none"> • Sovos SUT - 2-digit SUT state code, 5-digit ZIP Code, 2-digit SUT geocode • Sovos TWE - variable-length TWE geocode <p>If you specified <code>V</code>, <code>GeoTAXKey</code> contains the proprietary Vertex[®] jurisdiction code (comprised of a two-digit Vertex[®] state code, three-digit FIPS county code, and four-digit Vertex[®] city code). You can use this code in your Vertex[®] application to find out the tax rate for the jurisdiction.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GeoTAXKey.MatchCode | 2 | <p>Return code denoting the level of match obtained against the Vertex or Sovos cross reference files.</p> <p>E Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name.</p> <p>P Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code.</p> <p>A Alternate match using two fields: ZIP Code, FIPS place name.</p> <p>N Record is default coded based on valid state code.</p> <p>null No matching record found.</p> |
| GeoTAXKey.MatchLevel | 12 | <p>A description of the value returned in the <code>GeoTAXKey.MatchCode</code> field.</p> <p>Exact Exact match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>Partial Partial match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>Alternate Alternate match. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>DefaultCode Record is default coded. See description in <code>GeoTAXKey.MatchCode</code>.</p> <p>NoMatch No matching record found.</p> |
| GNISCode* | 10 | Unique nine-digit Geographic Names Information System (GNIS) code. |
| Place.ClassCode* | 3 | Place class code. Place class codes are used to determine the proper taxing jurisdictions |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------------|--|--|
| Place.Code * | 6 | Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |
| Place.IncorporatedFlag * | 8 | Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. Inc Incorporated place code. Uninc Unincorporated place code. Unknown Incorporation status unknown. |
| Place.LastAnnexedDate * | 8 | Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information. |
| Place.LastUpdatedDate * | 8 | Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place. |
| Place.LastVerifiedDate * | 8 | Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. |
| Place.Name * | 41 | The name of the "place" where the address is located. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|--|
| Place.PointStatus* | 2 | <p>Returns the result for a comparison between the matched address point to the polygon defined by the Place.txb file.</p> <p>For more information on buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Place.txb is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.DistanceToBorder* | 10 | Returns the distance between the matched address point to the polygon defined by the Place.txb file. |
| Place.Confidence | 4 | Provides the percentage overlap of the geocode to the Place polygon layer. The returned percentage value describes the probability that the point falls in the specified Place. |

User-Defined Boundary File

The table below lists the output fields that contain data returned from user-defined boundary files. To include this data in the output, set `GeoTAXOutputRecordType = U`.

Note: AssignGeoTAXInfo can return up to 10 user-defined areas for each input address.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|--|
| NumberUserBoundariesFound | 3 | The number of user-defined polygons found for the address. |
| UserBoundary <i>n</i> .BoundaryDescription | 51 per User Boundary | A description of the polygon. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------------------|--|--|
| UserBoundary.n.BoundaryID | 11 per User Boundary | The ID of the polygon as specified in the user-defined boundary file. |
| UserBoundary.n.BufferRelation | 2 per User Boundary | <p>Indicates where in the polygon the address resides in relation to the edge of the area. Buffer width is specified by the option <code>DefaultUserBufferWidth</code> or by the input field <code>BufferWidth</code> .</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the polygon at a distance from the edge that is greater than the specified buffer width. I The address is inside the polygon but is close to the edge. B The address is outside the polygon but is close to the edge. <p>For more information, see Buffering on page 1008.</p> |
| UserBoundary.n.DistanceToBorder | 10 per User Boundary | Indicates the distance from the address to the border of the polygon. The distance is in the units specified by the option <code>DistanceUnits</code> . |
| UserBoundary.n.SupplementalBoundaryID | 11 per User Boundary | A supplemental ID as specified in the user-defined boundary file. |
| UserBoundary.n.BoundaryConfidence | 4 per User Boundary | Provides the percentage overlap of the geocode to the User-defined boundary polygon layer. The returned percentage value describes the probability that the point falls in the User-defined boundary area. |

Insurance Premium Tax Districts

The table below lists the output fields that contain Insurance Premium Tax Districts (IPD) data. To include IPD data in the output, set `GeoTAXOutputRecordType = I`.

Note: `AssignGeoTAXInfo` returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberIPDsFound | 3 | The number of Insurance Premium Tax Districts found for the address |
| IPD <i>n</i> .BoundaryBuffer.BufferRelation | 2 per IPD | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| IPD <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per IPD | Indicates the distance from the address to the border of the district. |
| IPD <i>n</i> .BoundaryConfidence | 4 per IPD | Provides the percentage overlap of the geocode to the IPD boundary polygon layer. The returned percentage value describes the probability that the point falls in the IPD boundary area. |
| IPD <i>n</i> .DistrictID | 11 per IPD | IPD ID. |
| IPD <i>n</i> .DistrictName | 61 per IPD | IPD name. |
| IPD <i>n</i> .DistrictType | 7 per IPD | IPD district type. |
| IPD <i>n</i> .UpdateDate | 7 per IPD | IPD update date (MMYYYY). |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| IPDn.VersionDate | 7 per IPD | IPD compiled date (MMYYYY). |
| IPDn.Notes | 21 per IPD | Tax code descriptions. For example: 01, 33, A, B |
| IPDn.ChangeDate | 7 per IPD | IPD change date. |
| IPDn.EffectiveDate | 7 per IPD | MMDDYY - Identifies when district becomes active - State supplied For example: 010108 |
| IPDn.ExpirationDate | 7 per IPD | MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009 |
| IPDn.FireRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.FireFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semi colon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.CasualtyRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.CasualtyFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.VehicleRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.VehicleFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MarineRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MarineFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.HealthRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.HealthFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.LifeRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.LifeFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.OtherRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.OtherFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MinimumRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.MinimumFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

Payroll Tax Districts

The table below lists the output fields that contain Payroll Tax District (PAY) data. For more information on payroll tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set GeoTAXOutputRecordType = R.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--------------------------|
| NumberPAYsFound | 3 | Number of PAYs returned. |

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| PAY <i>n</i> .BoundaryBuffer.BufferRelation | 2 per PAY | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| PAY <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per PAY | Indicates the distance from the address to the border of the district. The distance is in the units specified by the option <code>DistanceUnits</code> . |
| PAY <i>n</i> .BoundaryConfidence | 4 per PAY | Provides the percentage overlap of the geocode to the PAY boundary polygon layer. The returned percentage value describes the probability that the point falls in the PAY boundary area. |
| PAY <i>n</i> .DistrictID | 11 per PAY | PAY district ID. |
| PAY <i>n</i> .DistrictName | 61 per PAY | PAY district name. |
| PAY <i>n</i> .DistrictType | 7 per PAY | PAY district type. |
| PAY <i>n</i> .ID | 11 per PAY | PAY ID. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------------------|--|---|
| PAY <i>n</i> .MunicipalEMSTax | 2 per PAY | <p>PAY municipality emergency municipal services tax.</p> <p>The values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .MunicipalIncomeTax | 2 per PAY | <p>PAY municipality income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X None</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictEMSTax | 2 per PAY | <p>PAY school district emergency municipal services tax.</p> <p>The Values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictIncomeTax | 2 per PAY | <p>PAY school district income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X N</p> <p>The values for Ohio are:</p> <p>R Resident</p> <p>X None</p> <p>All other states are null.</p> |

Special Purpose Tax Districts

The table below lists the output fields that contain Special Purpose Tax Districts (SPD) data. For more information on special purpose tax districts, see [AssignGeoTAXInfo](#) on page 597. To include this data in the output, set `GeoTAXOutputRecordType = S`.

Note: `AssignGeoTAXInfo` returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| <code>NumberSPDsFound</code> | 3 | Number of SPDs returned. |
| <code>SPDn.BoundaryBuffer.BufferRelation</code> | 2 per SPD | <p>Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The address is inside the district at a distance from the edge that is greater than the specified buffer width. I The address is inside the district but is close to the edge. This indicates that the address is in the buffer area. B The address is outside the district but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| <code>SPDn.BoundaryBuffer.DistanceToBorder</code> | 10 per SPD | Indicates the distance from the address to the border of the district. The distance is in the units specified by the option <code>DistanceUnits</code> . |
| <code>SPDn.BoundaryConfidence</code> | 4 per SPD | Provides the percentage overlap of the geocode to the SPD boundary polygon layer. The returned percentage value describes the probability that the point falls in the SPD boundary area. |
| <code>SPDn.CompiledDate</code> | 7 per SPD | SPD compiled date. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|-----------------------------|
| SPD <i>n</i> .DistrictCode | 4 per SPD | 3-digit district type code. |
| SPD <i>n</i> .DistrictName | 61 per SPD | SPD name. |
| SPD <i>n</i> .DistrictNumber | 6 per SPD | SPD district number. |
| SPD <i>n</i> .EffectiveDate | 7 per SPD | SPD effective date. |
| SPD <i>n</i> .UpdateDate | 7 per SPD | SPD update date. |
| SPD <i>n</i> .VersionDate | 7 per SPD | SPD version date. |

Sales and Use Tax Rates

The table below lists the output fields that contain the sales and use tax rate data.

To include tax rate data in the output, set `GeoTAXOutputRecordType = B`.

To select the tax rate type, set `TaxRate` to one of the following:

- N** Do not return sales and use tax rates. (default)
- G** Return the General sales and use tax rates.
- A** Return the Automotive sales and use tax rates.
- C** Return the Construction sales and use tax rates.
- M** Return the Medical sales and use tax rates.

Note: You must be a licensed user of the Precisely Sales and Use Tax Rate file to use this feature.

The following table describes the Sales and Use Tax Rate output fields.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| TaxRate.RC | 2 | Tax Rate return code denoting the level of match obtained against the Precisely Sales and Use Tax Rate file: E Exact match, using all 5 fields P Partial match, using 4 fields A Alternate match, using 3 fields N Record is default-coded based on valid state code. Blank No matching PB Software Sales and Use Tax Rate record found. |
| Municipal.SalesTaxRate | 11 | Municipality sales tax rate for the selected tax rate type. |
| County.SalesTaxRate | 11 | County sales tax rate for the selected tax rate type. |
| State.SalesTaxRate | 11 | State sales tax rate for the selected tax rate type. |
| SPD n .SalesTaxRate | 11 per SPD | Sales tax rate for up to 10 Special Purpose Districts (SPD). |
| TaxRate.SalesTotal | 11 | The sum of the individual Municipal, County, State and SPD sales tax rates. |
| Municipal.UseTaxRate | 11 | Municipality use tax rate for the selected tax rate type. |
| County.UseTaxRate | 11 | County use tax rate for the selected tax rate type. |
| State.UseTaxRate | 11 | State use tax rate for the selected tax rate type. |
| SPD n .UseTaxRate | 11 per SPD | Use tax rate for up to 10 Special Purpose Districts (SPD). |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| TaxRate.UseTotal | 11 | The sum of the individual Municipal, County, State and SPD use tax rates. |

Error Reporting

The table below defines the error reporting output fields.

Note: Fields denoted by an asterisk "*" are always included in the output. Contents returned when available; otherwise, left blank.

| Response Element | Max. Field Length with null terminator | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|--|---|--------------|-----------------------------|----------|-----------------------------|-----------|--------------------------|-----------|---------------------------|----------|---------------|----------|------------------------------------|----------|---------------------------------|----------|-------------------|----------|--------------------|----------|-------------------------------|----------|--|----------|----------------------|----------|----------------------------------|----------|------------------|----------|---------------|
| GTX.ErrorCode* | 3 | <p>This field contains a return code if the GeoTAX engine experiences an abnormal termination.</p> <p>Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <p>The first character indicates the file (or set of files affected).</p> <table border="0"> <tr> <td>Blank</td> <td>Matcher terminated normally</td> </tr> <tr> <td>A</td> <td>User Auxiliary file problem</td> </tr> <tr> <td>CE</td> <td>coubsub.txb file problem</td> </tr> <tr> <td>CI</td> <td>Confidence engine problem</td> </tr> <tr> <td>D</td> <td>Boundary file</td> </tr> <tr> <td>F</td> <td>User-defined boundary file problem</td> </tr> <tr> <td>G</td> <td>Address Matching engine problem</td> </tr> <tr> <td>L</td> <td>Licensing problem</td> </tr> <tr> <td>S</td> <td>State file problem</td> </tr> <tr> <td>U</td> <td>GeoTAX Auxiliary file problem</td> </tr> <tr> <td>X</td> <td>Combination of Street and state file problem</td> </tr> <tr> <td>Z</td> <td>zip.gsb file problem</td> </tr> </table> <p>The second position is one of the following:</p> <table border="0"> <tr> <td>E</td> <td>Fatal issue, program terminating</td> </tr> <tr> <td>F</td> <td>Expired database</td> </tr> <tr> <td>I</td> <td>Informational</td> </tr> </table> | Blank | Matcher terminated normally | A | User Auxiliary file problem | CE | coubsub.txb file problem | CI | Confidence engine problem | D | Boundary file | F | User-defined boundary file problem | G | Address Matching engine problem | L | Licensing problem | S | State file problem | U | GeoTAX Auxiliary file problem | X | Combination of Street and state file problem | Z | zip.gsb file problem | E | Fatal issue, program terminating | F | Expired database | I | Informational |
| Blank | Matcher terminated normally | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | User Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CE | coubsub.txb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CI | Confidence engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | Boundary file | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | User-defined boundary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| G | Address Matching engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| L | Licensing problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| S | State file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| U | GeoTAX Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| X | Combination of Street and state file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Z | zip.gsb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| E | Fatal issue, program terminating | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | Expired database | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| I | Informational | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------|--|--|
| GTX.ErrorDescription* | 81 | <p>If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <hr/> <p>SI-"TS158 FILES NOT FOUND" SI-"TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" SI-"STATE FILES NOT FOUND" SE-"STATE AND TS158 FILES NOT FOUND" SE-"STATE NOT FOUND AND TS158 VINTAGE ERROR" SI-"STATE FILES VINTAGE OR INCOMPLETE DB ERROR" SE-"STATE VINTAGE ERROR AND TS158 NOT FOUND" SE-"STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR"</p> <hr/> <p>GI-"STREET FILES NOT FOUND" XI-"STREET AND TS158 FILES NOT FOUND" XI-"STREET NOT FOUND AND TS158 FILES VINTAGE ERROR" XI-"STREET AND STATE FILES NOT FOUND" XE-"STREET STATE AND TS158 FILES NOT FOUND" XE-"STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR" XI-"STREET NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR"</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | GI-"STREET FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND TS158 NOT FOUND" XI-"STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND STATE NOT FOUND" XE-"STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND" XE-"STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND" XI-"STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR" XE-"STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND" XE-"STREET STATE AND TS158 VINTAGE ERROR" |
| | | LF-"INVALID FUNCTION PASSED TO GTDBLIO : "AI-"GENIO ERROR:FILE = G1GTAUX , FUNC = ,ST = " UI-"GENIO ERROR: FILE =G1GTAX2 , FUNC = , ST = " XF-"The (DB Vintage) database has expired!" XF-"The (SPD file Vintage) SPD File has expired!" |
| | | DI- "UNABLE TO VALIDATE BOUNDARY LICENSE" DI- "UNABLE TO OPEN BOUNDARY FILE" DI- "BOUNDARY FILE NOT FOUND" FI- "UNABLE TO VALIDATE USER BOUNDARY LICENSE" FI- "UNABLE TO OPEN USER BND FILE" FI- "USER BND FILE NOT FOUND" |
| GTX.WarnCode* | 3 | This field contains warning codes returned by the GeoTAX engine. It is blank if no warnings were issued. A value of <code>WN</code> indicates a database will expire next month. Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform. |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|---|
| GTX.WarnDescription* | 81 | <p>A text description of any warnings returned by the GeoTAX engine.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> |
| Status | 2 | <p>Reports the success or failure of the match attempt.</p> <p>null Success</p> <p>F Failure. Some examples of failures are your license expired or you did not select any output record types and fields for AssignGeoTAXInfo to return.</p> |
| Status.Code | 12 | <p>If AssignGeoTAXInfo could not process the address, this field will show the reason. Currently there is one possible value for this field: Invalid Address.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|--|
| Status.Description | 64 | <p>If AssignGeoTAXInfo could not process the address, this field will show a description of the failure. One of the following:</p> <hr/> <p>TS158 FILES NOT FOUND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR STATE FILES NOT FOUND STATE AND TS158 FILES NOT FOUND STATE NOT FOUND AND TS158 VINTAGE ERROR STATE FILES VINTAGE OR INCOMPLETE DB ERROR STATE VINTAGE ERROR AND TS158 NOT FOUND STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR</p> <hr/> <p>STREET FILES NOT FOUND STREET AND TS158 FILES NOT FOUND STREET NOT FOUND AND TS158 FILES VINTAGE ERROR STREET AND STATE FILES NOT FOUND STREET STATE AND TS158 FILES NOT FOUND STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR STREET NOT FOUND AND STATE VINTAGE ERROR STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR STREET FILES VINTAGE OR INCOMPLETE DB ERROR STREET VINTAGE ERROR AND TS158 NOT FOUND</p> <hr/> <p>STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR STREET VINTAGE ERROR AND STATE NOT FOUND STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND STREET STATE AND TS158 VINTAGE ERROR</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| | | INVALID FUNCTION PASSED TO GTDBLIO : GENIO ERROR: FILE = G1GTAUX , FUNC = , ST = GENIO ERROR: FILE = G1GTAX2 , FUNC = , ST = The (DB Vintage) database has expired! The (SPD file Vintage) SPD File has expired! UNABLE TO VALIDATE BOUNDARY LICENSE UNABLE TO OPEN BOUNDARY FILE BOUNDARY FILE NOT FOUND UNABLE TO VALIDATE USER BOUNDARY LICENSE UNABLE TO OPEN USER BND FILE USER BND FILE NOT FOUND |

CalculateDistance

CalculateDistance takes two sets of latitude/longitude coordinates as input, calculates the distance between the coordinates, and returns the distance between the two points.

CalculateDistance is part of Spectrum Enterprise Tax.

Resource URL

```
http://server:port/soap/CalculateDistance
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:cal="http://www.precisely.com/spectrum/services/CalculateDistance"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <cal:CalculateDistanceRequest>
      <spec:options>
        <cal:LatLongFormat>Decimal</cal:LatLongFormat>
      </spec:options>
      <cal:input_port>
        <cal:Coordinates>
          <cal:FirstLatitude>41.857333</cal:FirstLatitude>
          <cal:FirstLongitude>-88.325183</cal:FirstLongitude>
        </cal:Coordinates>
      </cal:input_port>
    </cal:CalculateDistanceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

```

        <cal:SecondLatitude>41.881833</cal:SecondLatitude>
        <cal:SecondLongitude>-87.785587</cal:SecondLongitude>
    </cal:Coordinates>
</cal:input_port>
</cal:CalculateDistanceRequest>
</soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:CalculateDistanceResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/CalculateDistance">

      <ns3:output_port>
        <ns3:Result>
          <ns3:Distance>27.799</ns3:Distance>
          <ns3:user_fields/>
        </ns3:Result>
      </ns3:output_port>
    </ns3:CalculateDistanceResponse>
  </soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

CalculateDistance takes latitude and longitude information as input.

The table below defines the CalculateDistance input data.

| Parameter | Description | | | | |
|---------------------------|--|----------|-------|----------|-------|
| FirstLatitude | Latitude of the first point for which you want distance returned. | | | | |
| FirstLatitude.Directional | First latitude directional. <table border="0" style="margin-left: 20px;"> <tr> <td>N</td> <td>North</td> </tr> <tr> <td>S</td> <td>South</td> </tr> </table> | N | North | S | South |
| N | North | | | | |
| S | South | | | | |
| FirstLongitude | Longitude of the first point for which you want distance returned. | | | | |

| Parameter | Description |
|-----------------------------|---|
| FirstLongitude.Directional | First longitude directional. E East W West |
| SecondLatitude | Latitude of the second point for which you want distance returned. |
| SecondLatitude.Directional | Second latitude directional. N North S South |
| SecondLongitude | Longitude of the second point for which you want distance returned. |
| SecondLongitude.Directional | Second longitude directional. E East W West |

Parameters for Options

The table below defines the output data and format options.

| Parameter | Description |
|---------------|---|
| LatLongFormat | Indicates the format of the input latitude/longitude. The options are: DegMinSec For example 90 00 00N180 00 00W. PreZero Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W. (default) PreZeroDecimal Decimal degrees using directional indicator. For example, 090.000000N180.000000W. Decimal Decimal degrees using signed latitude/longitude. For example, 90.000000-180.000000. DecimalAssumed Decimal degrees using signed latitude/longitude (no decimal point). For example, 90000000-180000000. |

| Parameter | Description |
|-------------|--|
| ReturnUnits | Indicates the measurement units returned for distance calculation: <ul style="list-style-type: none"> • Feet • Km • Meters • Miles (default) |

Response

CalculateDistance always returns the Confidence field to indicate the confidence in the output provided.

If CalculateDistance fails to process the data, it returns the fields Status, Status.Code, and Status.Descriptions. These fields provide information on why CalculateDistance failed to process the data. Some examples of failures are your license expired or you did not select any output record types and fields for CalculateDistance to return. The following table provides the record-level qualifiers and data outputs for CalculateDistance.

| Response Element | Max. Field Length with null terminator | Description |
|--------------------|--|---|
| Distance | 9 | Distance between the two input coordinates in the units of measurement that you specified. |
| Status | 2 | Reports the success or failure of the match attempt: null Success F Failure |
| Status.Code | 2 | Reason for failure or error. If Status = F, Status.Code = Failure. |
| Status.Description | 64 | Description of the problem. If Status = F, Status.Description = Unable to compute distance. |

ReverseGeoTAXInfoLookup

ReverseGeoTAXInfoLookup allows latitude/longitude coordinates to be supplied as input and identifies the tax districts that apply to the given coordinate. Specifically, ReverseGeoTAXInfoLookup can return the following information about a location:

- FIPS state codes and county codes
- State and county names
- MCD codes and names
- Place codes and names
- Boundary file districts
- Cross-reference tax keys
- The relationship of the input coordinates to user-defined polygons
- Sales and use tax rates, if licensed for the Precisely Sales and Use Tax Rate File

ReverseGeoTAXInfoLookup optionally includes enhanced tax jurisdiction information for a location, including:

- **Insurance premium districts**—Areas designated for the collection of taxes imposed on insurance policy premiums, based on the policy holder's address. Insurance premium districts are created by state governments.
- **Payroll tax districts**—Areas designated for the collection of taxes imposed on employers to support state or local government facilities and services, based on the employee's and/or employer's address. Examples include taxes collected for districts to pay for schools, police, or other services. Payroll tax districts are created by state or local governments.
- **Payroll system tax codes**—Codes that represent specific jurisdictions that collect payroll tax. Using payroll system tax codes has advantages over using the payroll tax district information returned by ReverseGeoTAXInfoLookup:
 - ReverseGeoTAXInfoLookup uses an additional database to determine payroll tax codes, resulting in more accurate payroll tax determination.
 - Many payroll systems use specific codes to determine withholding amounts. Since you can customize the payroll tax codes returned by ReverseGeoTAXInfoLookup, you can set up a process where ReverseGeoTAXInfoLookup returns the exact payroll tax codes required by your payroll system, instead of returning jurisdictional IDs that must then be translated into the codes used by your system.
- **Special purpose tax districts**—Areas designated for the collection of taxes imposed on residents to support specialized services for residents of the district, based on the resident's address. Examples include services such as sewer service, transit service, or water resources. Special purpose tax districts are created by legislative action, court action, or public referendums. This optional information requires the use of boundary files which require an additional license. Contact your Precisely sales representative for more information.

Using the optional Precisely Sales and Use Tax Rate file, ReverseGeoTAXInfoLookup includes tax rate data for a location, including:

Tax rate type:

- General
- Automotive
- Medical
- Construction

Sales and/or use tax rates for:

- State
- County
- Municipality
- Up to 10 SPDs
- Total Rate - the sum of the individual state, county, municipality and SPD rates.

Required input format

The required format for the input coordinates is as follows:

| Response Element | Format |
|------------------|--|
| InputLatitude | 00.000000 or without the decimal point 00000000 |
| InputLongitude | 000.000000 or without the decimal point 000000000, or 00.000000 or without the decimal point 00000000 |

ReverseGeoTAXInfoLookup is part of Spectrum Enterprise Tax.

Resource URL

```
http://server:port/soap/ReverseGeoTAXInfoLookup
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ass="http://www.precisely.com/spectrum/services/ReverseGeoTAXInfoLookup"
xmlns:spec="http://spectrum.precisely.com/">  <soapenv:Header/>
  <soapenv:Body>
    <rev:ReverseGeoTAXInfoLookupRequest>
```

```

    <rev:input_port>
      <rev:Address>
        <rev:InputLatitude>40.018998</rev:InputLatitude>
        <rev:InputLongitude>-105.239580</rev:InputLongitude>
      </rev:Address>
    </rev:input_port>
  </rev:ReverseGeoTAXInfoLookupRequest>
</soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ReverseGeoTAXInfoLookupResponse
xmlns:ns3="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/ReverseGeoTAXInfoLookup">
      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>100.0</ns3:Confidence>
          <ns3:ProcessedBy>GTX</ns3:ProcessedBy>
          <ns3:County.Code>013</ns3:County.Code>
          <ns3:County.Name>Boulder</ns3:County.Name>
          <ns3:StateCode>08</ns3:StateCode>
          <ns3:State.Abbreviation>CO</ns3:State.Abbreviation>

          <ns3:InputLatitude>40.018998</ns3:InputLatitude>
          <ns3:InputLongitude>-105.239580</ns3:InputLongitude>
          <ns3:GeoTAXKey.MatchCode></ns3:GeoTAXKey.MatchCode>

        <ns3:GeoTAXKey.MatchLevel>NoMatch</ns3:GeoTAXKey.MatchLevel><ns3:GeoTAXKey/>

          <ns3:Place.ClassCode>C1</ns3:Place.ClassCode>
          <ns3:Place.Code>07850</ns3:Place.Code>

        <ns3:Place.IncorporatedFlag>Inc</ns3:Place.IncorporatedFlag>
          <ns3:Place.Name>Boulder</ns3:Place.Name>

        <ns3:Place.LastAnnexedDate>10/2011</ns3:Place.LastAnnexedDate>

        <ns3:Place.LastUpdatedDate>04/2013</ns3:Place.LastUpdatedDate>

        <ns3:Place.LastVerifiedDate>01/2013</ns3:Place.LastVerifiedDate>
          <ns3:Place.PointStatus>P</ns3:Place.PointStatus>

        <ns3:Place.DistanceToBorder>000000387</ns3:Place.DistanceToBorder>
          <ns3:GNISCode>002409883</ns3:GNISCode>
          <ns3:GTX.ErrorCode/>
          <ns3:GTX.ErrorDescription/>
          <ns3:GTX.WarnCode/>

```

```

        <ns3:GTX.WarnDescription/>
        <ns3:user_fields/>
    </ns3:Address>
</ns3:output_port>
</ns3:ReverseGeoTAXInfoLookupResponse>
</soap:Body>
</soap:Envelope>

```

Request

Geocoding Options

Reverse geocoding information lookup is the process of taking an input latitude/longitude coordinate and returning jurisdictional tax information. The geocoding options specify the distance units and buffer distance to use when matching to a boundary file.

| Parameter | Description |
|--------------|--|
| Database.GTX | Select the database resource to use in the reverse geocoding lookup process. |

Boundary matching: The following options can be set when matching to a boundary file such as SPD, IPD, PAY, Place and MCD, or user-defined.

| | | | | | |
|---------------|---|-------------|---|---------------|-----------------------------------|
| DistanceUnits | Specifies the units in which to measure distance. One of the following: | | | | |
| | <table border="0"> <tr> <td>Feet</td> <td>Distances are measured in feet. (default)</td> </tr> <tr> <td>Meters</td> <td>Distances are measured in meters.</td> </tr> </table> | Feet | Distances are measured in feet. (default) | Meters | Distances are measured in meters. |
| Feet | Distances are measured in feet. (default) | | | | |
| Meters | Distances are measured in meters. | | | | |

Default buffer widths

| | |
|--------------------|--|
| DefaultBufferWidth | <p>Specifies the buffer width to use for tax district boundary files. The tax district boundary files are the Special Purpose Districts (SPD) file, the Insurance Premium Districts (IPD) file, the Payroll Tax Districts (PAY) file, and Place and MCD files.</p> <p>The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>For more information about buffers, see Buffering on page 1008.</p> |
|--------------------|--|

| Parameter | Description |
|------------------------|--|
| DefaultUserBufferWidth | <p>Specifies the buffer width to use for user-defined boundary files. The distance is in the units of measurement specified in the Distance units option. For information about buffers, see Buffering on page 1008. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field.</p> <p>Note: To use buffers, the user-defined boundary file must support buffers.</p> |

Output Data Options

Data options control the data returned by ReverseGeoTAXInfoLookup.

| Parameter | Description |
|------------------------|--|
| GeoTAXOutputRecordType | <p>Select one or more of the following to obtain the type of data you want returned. ReverseGeoTAXInfo Lookup groups the output fields into record types. If you do not want all of the fields in a record type returned, do not select the check box, and list only those fields you want returned in <code>OutputFields</code>.</p> <ul style="list-style-type: none"> • C—Census • T—Tax Jurisdiction • U—User-defined boundary file • W—Payroll System Tax Codes • X—Auxiliary File • B—PB Software Sales and Use Tax Rate file <p>You can also specify one, and only one, of the following:</p> <p>I Insurance Premium Tax District (IPD)</p> <p>R Payroll Tax District (PAY)</p> <p>S Special Purpose Tax District (SPD)</p> <p>For a description of the fields in each output group, see Response on page 661.</p> <p>Note: If you specify <code>W</code>, also specify <code>R</code> to obtain the best payroll system tax code match possible.</p> |
| TaxKey | <p>If you integrate ReverseGeoTAXInfo Lookup with third-party tax compliance software from Vertex or Sovos, select which vendor you use. This controls the value returned in the <code>GeoTAXKey</code> output field. One of the following:</p> <p>N Do not return either the Sovos or Vertex jurisdiction codes (default).</p> <p>T Return the Sovos jurisdiction code for the address.</p> <p>V Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Precisely.</p> |

| Parameter | Description |
|--------------|---|
| TaxRate | <p>Select the desired Sales tax rate type or none:</p> <p>N Do not return sales tax rates. (default)</p> <p>G Return the General sales tax rates.</p> <p>A Return the Automotive sales tax rates.</p> <p>C Return the Construction sales tax rates. Return the Medical sales tax rates.</p> |
| OutputFields | <p>Indicates the individual output fields you want returned. You can use this field instead of the Output Record Type to limit the output to those fields that are important to your current data needs.</p> <p>For a list of the fields included in each data type, see Response on page 661.</p> |

Output Format

Output format options control how ReverseGeoTAXInfo Lookup formats output data.

| Parameter | Description |
|--------------|--|
| OutputCasing | <p>Specifies the casing of these output fields: <code>County.Name</code>, <code>MCD.Name</code>, <code>Place.Name</code>, <code>IPDn.DistrictName</code>, <code>PAYn.DistrictName</code>, <code>SPDn.DistrictName</code>, and <code>PTCn.PayrollDescription</code>.</p> <p>One of the following:</p> <p>M Returns the output in mixed case (default). For example: Boulder.</p> <p>U Returns the output in upper case. For example: BOULDER.</p> |

Response

Auxiliary File

The table below lists the output fields that contain Auxiliary file data. To include Auxiliary file data in the output, set `GeoTAXOutputRecordType = X`. The following table lists the output fields that contain tax jurisdiction data.

| Response Element | Max. Field Length with null terminator | Description |
|-----------------------------|--|---|
| AuxiliaryData.AuxiliaryFile | 301 | Data retrieved as a result of an auxiliary match from the user-defined area of the auxiliary file. |
| AuxiliaryData.StateFile | 201 | Data retrieved as a result of a state match. Data content and format vary depending on the state file used. |

Census

The census output fields contains census information from the U.S. Census, including Minor Civil Divisions (MCDs) and Census County Division (CCD) names and codes. MCDs are the primary political or administrative divisions of a county, representing many kinds of legal entities with a variety of governmental and administrative functions. CCDs are established in states where there are no legally established MCDs. The Census Bureau recognizes MCDs in 28 states and has established CCDs in 21 states. The District of Columbia has no primary divisions, and the city of Washington, DC is considered equivalent to an MCD for data presentation purposes.

Census data also contains the Federal Information Processing Standards (FIPS) codes for each state and county. The FIPS State Code and the FIPS County Code are both used by the Census Bureau to identify these geographic units.

The following table lists the output fields that contain census data. To include census data in the output, set `GeoTAXOutputRecordType = C`.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| County.Code | 4 | Three-digit Federal Information Processing Standards (FIPS) county code extracted from the Census.BlockCode. Note: This field is always included in the output regardless of whether or not you choose to include census data in the output. |
| County.Name | 26 | Name of the county. Note: This field is always included in the output regardless of whether or not you choose to include census data in the output. |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|---|
| MCD.Code | 6 | Minor Civil Division/Census County Division (MCD/CCD) Code. |
| MCD.Name | 41 | Minor Civil Division/Census County Division (MCD/CCD) name. |
| MCD.PointStatus | 2 | <p>An address can be compared to a Minor Civil Division (MCD)/county subdivision file (Cousub.txt). This output field returns the result for a comparison between the matched geocode location to the polygon defined by the Cousub.txt file.</p> <p>For more information about buffers, see Buffering on page 1008.</p> <p>Note: The buffer distance for Cousub.txt is internally set to zero and cannot be modified.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| MCD.DistanceToBorder | 10 | Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txt file. |
| StateCode | 3 | <p>Two-digit Federal Information Processing Standards (FIPS) state code extracted from the Census.BlockCode.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include census data in the output.</p> |

Input Latitude/Longitude

ReverseGeoTAXInfoLookup always returns the input coordinates as part of the output. The input latitude/longitude fields are returned as input from the data. ReverseGeoTAXInfoLookup does not change these input values.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|------------------|
| InputLatitude | 12 | Input latitude. |
| InputLongitude | 12 | Input longitude. |

Payroll System Tax Code

The following table lists the output fields that contain Payroll System Tax Code (PTC) data. For more information on payroll tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `GeoTAXOutputRecordType = W`.

Note: ReverseGeoTAXInfoLookup returns up to six payroll tax codes per input location.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| NumberPTCsFound | 2 | The number of payroll system tax codes found for this location. |
| PTCn.MatchCode | 2 per PTC | <p>Indicates the level of match obtained for the location. In order from most specific match to least, the possible match codes are:</p> <ul style="list-style-type: none"> P The address was matched to a specific Payroll District ID. This is the most specific match. G The address was matched to a GNIS Code. F The address was matched to a county's FIPS code. S The address was matched to a state's FIPS code. This is the least specific match. |
| PTCn.PayrollCode | 16 per PTC | A code that represents a taxing authority in a payroll application. This is a user-defined code. The specific codes are determined by the payroll application that utilizes the data returned by ReverseGeoTAXInfo Lookup. |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------------|--|--|
| PTCn.PayrollDescription | 41 per PTC | A description of the purpose of this payroll code. |
| PTCn.PayrollFlag | 7 per PTC | A user-defined flag from the PTC database. |
| StateCounty | 33 | The state abbreviation and county name. |

Tax Jurisdiction

Tax jurisdiction data contains information about the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state; or, an area recognized as significant because it is located in an incorporated municipality. Places are used to determine tax jurisdiction.

The following table lists the output fields that contain tax jurisdiction data. To include tax jurisdiction data in the output, set `GeoTAXOutputRecordType = T`.

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| GeoTAXKey | 10 | <p>The value in this field varies depending on the option you specified in the <code>TaxKey</code> option:</p> <p>If you specified <code>T</code>, <code>GeoTAXKey</code> contains the proprietary codes used in Sovos tax compliance software. You can use this code in your Sovos application to find out the tax rate for the jurisdiction. The Sovos jurisdiction code formats are as follows:</p> <ul style="list-style-type: none"> • Sovos SUT - 2-digit SUT state code, 5-digit ZIP Code, 2-digit SUT geocode • Sovos TWE - variable-length TWE geocode <p>If you specified <code>V</code>, <code>GeoTAXKey</code> contains the proprietary Vertex[®] jurisdiction code (comprised of a two-digit Vertex[®] state code, three-digit FIPS county code, and four-digit Vertex[®] city code). You can use this code in your Vertex[®] application to find out the tax rate for the jurisdiction.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GeoTAXKey.MatchCode | 2 | <p>Return code denoting the level of match obtained against the Vertex or Sovos cross reference files.</p> <p>E Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name.</p> <p>P Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code.</p> <p>A Alternate match using two fields: ZIP Code, FIPS place name.</p> <p>N Record is default coded based on valid state code.</p> <p>null No matching record found.</p> |
| GeoTAXKey.MatchLevel | 12 | <p>A description of the value returned in the GeoTAXKey.MatchCode field.</p> <p>Exact Exact match. See description in GeoTAXKey.MatchCode.</p> <p>Partial Partial match. See description in GeoTAXKey.MatchCode.</p> <p>Alternate Alternate match. See description in GeoTAXKey.MatchCode.</p> <p>DefaultCode Record is default coded. See description in GeoTAXKey.MatchCode.</p> <p>NoMatch No matching record found.</p> |
| GNISCode | 10 | <p>Unique nine-digit Geographic Names Information System (GNIS) code.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include census data in the output.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|---|
| Place.ClassCode | 3 | <p>Place class code. Place class codes are used to determine the proper taxing jurisdictions</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.Code | 6 | <p>Five-digit Federal Information Processing Standards (FIPS) place code. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.IncorporatedFlag | 8 | <p>Indicates whether the address is located in an incorporated or unincorporated place. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> <p>Inc Incorporated place code.</p> <p>Uninc Unincorporated place code.</p> <p>Unknown Incorporation status unknown.</p> |
| Place.LastAnnexedDate | 8 | <p>Last annexed date, in the format MM/YYYY, representing the month and year of the most recent boundary change or the most recent available boundary information.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| Place.LastUpdatedDate | 8 | <p>Last updated date, in the format MM/YYYY, reflecting the month and year when TomTom updated the database to reflect attribute (name change, FIPS change, etc.) or boundary edits to the Place.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.LastVerifiedDate | 8 | <p>Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.Name | 41 | <p>The name of the "place" for the location. A "place" is a geographic area defined on the basis of population criteria that vary by state. Or, an area recognized as significant because it is located in an incorporated municipality.</p> <p>Note: This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.</p> |
| Place.PointStatus | 2 | <p>Returns the result for a comparison between the matched geocode location to the polygon defined by the Place.txb file. For more information on buffers, see Buffering on page 1008.</p> <p>P The point is in the polygon.</p> <p>I The point is in the buffer area inside the polygon.</p> <p>B The point is in the buffer area and outside of the polygon.</p> <p>blank Polygon not found.</p> |
| Place.DistanceToBorder | 10 | <p>Returns the distance in feet between the matched address point to the polygon defined by the Place.txb file.</p> |

User-Defined Boundary File

The following table lists the output fields that contain data returned from user-defined boundary files. To include this data in the output, set `GeoTAXOutputRecordType = U`.

Note: ReverseGeoTAXInfoLookup can return up to 10 user-defined areas for each input location.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|--|
| NumberUserBoundariesFound | 3 | The number of user-defined polygons found for the address. |
| UserBoundary <i>n</i> .BoundaryDescription | 51 per User Boundary | A description of the polygon. |
| UserBoundary <i>n</i> .BoundaryID | 11 per User Boundary | The ID of the polygon as specified in the user-defined boundary file. |
| UserBoundary <i>n</i> .BufferRelation | 2 per User Boundary | <p>Indicates where in the polygon the location resides in relation to the edge of the area. Buffer width is specified by the option <code>DefaultUserBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The geocode is inside the polygon at a distance from the edge that is greater than the specified buffer width. I The geocode is inside the polygon but is close to the edge. This indicates that the address is in the buffer area. B The geocode is outside the polygon but is close to the edge. This indicates that the address is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| UserBoundary <i>n</i> .DistanceToBorder | 10 per User Boundary | Indicates the distance in feet from the input location to the border of the polygon. |

| Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| UserBoundary n .SupplementalBoundaryID | 11 per User Boundary | A supplemental ID as specified in the user-defined boundary file. |

Insurance Premium Tax Districts

The following table lists the output fields that contain Insurance Premium Tax Districts (IPD) data. For more information on insurance premium tax districts, see [ReverseGeoTAXInfoLookup](#) on page 656. To include IPD data in the output, set `GeoTAXOutputRecordType = I`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| NumberIPDsFound | 3 | The number of Insurance Premium Tax Districts found for the location. |
| IPD n .BoundaryBuffer.BufferRelation | 2 per IPD | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| IPD n .BoundaryBuffer.DistanceToBorder | 10 per IPD | Indicates the distance in feet from the location to the border of the district. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|---|
| IPDn.DistrictID | 11 per IPD | IPD ID. |
| IPDn.DistrictName | 61 per IPD | IPD name. |
| IPDn.DistrictType | 7 per IPD | IPD district type. |
| IPDn.UpdateDate | 7 per IPD | IPD update date (MMYYYY). |
| IPDn.VersionDate | 7 per IPD | IPD compiled date (MMYYYY). |
| IPDn.Notes | 21 per IPD | Tax code descriptions. For example: 01, 33, A, B |
| IPDn.ChangeDate | 7 per IPD | IPD change date. |
| IPDn.EffectiveDate | 7 per IPD | MMDDYY - Identifies when district becomes active - State supplied For example: 010108 |
| IPDn.ExpirationDate | 7 per IPD | MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009 |
| IPDn.FireRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.FireFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semi colon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|-------------------|--|--|
| IPDn.CasualtyRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.CasualtyFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.VehicleRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.VehicleFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MarineRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MarineFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.HealthRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.HealthFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| IPDn.LifeRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.LifeFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.OtherRate | 21 per IPD | Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7 |
| IPDn.OtherFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |
| IPDn.MinimumRate | 21 per IPD | Format is dependent on associated flag For example: .13, 15.00 or 3;7 |
| IPDn.MinimumFlag | 6 per IPD | P - Percentage; .1 = 10%, .0575 = 5.75% F - Flat Fee dollar amount M - Multiple Percentages has a semicolon as a delimiter. 3;7 = "3% or 7%" |

Payroll Tax Districts

The following table lists the output fields that contain Payroll Tax District (PAY) data. For more information on payroll tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `GeoTAXOutputRecordType = R`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberPAYsFound | 3 | Number of payroll tax districts found for the location. |
| PAY <i>n</i> .BoundaryBuffer.BufferRelation | 2 per PAY | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| PAY <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per PAY | Indicates the distance in feet from the location to the border of the district. |
| PAY <i>n</i> .DistrictID | 11 per PAY | PAY district ID. |
| PAY <i>n</i> .DistrictName | 61 per PAY | PAY district name. |
| PAY <i>n</i> .DistrictType | 7 per PAY | PAY district type. |
| PAY <i>n</i> .ID | 11 per PAY | PAY ID. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------------------------|--|---|
| PAY <i>n</i> .MunicipalEMSTax | 2 per PAY | <p>PAY municipality emergency municipal services tax.</p> <p>The values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .MunicipalIncomeTax | 2 per PAY | <p>PAY municipality income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X None</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictEMSTax | 2 per PAY | <p>PAY school district emergency municipal services tax.</p> <p>The Values for Pennsylvania are:</p> <p>Y Levies the tax</p> <p>N Does not levy the tax</p> <p>All other states are null.</p> |
| PAY <i>n</i> .SchoolDistrictIncomeTax | 2 per PAY | <p>PAY school district income tax.</p> <p>The values for Pennsylvania are:</p> <p>R Resident</p> <p>N Non-resident</p> <p>B Both</p> <p>X N</p> <p>The values for Ohio are:</p> <p>R Resident</p> <p>X None</p> <p>All other states are null.</p> |

Special Purpose Tax Districts

The following table lists the output fields that contain Special Purpose Tax Districts (SPD) data. For more information on special purpose tax districts, see [Reverse GeoTAX Info Lookup](#). To include this data in the output, set `GeoTAXOutputRecordType = S`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, and PAYs.

| Response Element | Max. Field Length with null terminator | Description |
|---|--|---|
| NumberSPDsFound | 3 | Number of Special Purpose Tax Districts found for the location. |
| SPD <i>n</i> .BoundaryBuffer.BufferRelation | 2 per SPD | <p>Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option <code>DefaultBufferWidth</code> or by the input field <code>BufferWidth</code>.</p> <p>One of the following:</p> <ul style="list-style-type: none"> P The location is inside the district at a distance from the edge that is greater than the specified buffer width. I The location is inside the district but is close to the edge. This indicates that the location is in the buffer area. B The location is outside the district but is close to the edge. This indicates that the location is in the buffer area. <p>For more information, see Buffering on page 1008.</p> |
| SPD <i>n</i> .BoundaryBuffer.DistanceToBorder | 10 per SPD | Indicates the distance in feet from the address to the border of the district. |
| SPD <i>n</i> .CompiledDate | 7 per SPD | SPD compiled date. |
| SPD <i>n</i> .DistrictCode | 4 per SPD | 3-digit district type code. |
| SPD <i>n</i> .DistrictName | 61 per SPD | SPD name. |

| Response Element | Max. Field Length with null terminator | Description |
|------------------------------|--|----------------------|
| SPD <i>n</i> .DistrictNumber | 6 per SPD | SPD district number. |
| SPD <i>n</i> .EffectiveDate | 7 per SPD | SPD effective date. |
| SPD <i>n</i> .UpdateDate | 7 per SPD | SPD update date. |
| SPD <i>n</i> .VersionDate | 7 per SPD | SPD version date. |

Sales and Use Tax Rates

The table below lists the output fields that contain the sales and use tax rate data.

To include tax rate data in the output, set `GeoTAXOutputRecordType = B`.

To select the tax rate type, set `TaxRate` to one of the following:

- N** Do not return sales and use tax rates. (default)
- G** Return the General sales and use tax rates.
- A** Return the Automotive sales and use tax rates.
- C** Return the Construction sales and use tax rates.
- M** Return the Medical sales and use tax rates.

Note: You must be a licensed user of the Precisely Sales and Use Tax Rate file to use this feature.

The following table describes the Sales and Use Tax Rate output fields.

| Response Element | Max. Field Length with null terminator | Description |
|------------------------|--|--|
| TaxRate.RC | 2 | Tax Rate return code denoting the level of match obtained against the Precisely Sales and Use Tax Rate file: E Exact match, using all 5 fields P Partial match, using 4 fields A Alternate match, using 3 fields N Record is default-coded based on valid state code. Blank No matching PB Software Sales and Use Tax Rate record found. |
| Municipal.SalesTaxRate | 11 | Municipality sales tax rate for the selected tax rate type. |
| County.SalesTaxRate | 11 | County sales tax rate for the selected tax rate type. |
| State.SalesTaxRate | 11 | State sales tax rate for the selected tax rate type. |
| SPD n .SalesTaxRate | 11 per SPD | Sales tax rate for up to 10 Special Purpose Districts (SPD). |
| TaxRate.SalesTotal | 11 | The sum of the individual Municipal, County, State and SPD sales tax rates. |
| Municipal.UseTaxRate | 11 | Municipality use tax rate for the selected tax rate type. |
| County.UseTaxRate | 11 | County use tax rate for the selected tax rate type. |
| State.UseTaxRate | 11 | State use tax rate for the selected tax rate type. |
| SPD n .UseTaxRate | 11 per SPD | Use tax rate for up to 10 Special Purpose Districts (SPD). |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|---|
| TaxRate.UseTotal | 11 | The sum of the individual Municipal, County, State and SPD use tax rates. |

Error Reporting

The table below defines the error reporting output fields.

| Response Element | Max. Field Length with null terminator | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|--|---|--------------|-----------------------------|----------|-----------------------------|-----------|--------------------------|-----------|---------------------------|----------|---------------|----------|------------------------------------|----------|---------------------------------|----------|-------------------|----------|--------------------|----------|-------------------------------|----------|--|----------|----------------------|----------|----------------------------------|----------|------------------|----------|---------------|
| GTX.ErrorCode | 3 | <p>This field contains a return code if the GeoTAX engine experiences an abnormal termination.</p> <p>Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <p>The first character indicates the file (or set of files affected).</p> <table border="0"> <tr> <td>Blank</td> <td>Matcher terminated normally</td> </tr> <tr> <td>A</td> <td>User Auxiliary file problem</td> </tr> <tr> <td>CE</td> <td>coubsub.txb file problem</td> </tr> <tr> <td>CI</td> <td>Confidence engine problem</td> </tr> <tr> <td>D</td> <td>Boundary file</td> </tr> <tr> <td>F</td> <td>User-defined boundary file problem</td> </tr> <tr> <td>G</td> <td>Address Matching engine problem</td> </tr> <tr> <td>L</td> <td>Licensing problem</td> </tr> <tr> <td>S</td> <td>State file problem</td> </tr> <tr> <td>U</td> <td>GeoTAX Auxiliary file problem</td> </tr> <tr> <td>X</td> <td>Combination of Street and state file problem</td> </tr> <tr> <td>Z</td> <td>zip.gsb file problem</td> </tr> </table> <p>The second position is one of the following:</p> <table border="0"> <tr> <td>E</td> <td>Fatal issue, program terminating</td> </tr> <tr> <td>F</td> <td>Expired database</td> </tr> <tr> <td>I</td> <td>Informational</td> </tr> </table> | Blank | Matcher terminated normally | A | User Auxiliary file problem | CE | coubsub.txb file problem | CI | Confidence engine problem | D | Boundary file | F | User-defined boundary file problem | G | Address Matching engine problem | L | Licensing problem | S | State file problem | U | GeoTAX Auxiliary file problem | X | Combination of Street and state file problem | Z | zip.gsb file problem | E | Fatal issue, program terminating | F | Expired database | I | Informational |
| Blank | Matcher terminated normally | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | User Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CE | coubsub.txb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CI | Confidence engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| D | Boundary file | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | User-defined boundary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| G | Address Matching engine problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| L | Licensing problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| S | State file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| U | GeoTAX Auxiliary file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| X | Combination of Street and state file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Z | zip.gsb file problem | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| E | Fatal issue, program terminating | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| F | Expired database | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| I | Informational | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Response Element | Max. Field Length with null terminator | Description |
|----------------------|--|--|
| GTX.ErrorDescription | 81 | <p>If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally. The maximum length is 80.</p> <p>Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform.</p> <hr/> <p>SI-"TS158 FILES NOT FOUND" SI-"TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" SI-"STATE FILES NOT FOUND" SE-"STATE AND TS158 FILES NOT FOUND" SE-"STATE NOT FOUND AND TS158 VINTAGE ERROR" SI-"STATE FILES VINTAGE OR INCOMPLETE DB ERROR" SE-"STATE VINTAGE ERROR AND TS158 NOT FOUND" SE-"STATE AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR"</p> <hr/> <p>GI-"STREET FILES NOT FOUND" XI-"STREET AND TS158 FILES NOT FOUND" XI-"STREET NOT FOUND AND TS158 FILES VINTAGE ERROR" XI-"STREET AND STATE FILES NOT FOUND" XE-"STREET STATE AND TS158 FILES NOT FOUND" XE-"STREET AND STATE NOT FOUND AND TS158 VINTAGE ERROR" XI-"STREET NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET AND TS158 NOT FOUND AND STATE VINTAGE ERROR" XE-"STREET NOT FOUND AND STATE AND TS158 VINTAGE ERROR"</p> <hr/> |

| Response Element | Max. Field Length with null terminator | Description |
|------------------|--|--|
| | | GI-"STREET FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND TS158 NOT FOUND" XI-"STREET AND TS158 FILES VINTAGE OR INCOMPLETE DB ERROR" XI-"STREET VINTAGE ERROR AND STATE NOT FOUND" XE-"STREET VINTAGE ERROR AND STATE AND TS158 NOT FOUND" XE-"STREET AND TS158 VINTAGE ERROR AND STATE NOT FOUND" XI-"STREET AND STATE FILES VINTAGE OR INCOMPLETE DB ERROR" XE-"STREET AND STATE VINTAGE ERROR AND TS158 NOT FOUND" XE-"STREET STATE AND TS158 VINTAGE ERROR" |
| | | LF-"INVALID FUNCTION PASSED TO GTDBLIO : " AI-"GENIO ERROR: FILE = G1GTAUX , FUNC = , ST = " UI-"GENIO ERROR: FILE = G1GTAX2 , FUNC = , ST = " XF-"The (DB Vintage) database has expired!" XF-"The (SPD file Vintage) SPD File has expired!" |
| | | DI- "UNABLE TO VALIDATE BOUNDARY LICENSE" DI- "UNABLE TO OPEN BOUNDARY FILE" DI- "BOUNDARY FILE NOT FOUND" FI- "UNABLE TO VALIDATE USER BOUNDARY LICENSE" FI- "UNABLE TO OPEN USER BND FILE" FI- "USER BND FILE NOT FOUND" |
| GTX.WarnCode | 3 | This field contains warning codes returned by the GeoTAX engine. It is blank if no warnings were issued. A value of <code>WN</code> indicates a database will expire next month. Note: This field contains the same set of codes returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform. |

| Response Element | Max. Field Length with null terminator | Description |
|---------------------|--|--|
| GTX.WarnDescription | 81 | A text description of any warnings returned by the GeoTAX engine. Note: This field contains the same set of descriptions returned by the standalone GeoTAX software and is intended for users who have migrated from GeoTAX to Spectrum Technology Platform. |

Match and Location Codes

Match Codes

Match Codes indicate the portions of the address that matched or did not match to the reference file. If a match could not be made, the Match Code begins with "E" and the remaining digits indicate why the address did not match (see [Match Codes for No Match - Definitions for "Ennn" return codes](#) on page 689). The digits do not specifically refer to which address elements did not match, but rather why the address did not match. These fields are always included in the output from AssignGeoTAXInfo.

Match Code Definitions

| Response Element | Description |
|------------------|---|
| Ahhh | Same as Shhh, but indicates match to an alias name record or an alternate record. |
| Chh | Street address did not match, but 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 | House number was changed. |
| 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. |

| Response Element | Description |
|------------------|---|
| 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. |
| 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. |
| Z | No address given, but verified the provided ZIP Code. |

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: 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 686.

| 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. |

| Code | In first hex position means: | In second and third hex position means: |
|------|------------------------------------|---|
| 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. |
| 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. |

| Code | In first hex position means: | In second and third hex position means: |
|------|--|--|
| 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 additional information is returned 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, Q, R, S, T or U (see [Match Codes](#) on page 683).

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 684
- Extended Match Codes, see the table below.

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

- The output address has extra information (for example, a mailstop) in the address line.
- The output address has a second address line (`AddressLine2`).
- 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. |
| 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 moved to <code>AddressLine2</code> . |

| Input Addressline | Output Addressline | Extended Code | Description |
|--|------------------------|---------------|--|
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |
| 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 moved to AddressLine2. |

| 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 moved to <code>AddressLine2</code> . |
| 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 moved to <code>AddressLine2</code> . |

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

Note: For Landmark Auxiliary file matches, the 3rd hex digit is always "0".

| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 2 | Matched on Unit Number. Unit Type changed. |
| 3 | Matched on Unit Number. Unit Type changed. Extra information on address line ignored. Extra information on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 4 | Unit Number changed or ignored. |

| Code | In 3rd hex position means: |
|------|---|
| 5 | Unit Number changed or ignored. Extra information on address line ignored. Extra information on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 8 | Matched on Unit Number and Unit Type if included. House Number changed or ignored. |
| 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 moved to <code>AddressLine2</code> . |
| 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 moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |
| 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 on address line ignored. Extra information not considered for matching moved to <code>AddressLine2</code> . |

Match Codes for No Match - Definitions for "Ennn" return codes

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

| Code | "nnn" values | Description |
|------|--------------|--|
| Ennn | | 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 = 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 Group 1 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" values | 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. |

Location Codes

The Location Codes indicate the methodology used to compute the geocode and may also provide some information about the quality of the geocode.

A Location Code of ""E" indicates a location code is not available. This usually occurs when you have requested ZIP Code centroids of a high quality, and one is not available for that match. It can occur infrequently when Spectrum Enterprise Tax does not have a 5-digit centroid location. An "E" location code type may also be returned when the input address cannot be standardized and there is no input ZIP Code. In this case, do not assume the ZIP Code returned with the nonstandardized address is the correct ZIP Code because Spectrum Enterprise Tax did not standardize the address; therefore, Spectrum Enterprise Tax does not return geocoding or Census Block information.

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.

| | |
|---|---|
| 1 st character | Always an "A" indicating an address location. |
| 2 nd 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 |
| 3 rd and 4 th character | Digit indicating other qualities about the location. |

Location Codes

| Code | Description |
|---------|---|
| AGn | Indicates a geocode match to a GeoTAX Auxiliary or Landmark Auxiliary file where n is one of the following values: |
| n = 0 | The geocode represents the center of a parcel, building or landmark. |
| 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 = 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 | <p data-bbox="873 342 1029 363">Structure point</p> <p data-bbox="873 384 1427 436">Indicates a location within a building footprint polygon that is associated with the matched address.</p> <p data-bbox="873 457 1427 573">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.</p> <p data-bbox="873 594 1427 762">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.</p> <p data-bbox="873 783 1427 989">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.</p> |
| nn = 07 | <p data-bbox="873 1073 1045 1094">Manually placed</p> <p data-bbox="873 1115 1427 1199">Address points are manually placed to coincide with the midpoint of a parcel's street frontage at a distance from the center line.</p> |
| nn = 08 | <p data-bbox="873 1283 1045 1304">Front door point</p> <p data-bbox="873 1325 1427 1503">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.</p> |
| nn = 09 | <p data-bbox="873 1587 1094 1608">Driveway offset point</p> <p data-bbox="873 1629 1427 1745">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.</p> |

| Code | Description |
|---------|--|
| nn = 10 | <p>Street access point</p> <p>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.</p> |
| nn = 21 | <p>Base parcel point</p> <p>The Centrus point data includes individual parcels that may be "stacked". These stacked parcels are individually identified by their unit or suite number, and Spectrum Enterprise Tax is able to match to this unit number and return the correct tax jurisdictions. If an input address is for a building or complex without a unit number, the "base" parcel information returns and will not standardize to a unit number or return additional information such as tax jurisdictions.</p> |
| nn = 22 | <p>Backfill address point</p> <p>The precise parcel centroid is unknown. The address location assigned is based on two known parcel centroids.</p> |
| nn = 23 | <p>Virtual address point</p> <p>The precise parcel centroid is unknown. The address location assigned is relative to a known parcel centroid and a street segment end point.</p> |
| nn = 24 | <p>Interpolated address point</p> <p>The precise parcel centroid is unknown. The address location assigned is based on street segment end points.</p> |
| AIn | <p>The correct segment is inferred from the candidate records at match time.</p> |
| ASn | <p>House range address geocode. This is the most accurate street interpolated geocode available.</p> |

| Code | Description |
|--|--|
| AIn, ASn and ACn share the same values for the 3 rd character "n" as follows: | |
| n = 0 | Best location. |
| 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 | <p>Indicates one or both of the following:</p> <ul style="list-style-type: none"> • 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. <p>Note: Only the second case is valid for non-TIGER data because segment range interpolation is only completed for TIGER data.</p> |
| 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. |
| 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. |

| Code | Description |
|-------|--|
| 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. |
| 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

Street centroid location codes indicate the Census ID accuracy and the position of the geocode on the returned street segment. A street centroid location code has the following characters.

| | |
|---------------------------|---|
| 1 st character | Always "C" indicating a location derived from a street segment. |
| 2 nd 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 table below contains the values and descriptions for the 2nd - 3rd characters in the street centroid location codes.

| Character position | Code | Description |
|---------------------------|------|--|
| 2 nd 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. |
| 3 rd Character | | |
| | C | Segment Centroid. |
| | L | Segment low-range end point. |
| | H | Segment high-range end point. |

ZIP + 4 Location Codes

Location codes that begin with a "Z" are ZIP + 4 centroid location codes. 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.

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

The table below contains the values and descriptions for the 2nd-4th characters in the ZIP + 4 location codes.

| Character Position | Code | Description |
|---------------------------|------|---|
| 2 nd Character | | |
| | B | Block Group accuracy (most accurate). |
| | T | Census Tract accuracy. |
| | C | Unclassified Census accuracy. Normally accurate to at least the County level. |
| 3 rd Character | | |

| Character Position | Code | Description |
|---------------------------|------|--|
| | 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 4 th character for a precise indication of locational accuracy. |
| | 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 4 th character for a precise indication of locational accuracy. |
| 4 th 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | 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 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. |
| | 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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. |
| | M | Created ZIP+2 cluster centroid as defined by methodology 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 methodology 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. |

| Character Position | Code | Description |
|--------------------|------|--|
| | 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 geographic centroid fallback option is enabled and an address-level geocode could not be determined. Geographic centroid location codes indicate the quality of a city, county, or state centroid.

| | |
|---------------------------|--|
| 1 st character | Always "G" indicating a location derived from a geographic centroid. |
| 2 nd character | Geographic area type. One of the following: <ul style="list-style-type: none"> M Municipality (for example, a city) C County S State |

Type Codes

The returned type code is referenced from an installed tax district file and indicates the type of tax district or tax jurisdiction for the address location.

This appendix provides the definitions for the following tax district files' type codes:

- **Special Purpose Districts (SPD)**
- **Insurance Premium Districts (IPD)**
- **Payroll Tax Districts (PAY)**

Special Purpose Districts (SPD)

| Type | Descriptions |
|------|-----------------------------------|
| AMB | AMBULANCE DISTRICT |
| ASC | SALES AND USE TAX |
| ATA | ADVANCED TRANSPORTATION AUTHORITY |
| ATD | AIRPORT TAX DISTRICT |
| BSD | BASEBALL STADIUM DISTRICT |
| CAD | COUNTY ASSISTANCE DISTRICT |
| CCD | CRIME CONTROL DISTRICT |
| CFA | COUNTY FINANCE AUTHORITY |
| CMB | COMBINED DISTRICT |
| CTY | CITY TRANSACTIONS |
| DVD | DEVELOPMENT DISTRICT |
| EDD | ECONOMIC DEVELOPMENT DISTRICT |
| EDZ | ECONOMIC DEVELOPMENT ZONE |
| ESD | EMERGENCY SERVICES DISTRICT |
| FCD | FIRE CONTROL DISTRICT |
| FPA | FLOOD PROTECTION AUTHORITY |
| FPD | FIRE PROTECTION DISTRICT |
| FSD | FOOTBALL STADIUM DISTRICT |

| Type | Descriptions |
|------|-----------------------------------|
| HBZ | HOSPITAL BENEFIT ZONE |
| HSA | HOUSING AUTHORITY |
| HSD | HEALTHCARE SERVICES DISTRICT |
| HSP | HOSPITAL DISTRICT |
| IMP | IMPROVEMENT DISTRICT |
| IRD | INDIAN RESERVATION |
| LFW | LFW/CDC |
| LIB | LIBRARY DISTRICT |
| MSD | MUSEUM DISTRICT |
| MTA | METRO TRANSPORTATION AUTHORITY |
| OSA | OPEN SPACE AUTHORITY |
| PFD | PUBLIC FACILITY DISTRICT |
| POL | POLICE DISTRICT |
| PRD | PARK AND RECREATION DISTRICT |
| PSI | PUBLIC SAFETY IMPROVEMENT |
| RCT | RACE TRACK |
| RDA | REVENUE DEVELOPMENT AREA |
| RMA | ROAD MAINTENANCE AUTHORITY |
| RTA | REGIONAL TRANSPORTATION AUTHORITY |
| RTD | RESTAURANT TAX DISTRICT |

| Type | Descriptions |
|------|--|
| SAD | SPORTS DISTRICT |
| SCD | SCIENCE AND CULTURAL DISTRICT |
| SUT | SALES AND USE TAX |
| TDD | TRANSPORTATION DEVELOPMENT DISTRICT |
| TED | TOURISM COMMUNITY ENHANCEMENT DISTRICT |
| UNI | SCHOOL DISTRICT |
| URA | URBAN RENEWAL AUTHORITY |
| WCD | WATER COMMISSION DISTRICT |
| ZOO | ZOO DISTRICT |

Insurance Premium Districts (IPD)

| State | Type | Descriptions |
|-------|--------|-------------------------|
| AL | FIRE | Fire District |
| AL | NT-MUN | Non-Taxing Municipality |
| AL | PREM | Premium Tax District |
| AZ | PRIV | Private Fire District |
| AZ | PUB | Public Fire District |
| DE | FIRE | Fire District |
| FL | FIRE | Fire District |
| FL | POLICE | Police District |

| State | Type | Descriptions |
|-------|--------|--------------------------------------|
| GA | PREM | Premium Tax District |
| IL | FIRE | Fire District |
| KY | COUNTY | County |
| KY | MUNI | Municipality |
| KY | USD | Urban Services District |
| LA | PREM | Premium Tax District |
| MN | FIRE | Fire District |
| ND | FIRE | Fire District |
| NJ | FIRE | Fire District |
| NY | FIRE | Fire District |
| SC | FIRE | Fire District |
| SC | NT-MUN | Non-Taxing Municipality |
| SC | PREM | Premium Tax District |
| TX | PROP | Windstorm Surcharge on Property Line |

Payroll Tax Districts (PAY)

| Type | Descriptions |
|------|-------------------------------------|
| JED | Joint Economic Development District |
| MTA | Mass Transit Authority |
| MUN | Municipality |

| Type | Descriptions |
|------|-----------------|
| UNI | School District |

Class Codes

This appendix lists definitions for the FIPS Class Codes.

Class C—Incorporated Places

| Class Code | Description |
|------------|--|
| C1 | <p>Identifies an active incorporated place that is not also recognized as an Alaska Native Village Statistical area, and does not also serve as a primary county division; that is, it is included in and is part of a primary county division.</p> <p>For example, the city of Hammond, Indiana is within and part of North township; the city of Austin, Texas is within and part of several census county divisions in several counties; Hammond and Austin are coded C1.</p> |
| C2 | <p>Identifies an incorporated place that also serves as a primary county division because, although the place is coextensive with a minor civil division (MCD), the Census Bureau, in agreement with State officials, does not recognize the MCD for presenting census data because the MCD is a nonfunctioning entity; applies to Iowa and Ohio only.</p> <p>For example, the city of Dubuque, Iowa is coextensive with Julien township, which does not function as a governmental unit and may not be well-known even to local residents; the city is assigned code C2, and the township, Z8. This subclass is new for FIPS 55-3. Also see subclass C5.</p> |
| C3 | <p>Identifies a consolidated city; that is, an incorporated place that has consolidated its governmental functions with a county or MCD, but continues to include other incorporated places that are legally part of the consolidated government.</p> <p>For example, the city of Columbus, Georgia is consolidated with Muscogee County, which continues to exist as a nonfunctioning legal entity in the State; however, the town of Bibb City continues to exist as a separate active incorporated place within the consolidated government and, therefore, Columbus is treated as a consolidated city. At the time of publication, there are seven consolidated cities in the United States: Athens-Clarke County, Georgia; Butte-Silver Bow, Montana; Columbus, Georgia; Indianapolis, Indiana; Jacksonville, Florida; Milford, Connecticut; and Nashville-Davidson, Tennessee. This subclass is new for FIPS 55-3.</p> |

| Class Code | Description |
|------------|---|
| C4 | <p>Identifies an alternate authoritative common name of any member of the other subclasses of Class C. The entity code of the legal name is referenced in the "Other Name Code" of the record, and in the entry for the legal name, the Other Name Code references the alternate.</p> <p>For example, the entity in California whose legal name is San Buenaventura (subclass C1) is commonly known as Ventura, which is coded C4.</p> |
| C5 | <p>Identifies an incorporated place that also serves as a primary county division; that is, it is not included in any adjacent primary county division of class T or Z. For example, Boston, MA, is legally a primary division of the county and recognized as an incorporated place and, therefore, is coded C5. Also see subclass C2.</p> |
| C6 | <p>Identifies an incorporated place that is coincident with or approximates an Alaska Native Village statistical area. The Other Name Code references the Alaska Native Village statistical area; see code E6.</p> |
| C7 | <p>Identifies an independent city. At the time of publication, independent cities exist in only four States: Maryland (Baltimore City), Nevada (Carson City), Missouri (St. Louis City), and Virginia (41 cities). These cities also serve as county equivalents, and all but Carson City also serve as primary county divisions.</p> |
| C8 | <p>Identifies the portion of a consolidated city that is not within another incorporated place; see subclass C3. The Census Bureau identifies these nonfunctioning entities by taking the name of the consolidated city and appending in parentheses the word remainder. For example, Columbus (remainder) identifies the portion of the Columbus, Georgia consolidated city that is not also in Bibb City. This code is new for FIPS 55-3.</p> |
| C9 | <p>Identifies an inactive or nonfunctioning incorporated place.</p> |

Class U—Unincorporated Places (Except Those Associated with Facilities)

| Type | Descriptions |
|------|--|
| U1 | <p>Identifies a census designated place (CDP) with a name identical to the authoritative common name that describes essentially the same population. Also see code M2.</p> |
| U2 | <p>Identifies a CDP with a name not identical to an authoritative common name of essentially the same area. If there is an alternate authoritative common name, it is referenced in the Other Name Code field. For example, Suitland-Silver Hill, Maryland is the name of a locally delineated CDP recognized by the Census Bureau which is a combination of two communities Suitland and Silver Hill and, therefore, because it is not the authoritative name of the area, is coded U2; Sierra Vista Southeast, Arizona is a CDP that includes the built-up area adjoining the city of Sierra Vista on the southeast, but is not an authoritative name for that area and, therefore, is coded U2. Also see code M2.</p> |

| Type | Descriptions |
|------|---|
| U3 | Identifies (a) an alternate, authoritative common name of a population essentially described by a specific CDP with a different name (the Other Name Code references the CDP), or (b) a community wholly or substantially within the boundaries of a CDP with a different name (the Part of Code references the CDP). For example, Silver Hill and Suitland are coded U3 and cross-referenced to the CDP of Suitland-Silver Hill (see code U2). |
| U4 | Identifies a populated place wholly or substantially within the boundaries of an incorporated place with a different name; the Part of Code identifies the incorporated place. For example, Harlem and Greenwich Village, which are part of New York city, and Hollywood, which is part of Los Angeles, California, are coded U4. |
| U5 | Dropped. Only one place the CDP of Arlington, Virginia was in this subclass in FIPS PUB 95-2; it has been recoded as U1 as a place and as Z3 as a subclass in FIPS 55-3 as a county subdivision. |
| U6 | Identifies a populated place located wholly or substantially outside the boundaries of any incorporated place or CDP with an authoritative common name recognized by the U.S. Geological Survey. |
| U8 | Identifies a populated place located wholly or substantially outside the boundaries of an incorporated place or CDP but whose name has not been verified as authoritative by the U.S. Geological Survey. |
| U9 | Identifies a CDP that is coincident with or approximates the area of an Alaska Native Village statistical area. The Other Name Code references the Alaska Native Village statistical area; see code E2. This code is new for FIPS 55-3. |

Spectrum GeoConfidence

GeoConfidenceSurface

GeoConfidenceSurface returns geoconfidence polygons (also called surfaces) based on the quality of the geocode information generated by Spectrum Enterprise Geocoding. With the geoconfidence polygons generated, you can then overlap this polygon with other spatial data to determine a risk or probability.

This service is used by the Spectrum GeoConfidence's FloodZoneAnalysis dataflow template.

Note: GeoConfidence uses services provided by the Spectrum Enterprise Geocoding and Spatial modules.

Resource URL

```
http://server:port/soap/GeoConfidenceSurface
```

Request

The input fields for GeoConfidenceSurface are the output fields returned by the GeoConfidence output category of the Enterprise Geocoding Module. These fields are described below.

| columnName Field Name Response Element | Max. Field Length with null terminator | Description |
|--|--|---|
| GeoConfidenceCode | 13 | <p>The value returned in this field indicates which geoconfidence surface type has been returned.</p> <p>The possible values are:</p> <p>INTERSECTION A geocode point for the intersection of two streets.</p> <p>ADDRESS An array of street segment points representing the street segment where the address is located.</p> <p>POINT If the geocoder was able to match the address using point data, the point geometry where the address is located.</p> <p>POSTAL1 A geocode point for the ZIP centroid.</p> <p>POSTAL2 An array of points for all street segments in the ZIP + 2 in which the address is located.</p> <p>POSTAL3 An array of points for street segments in the ZIP + 4 in which the address is located.</p> <p>ERROR An error has occurred.</p> |
| StreetSegmentPoints | 1024 | <p>An array of latitude/longitude values that represent the street segment points.</p> <p>Note: This field contains values only if the GeoConfidenceCode field returns a value of ADDRESS, POSTAL2, or POSTAL3.</p> |
| GeoConfidenceCentroidLatitude | 11 | The latitude of the centroid of the geoconfidence polygon. |
| GeoConfidenceCentroidLongitude | 12 | The longitude of the centroid of the geoconfidence polygon. |

Response

The GeoConfidenceSurface output field contains the geoconfidence polygon.

| Response Element | Description |
|------------------|--|
| Geometry | A geoconfidence polygon that represents the returned geometry. |

Spectrum Global Sentry

GlobalSentry

The GlobalSentry service matches transactions against government-provided watch lists that contain data from various countries. These lists include:

- Denied Persons List (United States)
- Unverified List (BIS Red Flag) (United States)
- Consolidated Financial Sanction Targets (Individuals and Entities) (United Kingdom or European Union)
- Consolidated lists of persons, groups, and entities subject to EU financial sanctions (European Union)
- DFAT Consolidated List (Australia)
- OSFI Consolidated List (Individuals and Entities) (Canada)
- Specially Designated Nationals, Terrorists, Narcotic Traffickers and other Blocked Persons List (United States)
- Statutorily Debarred Parties List (United States)
- Politically Exposed Persons (PEP) list
- The consolidated Sanctions List including all individuals and entities who have been subjected to sanctions by the United Nations Security Council.

Matches are performed against Sanctioned Countries, Name, Address, ID Number and other information such as DOB to provide an "Overall Risk Level Score" that allows your organization to make the right choice before making a decision to block a particular transaction and avoid false positive results.

These steps describe how GlobalSentry processes data:

1. The service first scans all required data in the transaction to identify countries that have been sanctioned. If a sanction country match has been identified, the transaction bypasses all other matching criteria and is assigned the highest possible risk score.
2. If a sanctioned country match has not been identified, the service then attempts to match the transaction against the GlobalSentry database using the GlobalSentry Name Check, GlobalSentry Address Check or GlobalSentry ID Number Check subflows.
3. The GlobalSentry Name Check attempts to match individuals, entities and vessels. If a name match is identified a Name Score is returned from the service.
4. The GlobalSentry Address Check attempts to match addresses within a country. If an Address match is identified an Address Score is returned from the service.
5. The GlobalSentry ID Number Check attempts to match identification numbers, such as Passport, National ID, SSN, and Fiscal Code. If an ID Number match is identified an ID Number Score is returned from the service.
6. If a transaction is not identified as a Name, Address or ID Number match, the transaction record is written to the output and given an overall risk level score of zero.
7. If a transaction has been identified as a Name, Address or Identification Number match, the service attempts to match those transactions against the GlobalSentry database using the GlobalSentry Other Data Check subflow.
8. The GlobalSentry Other Data Check attempts to match the Place of Birth, Date of Birth, Nationality or Citizenship. If a match is identified a Place of Birth Score, Date of Birth Score, Nationality Score or Citizenship Score is returned by the service.
9. GlobalSentry assigns an Overall Risk Level score to each transaction. The score is a value between 0 and 16 and is returned in the OverallRiskLevel field. In calculating the risk level, GlobalSentry takes into account what data was provided in the input record and which inputs, if any, matched entries in the GlobalSentry database. Generally, a higher value indicates a higher risk associated with the transaction.

Resource URL

```
http://server:port/soap/GlobalSentry
```

Example

A SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:glob="http://www.precisely.com/spectrum/services/GlobalSentry">
  <soapenv:Header/>
  <soapenv:Body>
    <glob:GlobalSentryRequest>
      <glob:options/>
      <glob:Input>
        <glob:Row>
          <glob:FirstName>Miguel</glob:FirstName>
        </glob:Row>
      </glob:Input>
    </glob:GlobalSentryRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

```

        <glob:LastName>Batista</glob:LastName>
      </glob:Row>
    </glob:Input>
  </glob:GlobalSentryRequest>
</soapenv:Body>
</soapenv:Envelope>

```

The SOAP response would be:

Note: Empty response elements have been removed from this example. Only the first response record shown.

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:GlobalSentryResponse
xmlns:ns2="http://www.precisely.com/spectrum/services/GlobalSentry">
      <ns2:Output>
        <ns2:Row>
          <ns2:OverallRiskLevel>10</ns2:OverallRiskLevel>

          <ns2:SanctionedCountryIdentified>No</ns2:SanctionedCountryIdentified>
            <ns2:Status>S</ns2:Status>
            <ns2:FirstName>Miguel</ns2:FirstName>
            <ns2:LastName>Batista</ns2:LastName>
            <ns2:PlaceOfBirth>San Sebastian (Guipuzcoa)
Spain</ns2:PlaceOfBirth>
            <ns2:EntryID>315</ns2:EntryID>

          <ns2:InputFilteredFirstName>Miguel</ns2:InputFilteredFirstName>

          <ns2:InputFilteredLastName>Batista</ns2:InputFilteredLastName>
            <ns2:InputFirstName>Miguel</ns2:InputFirstName>
            <ns2:InputLastName>Batista</ns2:InputLastName>
            <ns2:ListType>DFAT Consolidated List</ns2:ListType>
            <ns2:MatchKey1>MGL</ns2:MatchKey1>
            <ns2:MatchKey2>BTST</ns2:MatchKey2>
            <ns2:NameMatchIdentified>Yes</ns2:NameMatchIdentified>
            <ns2:NameProvided>Yes</ns2:NameProvided>
            <ns2:AddressProvided>No</ns2:AddressProvided>
            <ns2:IDNumberProvided>No</ns2:IDNumberProvided>
            <ns2:AddressMatchIdentified>No</ns2:AddressMatchIdentified>

          <ns2:IDNumberMatchIdentified>No</ns2:IDNumberMatchIdentified>
            <ns2:CitizenshipScore>0</ns2:CitizenshipScore>

          <ns2:CitizenshipMatchIdentified>No</ns2:CitizenshipMatchIdentified>
            <ns2:DOBScore>0</ns2:DOBScore>
            <ns2:DOBMatchIdentified>No</ns2:DOBMatchIdentified>
            <ns2:NationalityScore>0</ns2:NationalityScore>

          <ns2:NationalityMatchIdentified>No</ns2:NationalityMatchIdentified>

```

```

        <ns2:PlaceOfBirthScore>0</ns2:PlaceOfBirthScore>
<ns2:PlaceOfBirthMatchIdentified>No</ns2:PlaceOfBirthMatchIdentified>
        <ns2:CitizenshipProvided>No</ns2:CitizenshipProvided>
        <ns2:DOBProvided>No</ns2:DOBProvided>
        <ns2:NationalityProvided>No</ns2:NationalityProvided>
        <ns2:PlaceOfBirthProvided>No</ns2:PlaceOfBirthProvided>
        <ns2:WatchListFirstName>Miguel</ns2:WatchListFirstName>
        <ns2:WatchListLastName>ALBISU
IRIARTE</ns2:WatchListLastName>
        <ns2:NameScore>100</ns2:NameScore>
        <ns2:user_fields/>
    </ns2:Row>
</ns2:Output>
</ns2:GlobalSentryResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

Table 74: Global Sentry Input Fields

| Parameter | Description |
|--------------|--|
| Name | Full name. Required if FirstName and LastName is not used. |
| FirstName | First name or all name elements other than last name. Required if Name is not used. |
| LastName | Last name only. Required if Name is not used. |
| AddressLine1 | The first address line. Recommended if provided. |
| AddressLine2 | The second address line. Recommended if provided. |

| Parameter | Description |
|--------------|---|
| AddressLine3 | The third address line. Recommended if provided. |
| Country | Full country name. Required if address lines are used. |
| IDNumber | Identification Number, such as SSN, Passport, and Visa. Recommended if provided. |
| PlaceOfBirth | Any place of birth data. Recommended if provided. |
| DOB | Date Of Birth, in the format of Year, Month, Day. Recommended if provided. |
| Citizenship | Full country name. Recommended if provided. |
| Nationality | Full country name. Recommended if provided. |

Response

Table 75: Global Sentry Service Output

| Response Element | Description |
|------------------|---|
| Status | Reports the success or failure of the match attempt. null : Success F : Failure |

| Response Element | Description |
|------------------------|--|
| Status.Code | Reason for failure. |
| Status.Description | Description of the problem that caused the failure. |
| Name | |
| InputName | Input Name from the original data source. |
| InputFilteredName | Input Name with titles, suffixes and special characters removed from the original data source. |
| Name | Name returned from database. |
| InputFirstName | Input First Name from the original data source. |
| InputFilteredFirstName | Input First Name with titles, suffixes and special characters removed from the original data source. |
| FirstName | First Name returned from database. |
| InputLastName | Input Last Name from the original data source. |
| InputFilteredLastName | Input Last Name with titles, suffixes and special characters removed from the original data source. |
| LastName | Last Name returned from database. |
| NameScore | Name match score. 0 - 100. |
| NameMatchIdentified | Identifies if the Name is a match. Values are Yes or No. |

| Response Element | Description |
|------------------------|--|
| NameProvided | Identifies if the Name is provided in the input data . Values are Yes or No. |
| Address | |
| InputAddressLine1 | Input Address line from the original data source. |
| AddressLine1 | Address line returned from database. |
| InputAddressLine2 | Input Address line from the original data source. |
| AddressLine2 | Address line returned from database. |
| InputAddressLine3 | Input Address line from the original data source. |
| AddressLine3 | Address line returned from database. |
| AddressScore | Address match score. 0 - 100. |
| AddressMatchIdentified | Identifies if the Address is a match. Values are Yes or No. |
| AddressProvided | Identifies if the Address is provided in the input data. Values are Yes or No. |
| InputCountry | Input Country from the original data source. |
| Country | Country returned from database. |
| ID Number | |

| Response Element | Description |
|-----------------------------|---|
| InputIDNumber | Input ID Number from the original data source. |
| IDNumber | ID Number returned from database. |
| IDNumberScore | ID Number match score. 0-100. |
| IDNumberMatchIdentified | Identifies if the ID Number is a match. Yes or No. |
| IDNumberProvided | Identifies if the ID Number is provided in the input data. Values are Yes or No. |
| Place of Birth | |
| InputPlaceOfBirth | Input Place of Birth from the original data source. |
| PlaceOfBirth | Place of Birth returned from database. |
| PlaceOfBirthScore | Place of Birth match score. 0-100. |
| PlaceOfBirthMatchIdentified | Identifies if the Place of Birth is a match. Values are Yes or No. |
| PlaceOfBirthProvided | Identifies if the Place of Birth is provided in the input data. Values are Yes or No. |
| Date of Birth | |
| InputDOB | Input Date of Birth from the original data source. |
| DOB | Date of Birth returned from database. |

| Response Element | Description |
|----------------------------|---|
| DOBScore | Date of Birth match score. 0-100. |
| DOBMatchIdentified | Identifies if the Date of Birth is a match. Values are <i>Yes</i> or <i>No</i> . |
| DOBProvided | Identifies if the Date of Birth is provided in the input data. Values are <i>Yes</i> or <i>No</i> . |
| Citizenship | |
| InputCitizenship | Input Citizenship from the original data source. |
| Citizenship | Citizenship returned from database. |
| CitizenshipScore | Citizenship match score. 0 to 100. |
| CitizenshipMatchIdentified | Identifies if Citizenship is a match. Values are <i>Yes</i> or <i>No</i> . |
| CitizenshipProvided | Identifies if the Citizenship is provided in the input data. Values are <i>Yes</i> or <i>No</i> . |
| Nationality | |
| InputNationality | Input Nationality from the original data source. |
| Nationality | Nationality returned from database. |
| NationalityScore | Nationality match score. 0-100. |
| NationalityMatchIdentified | Identifies Nationality was a match. Values are <i>Yes</i> or <i>No</i> . |

| Response Element | Description |
|------------------------------------|--|
| NationalityProvided | Identifies if the Nationality is provided in the input data. Values are <i>Yes</i> or <i>No</i> . |
| Government List Information | |
| EntryID | Entry ID that identifies a name, entity, vessel, address, id number, place of birth, date of birth, citizenship or nationality. This is provided by each govt. agency. |
| ListType | Name of list provided by the government agencies. SDN, EU, Bank Of England, Financial Institutions of Canada. |
| Risk Analysis | |
| OverAllRiskLevel | Risk score per match. 0-16. For more information, see Understanding the Risk Analysis Score on page 722. |
| SanctionedCountryIdentified | Indicates if the sanctioned country is identified as a match. Values are <i>Yes</i> or <i>No</i> . |

Understanding the Risk Analysis Score

Risk analysis processing assigns a point value to each of these inputs depending on whether the input was provided and whether it matched a record in the Global Sentry database. The risk analysis score is the sum of these point values. Points are assigned as shown in this table.

Table 76: Risk Analysis Scoring Method

| Input | No Data Provided | Matched | Did Not Match |
|---------|------------------|---------|---------------|
| Name | 0 | 4 | 0 |
| Address | 1 | 2 | 0 |

| Input | No Data Provided | Matched | Did Not Match |
|----------------|------------------|---------|---------------|
| ID | 1 | 2 | 0 |
| Date of Birth | 1 | 2 | 0 |
| Place of Birth | 1 | 2 | 0 |
| Citizenship | 1 | 2 | 0 |
| Nationality | 1 | 2 | 0 |

Generally, each input that matches the database is assigned 2 points; Name is the exception. A name match scores 4 points. Name score is weighted higher following guidance from sources including OFAC, who indicate that a name match is more significant than other types of matches.

If an input is provided and does not match an entry on the database, it is assigned 0 points and has no effect on the overall risk level. This is consistent with guidance stating that a name match, coupled with a significant amount of additional data which does not match that entry in the database, should not be considered a "hit" against a particular list.

If an input is not provided, it is assigned a score of 1. This has the effect of identifying as higher risk those transactions where one or more inputs match the database, but there are some inputs which are not available for matching. For these types of transactions, the true risk level cannot be accurately calculated because of the missing data. Guidance from agencies such as OFAC suggests that in these cases you should attempt to obtain as much of the missing data as possible in order to return a more accurate assessment of the risk involved in the transaction.

Although higher scores indicate a higher risk transactions, the risk level alone is not always sufficient to determine the appropriate action. This is because different combinations of matched, not-matched, and not-provided inputs can result in the same score. To provide additional information to determine whether an interdiction is appropriate, the Global Sentry service also returns two indicators for each of the seven inputs that are used in matching. These indicate whether the input was provided and whether the input matched the database. This allows you to perform additional analysis on transactions that are in the middle of the risk spectrum to understand whether it is appropriate to report the transaction to the watch list authority, to flag the transaction as needing additional input data for an accurate risk assessment, to approve the transaction, or to take some other action.

Customizing the Global Sentry Service

Global Sentry deploys five dataflow templates that you can modify in Enterprise Designer. Each dataflow consists of various components that were installed from Spectrum Technology Platform Universal Name, Data Normalization, and Advanced Matching.

The names of the dataflows are:

- Global Sentry
- Global Sentry Name Check
- Global Sentry Address Check
- Global Sentry ID Number Check
- Global Sentry Other Data Check
- Global Sentry Batch
- Global Sentry Name Check Batch
- Global Sentry Address Check Batch
- Global Sentry ID Number Check Batch
- Global Sentry Other Data Check Batch

Spectrum Information Extractor

InformationExtractor

InformationExtractor extracts entities such as names and addresses from strings of unstructured data (also known as plain text).

It is possible that not all entities for any selected type will be returned because accuracy varies depending on the type of input. Because Information Extractor uses natural-language processing, a string containing a grammatically correct sentence from a news article or blog would likely have a more accurate return of names than a simple list of names and dates.

Resource URL

```
http://server:port/soap/InformationExtractor
```

Example

This shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:inf="http://www.precisely.com/spectrum/services/InformationExtractor">
```

```

<soapenv:Header/>
<soapenv:Body>
  <inf:InformationExtractorRequest>
    <inf:options>
      <inf:EntityList>Person</inf:EntityList>
    </inf:options>
    <inf:input_port>
      <inf:PlainText>
        <inf:PlainText>My name is Arthur Pitney</inf:PlainText>
      </inf:PlainText>
      <inf:PlainText>
        <inf:PlainText>My name is Walter Bowes</inf:PlainText>
      </inf:PlainText>
    </inf:input_port>
  </inf:InformationExtractorRequest>
</soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:InformationExtractorResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/InformationExtractor">
      <ns3:output_port>
        <ns3:Result>
          <ns3:Entity>
            <ns3:Entity>
              <ns3:Text>Arthur Pitney</ns3:Text>
              <ns3:Type>Person</ns3:Type>
            </ns3:Entity>
          </ns3:Entity>
          <ns3:user_fields/>
        </ns3:Result>
        <ns3:Result>
          <ns3:Entity>
            <ns3:Entity>
              <ns3:Text>Walter Bowes</ns3:Text>
              <ns3:Type>Person</ns3:Type>
            </ns3:Entity>
          </ns3:Entity>
          <ns3:user_fields/>
        </ns3:Result>
      </ns3:output_port>
    </ns3:InformationExtractorResponse>
  </soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

InformationExtractor takes as input unstructured strings of data.

Table 77: Input Format

| Parameter | Description |
|-----------|---|
| PlainText | The unstructured string of data from which you want to extract information. |

Options

The InformationExtractor stage enables you to select entities for output data. It auto-assigns attributes for the entity types that were brought in to this stage. However, you can use the Quick Add function and select any or all of the 15 attributes:

| Parameter | Description |
|-----------------|---|
| CategorizerName | Specifies which model to use for text categorization. |
| CategoryCount | Specifies how many matching levels of the category should be output (closest match, closest plus second closest, etc.). |

| Parameter | Description |
|-------------------|---|
| EntityList | <p>Specifies the type of data you want to extract from the unstructured string. Specify one or more of these. Separate each entity type with a comma.</p> <ul style="list-style-type: none"> Address CreditCard Date Email HashTag ISBN Location Mention Organization Person Phone ProperNouns SSN WebAddress ZipCode |
| OutputEntityCount | <p>Specifies whether to return a count of how many times a particular entity occurred in the output.</p> <ul style="list-style-type: none"> true Return a count of the entities found in the unstructured string. false Do not return a count of the entities found in the unstructured string. |

Response

The output from InformationExtractor is a list of the entities found in the input string. For example, if you selected an entity type of "Person," the output would be a list of the names found in the input string. Likewise, if you selected an entity type of "Date," the output would be a list of the dates found in the input string. Each entity (whether it be a name, address, date, and so on) is returned only once even if the entity appears multiple times in the input string.

| Response Element | Description |
|------------------|--|
| Text | The text extracted from the string. |
| Type | The entity type of the extracted text. One of the following: Address CreditCard Date Email HashTag ISBN Location Mention Organization Person Phone ProperNouns SSN WebAddress ZipCode |
| Count | If the option to return a count is enabled, this field contains the number of times that particular entity appeared in the input. For example, if you choose to return Name entities and the input text contains five instances of the name "John," the name "John" will be included in the output just one time, with "Name" as the entity type, and "5" as the output count. |
| Category | If you used a categorizer, the predicted category for each record in the input file. |
| Rank | If you used a categorizer, the rank of categories from highest count to lowest count. |

Spectrum Spatial

Where to Find Documentation?

Spectrum Spatial provides spatial services that allows you to determine relationships between locations, areas, or points of interest and other business data, and visually show these relationships on a map. These services include:

- Geometry
- Feature
- Mapping
- MapTiling
- Named Resource
- Web Feature Service
- Web Map Service

Spectrum Spatial also includes routing services, which include:

- DescribeDatasets
- GetCapabilities
- GetRoute
- GetRouteCostMatrix
- GetSegmentData
- GetTravelBoundary
- PersistentUpdate

To learn about Spectrum Spatial services, see the *Spectrum Spatial Guide* on support.precisely.com.

GetTravelBoundary (Deprecated)

Important: This stage has been deprecated in the 12.2 release. The **Travel Boundary** stage should be used instead when creating new dataflows.

GetTravelBoundary determines a drive or walk time or distance boundary from a location. This feature obtains polygons corresponding to an isochrone or isodistance calculation. An isochrone is a polygon or set of points representing an area that can be traversed in a network from a starting point in a given amount of time. An isodistance is a polygon or set of points representing the area that is a certain distance from the starting point. The Get Travel Boundary operation (also known as an iso definition) takes a starting point, a unit (linear or time), one or more costs and their associated tags as input and returns the resulting travel boundary. Cost refers to the amount of time or distance to use in calculating an iso. A tag is a string that identifies the cost and is used to match the corresponding result. Multiple costs can be given as input by providing the costs as a “;” delimited string.

GetTravelBoundary is part of Spectrum Spatial.

Note: GetTravelBoundary is only available as a web service. GetTravelBoundary is not available through the Java, C++, C, .NET, or COM APIs.

Resource URL

```
http://server:port/soap/GetTravelBoundary
```

Example

Case 1, Single Cost:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://<server>:<port>/spectrum/services/GetTravelBoundary">

  <soapenv:Header/>
  <soapenv:Body>
    <get:GetTravelBoundaryRequest>
      <get:input_port>
        <get:IsoRouteRequest>
          <get:Latitude>33.751748</get:Latitude>
          <get:Longitude>-84.364014</get:Longitude>
          <get:TravelBoundaryCost>10</get:TravelBoundaryCost>

<get:TravelBoundaryCostUnits>Kilometers</get:TravelBoundaryCostUnits>
        </get:IsoRouteRequest>
      </get:input_port>
    </get:GetTravelBoundaryRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: Some of the points have been removed from this example to shorten it.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns4:GetTravelBoundaryResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.mapinfo.com/midev/service/geometries/v1"

xmlns:ns4="http://<server>:<port>/spectrum/services/GetTravelBoundary">

      <ns4:output_port>
        <ns4:IsoRouteResponse>
          <ns4:IsoNodeResponse/>
```

```

<ns4:IsoPolygonResponse
  xsi:type="ns3:MultiPolygon"
  srsName="epsg:4326"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <ns3:Polygon srsName="epsg:4326">
    <ns3:Exterior>
      <ns3:LineString>
        <ns3:Pos>
          <ns3:X>-84.34868168466456</ns3:X>
          <ns3:Y>33.68373169496257</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.36945064055561</ns3:X>
          <ns3:Y>33.69293307108579</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.3694506405556</ns3:X>
          <ns3:Y>33.69293307108579</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.3694506405556</ns3:X>
          <ns3:Y>33.69303002973829</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.37104825254721</ns3:X>
          <ns3:Y>33.69391558543121</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.37104825254721</ns3:X>
          <ns3:Y>33.6936408692491</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.42163929894845</ns3:X>
          <ns3:Y>33.716054477754355</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.4440058668311</ns3:X>
          <ns3:Y>33.710741143596806</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.43921303085625</ns3:X>
          <ns3:Y>33.72800947960886</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.45678676276404</ns3:X>
          <ns3:Y>33.73376559161287</ns3:Y>
        </ns3:Pos>
        ...
      </ns3:LineString>
    </ns3:Exterior>
  </ns3:Polygon>
</ns4:IsoPolygonResponse>

```

```

        <ns4:user_fields/>
      </ns4:IsoRouteResponse>
    </ns4:output_port>
  </ns4:GetTravelBoundaryResponse>
</soap:Body>
</soap:Envelope>

```

Case 2, Multiple Costs:

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://<server>:<port>/spectrum/services/GetTravelBoundary">

  <soapenv:Header/>
  <soapenv:Body>
    <get:GetTravelBoundaryRequest>
      <get:input_port>
        <get:IsoRouteRequest>
          <get:Latitude>33.751748</get:Latitude>
          <get:Longitude>-84.364014</get:Longitude>
          <get:TravelBoundaryCost>5;10</get:TravelBoundaryCost>

<get:TravelBoundaryCostUnits>Kilometers</get:TravelBoundaryCostUnits>
        </get:IsoRouteRequest>
      </get:input_port>
    </get:GetTravelBoundaryRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

Note: Some of the points have been removed from this example to shorten it.

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns4:GetTravelBoundaryResponse xmlns:ns2="http://spectrum.com/"
      xmlns:ns3="http://www.mapinfo.com/midev/service/geometries/v1"
      xmlns:ns4="http://<server>:<port>/spectrum/services/GetTravelBoundary">

      <ns4:output_port>
        <ns4:IsoRouteResponse>
          <ns4:cost>5</ns4:cost>
          <ns4:costUnits>Kilometers</ns4: costUnits >
          <ns4:IsoNodeResponse/>
            <ns4:IsoPolygonResponse
              xsi:type="ns3:MultiPolygon"
              srsName="epsg:4326"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

              <ns3:Polygon srsName="epsg:4326">

```

```

<ns3:Exterior>
  <ns3:LineString>
    <ns3:Pos>
      <ns3:X>-84.34868168466456</ns3:X>
      <ns3:Y>33.68373169496257</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.36945064055561</ns3:X>
      <ns3:Y>33.69293307108579</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.3694506405556</ns3:X>
      <ns3:Y>33.69293307108579</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.3694506405556</ns3:X>
      <ns3:Y>33.69303002973829</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.37104825254721</ns3:X>
      <ns3:Y>33.69391558543121</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.37104825254721</ns3:X>
      <ns3:Y>33.6936408692491</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.42163929894845</ns3:X>
      <ns3:Y>33.716054477754355</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.4440058668311</ns3:X>
      <ns3:Y>33.710741143596806</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.43921303085625</ns3:X>
      <ns3:Y>33.72800947960886</ns3:Y>
    </ns3:Pos>
    <ns3:Pos>
      <ns3:X>-84.45678676276404</ns3:X>
      <ns3:Y>33.73376559161287</ns3:Y>
    </ns3:Pos>
    ...
  </ns3:LineString>
</ns3:Exterior>
</ns3:Polygon>
</ns4:IsoPolygonResponse>
<ns4:user_fields/>
</ns4:IsoRouteResponse>
<ns4:IsoRouteResponse>
  <ns4:cost>10</ns4:cost>
  <ns4:costUnits>Kilometers</ns4: costUnits >
  <ns4:IsoNodeResponse/>

```

```

<ns4:IsoPolygonResponse
  xsi:type="ns3:MultiPolygon"
  srsName="epsg:4326"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <ns3:Polygon srsName="epsg:4326">
    <ns3:Exterior>
      <ns3:LineString>
        <ns3:Pos>
          <ns3:X>-84.34868168466456</ns3:X>
          <ns3:Y>33.68373169496257</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.36945064055561</ns3:X>
          <ns3:Y>33.69293307108579</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.3694506405556</ns3:X>
          <ns3:Y>33.69293307108579</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.3694506405556</ns3:X>
          <ns3:Y>33.69303002973829</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.37104825254721</ns3:X>
          <ns3:Y>33.69391558543121</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.37104825254721</ns3:X>
          <ns3:Y>33.6936408692491</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.42163929894845</ns3:X>
          <ns3:Y>33.716054477754355</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.4440058668311</ns3:X>
          <ns3:Y>33.710741143596806</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.43921303085625</ns3:X>
          <ns3:Y>33.72800947960886</ns3:Y>
        </ns3:Pos>
        <ns3:Pos>
          <ns3:X>-84.45678676276404</ns3:X>
          <ns3:Y>33.73376559161287</ns3:Y>
        </ns3:Pos>
        ...
      </ns3:LineString>
    </ns3:Exterior>
  </ns3:Polygon>
</ns4:IsoPolygonResponse>

```

```

        <ns4:user_fields/>
      </ns4:IsoRouteResponse>
    </ns4:output_port>
  </ns4:GetTravelBoundaryResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

GetTravelBoundary takes cost, cost unit, point latitude, and point longitude as input. The following table provides information on the format and layout of the input.

Table 78: GetTravelBoundary Input Data

| Parameter | Format | Description |
|--------------------|--------|---|
| Latitude | String | Latitude of the point. Specify latitude in the format chosen in the CoordinateFormat option. |
| Longitude | String | Longitude of the point. Specify longitude in the format chosen in the CoordinateFormat option. |
| TravelBoundaryCost | String | <p>(Optional) The cost distance or time, in the units specified by either the TravelBoundaryCostUnits field or the DefaultTravelBoundaryCostUnits option. For example, if the unit specified is miles and you specify 10 in this field, the cost would be 10 miles.</p> <p>Use this field to override the default travel boundary cost on a record-by-record basis.</p> <p>You can also specify multiple costs by specifying the values as a ";" delimited string. It will return a separate Iso Route Response for every cost specified. If you specify multiple costs, every response will have cost and costUnits associated with that response.</p> |

| Parameter | Format | Description |
|-------------------------|--------|---|
| TravelBoundaryCostUnits | String | <p>(Optional) The type of metric used to calculate the travel boundary. One of the following:</p> <ul style="list-style-type: none"> • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Yards <p>Use this field to override the default travel boundary cost units on a record-by-record basis.</p> |

Parameters for Options

Input

Table 79: GetTravelBoundary Input Options

| Parameter | Description |
|---------------------|---|
| DataSetResourceName | The name of the database that contains the data to use in the search process. Use a valid routing database resource name defined in the Resources section of the Management Console. For more information, see the <i>Spectrum Technology Platform Spatial Guide</i> . |
| CoordinateSystem | <p>The coordinate system of the input coordinates.</p> <p>For more information on EPSG codes, see www.spatialreference.org. To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <code>http://<server>:<port>/Spatial/GeometryService/DemoPage.html</code></p> |

| Parameter | Description | | | | | | | | | | |
|--------------------------------|---|----------------|-------------------------|-----------------------|------------------------------------|------------------------------|-------------------------|----------------|----------------------------|-----------------------|----------------------------|
| CoordinateFormat | <p>Specifies the format of latitude/longitude coordinates in the input.</p> <p>Note: Use this option only if you specify a Latitude/Longitude coordinate system. If the coordinate system is not a Latitude/Longitude coordinate system, set the coordinate format to Decimal.</p> <p>One of the following:</p> <table> <tr> <td>Decimal</td> <td>(90.000000, 180.000000)</td> </tr> <tr> <td>DecimalAssumed</td> <td>(90000000, 180000000). Default.</td> </tr> <tr> <td>DegreesMinutesSeconds</td> <td>(90 00 00N, 180 00 00W)</td> </tr> <tr> <td>PreZero</td> <td>(0900000000N, 1800000000W)</td> </tr> <tr> <td>PreZeroDecimal</td> <td>(090.000000N, 180.000000W)</td> </tr> </table> | Decimal | (90.000000, 180.000000) | DecimalAssumed | (90000000, 180000000). Default. | DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | PreZero | (0900000000N, 1800000000W) | PreZeroDecimal | (090.000000N, 180.000000W) |
| Decimal | (90.000000, 180.000000) | | | | | | | | | | |
| DecimalAssumed | (90000000, 180000000). Default. | | | | | | | | | | |
| DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | | | | | | | | | | |
| PreZero | (0900000000N, 1800000000W) | | | | | | | | | | |
| PreZeroDecimal | (090.000000N, 180.000000W) | | | | | | | | | | |
| DefaultTravelBoundaryCost | Number of cost units. This can be any double value (includes decimals). The default is 10. | | | | | | | | | | |
| DefaultTravelBoundaryCostUnits | <p>Type of metric you want to use to calculate the travel boundary. One of the following:</p> <ul style="list-style-type: none"> • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Milliseconds • Yards | | | | | | | | | | |

| Parameter | Description |
|---------------------------|---|
| HistoricTrafficTimeBucket | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |

Output

Table 80: GetTravelBoundary Output Options

| Parameter | Description |
|----------------------|--|
| ResultType | <p>Specifies the type of result you want returned. One of the following:</p> <p>AccessibleNodes Returns all of the points along the road network that can be reached for the isoChrone calculation.</p> <p>Geometry Returns the entire isoChrone.</p> <p>StartNodes Returns the location specified.</p> |
| SimplificationFactor | Specifies what percentage of the original points should be returned or upon which the resulting polygon should be based. |

| Parameter | Description |
|----------------|---|
| BandingStyle | Specifies the style of banding to be used in the result. Banding styles are the types of multiple isoChrones or distance bands that can be displayed based on multiple costs. |
| | Donut Each boundary is determined by subtracting out the next smallest boundary. |
| | Encompassing Each boundary is determined independent of all others. |
| ReturnHoles | Specifies whether you want to return holes, which are areas within the larger boundary that cannot be reached within the desired time or distance, based on the road network. |
| | Y Yes, return holes. |
| | N Do not return holes. Default. |
| ReturnsIslands | Specifies whether you want to return islands, which are small areas outside the main boundary that can be reached within the desired time or distance. |
| | Y Yes, return islands. |
| | N Do not return islands. Default. |

Travel

Travel options specify assumptions to make about travel speed off network roads and whether to use only major roads when calculating the travel boundary. Most travel options have to do with ambient speed.

Table 81: GetTravelBoundary Travel Options

| Parameter | Description |
|------------------------|--|
| MaximumOffRoadDistance | Specifies the maximum distance to allow travel off the road network. Examples of off-network roads include driveways and access roads. For example, if you specify a maximum off road distance of 1 mile the travel boundary will extend no further than one mile from the network road. If you specify a value of 0 the travel boundary will not extend beyond the road itself. Use the ambient speed options to specify the speed of travel along non-network roads. |

| Parameter | Description |
|-----------|--|
| Units | The units of measure in which you want the data returned. One of the following: <ul style="list-style-type: none"><li data-bbox="677 457 894 485">• Kilometer (default)<li data-bbox="677 491 760 518">• Meter<li data-bbox="677 525 743 552">• Mile |

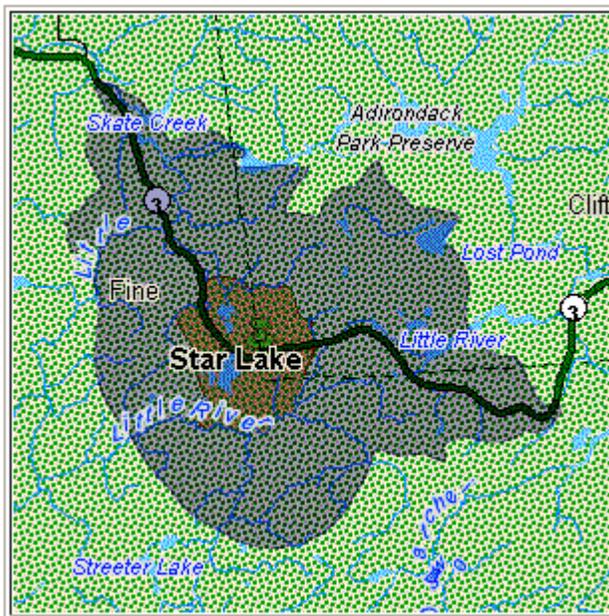
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|------------|--|
| MajorRoads | |
|------------|--|

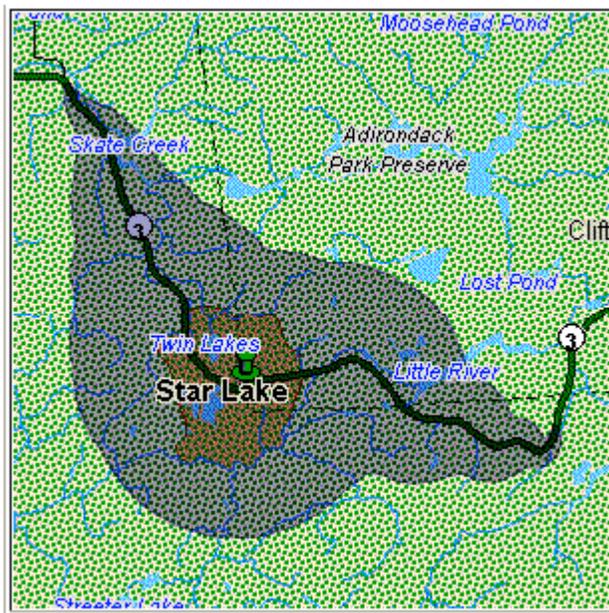
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Specifies whether to include all roads in the calculation or just major roads. By default, the Get Travel Boundary is calculated with major roads set to true. This improves performance but the accuracy may decrease.

This map represents a travel boundary with travel allowed on all roads:



This map represents a travel boundary with travel restricted to major roads only:



| Parameter | Description |
|---------------------|---|
| | <p>One of the following:</p> <p>Y Include only major roads in the calculation. Default.</p> <p>N Include all roads in the calculation.</p> |
| DefaultAmbientSpeed | <p>Specifies the speed to travel when going off a network road to find the travel boundary. Examples of off-network travel include driveways and access roads.</p> <p>This option is available only when you specify a time value in the DefaultCostUnits option or the TravelBoundaryCostUnits field. The default is 15. Specify the speed units in the AmbientSpeedUnit option.</p> <p>To control how off-network travel is used in the travel boundary calculation, you need to specify the speed of travel off the road network (the ambient speed). Ambient speed can affect the size and shape of the travel boundary polygon. In general, the faster the ambient speed, the larger the polygon. For example, if you were at a point with 5 minutes left, and if the ambient speed were 15 miles per hour, boundary points would be put at a distance of 1.25 miles. If the ambient speed were reduced to 10 miles per hour, boundary points would be put at a distance of 0.83 miles. Note that you can limit the distance allowed off a network road by using the MaximumOffRoadDistance option.</p> <p>Note: If you are calculating pedestrian travel boundaries we recommend that you change the default ambient speed to 3 MPH (5 KPH).</p> |
| AmbientSpeedUnit | <p>The unit of measure to use with the value specified in the DefaultAmbientSpeed option.</p> <p>KPH Kilometers per hour.</p> <p>MPH MILES per hour. Default.</p> <p>MTPS Meters per second.</p> <p>MTPM Meters per minute.</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

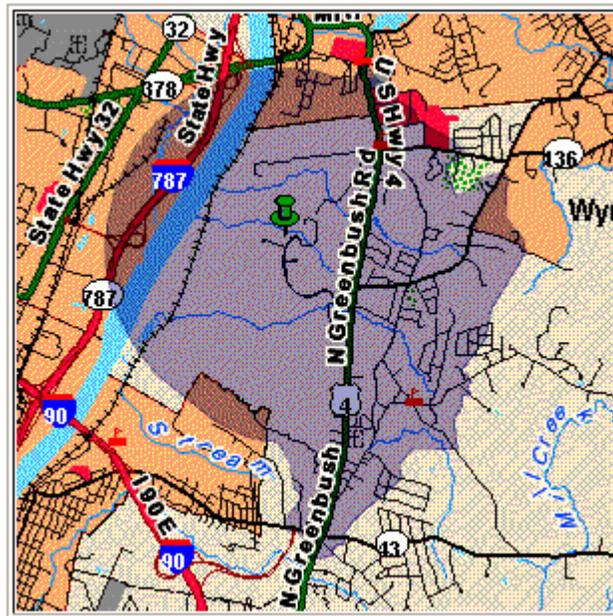
| | |
|------------------------------|--|
| AmbientSpeed.RoadType.<Type> | |
|------------------------------|--|

Parameter

Description

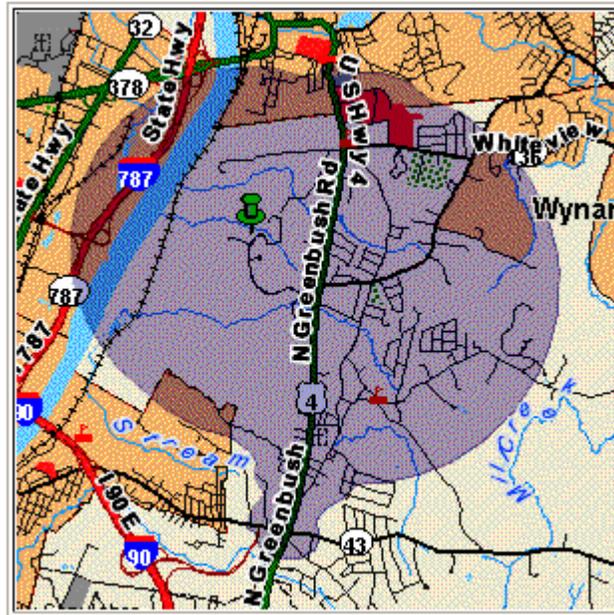
Specifies the ambient speed to use for off-network travel based on the road type. If you do not specify an ambient speed for a road type, the default ambient speed will be used, as specified in the DefaultAmbientSpeed option.

The following map shows an example of a travel boundary without ambient speed overrides:



For comparison, this map shows the same travel boundary with ambient speed overrides:

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|



<Type> can be one of the following:

| Parameter | Description |
|-----------|--|
| | <ul style="list-style-type: none"> • AccessWay • Backroad • Connector • Ferry • Footpath • LimitedAccessDenseUrban • LimitedAccessRural • LimitedAccessSuburban • LimitedAccessUrban • LocalRoadDenseUrban • LocalRoadRural • LocalRoadSuburban • LocalRoadUrban • MajorLocalRoadDenseUrban • MajorLocalRoadRural • MajorLocalRoadSuburban • MajorLocalRoadUrban • MajorRoadDenseUrban • MajorRoadRural • MajorRoadSuburban • MajorRoadUrban • MinorLocalRoadDenseUrban • MinorLocalRoadRural • MinorLocalRoadSuburban • MinorLocalRoadUrban • NormalRoadDenseUrban • NormalRoadRural • NormalRoadRural • NormalRoadUrban • PrimaryHighwayDenseUrban • PrimaryHighwayRural • PrimaryHighwaySuburban • PrimaryHighwayUrban • RampDenseUrban • RampLimitedAccess • RampMajorRoad • RampPrimaryHighway • RampRural • RampSecondaryHighway • RampUrban • RampSuburban • SecondaryHighwayDenseUrban • SecondaryHighwayRural • SecondaryHighwaySuburban • SecondaryHighwayUrban |

determine the risk of an insured property or person based on the probability that ambulance, public safety, or fire personnel can reach the property/person in a reasonable amount of time.

Note: Get Travel Cost Matrix is only available as a SOAP web service. Get Travel Cost Matrix is not available through REST. It is also not available through the Java, C++, C, .NET, or COM APIs.

GetTravelCostMatrix is part of the Enterprise Routing Module.

Resource URL

```
http://server:port/soap/GetTravelCostMatrix
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:get="http://<server>:<port>/spectrum/services/GetTravelCostMatrix"

xmlns:spec="http://spectrum.com/"
xmlns:get1="http://www.g1.com/services/GetTravelCostMatrix"
xmlns:typ="http://www.g1.com/services/erm/types
<soapenv:Header/>
<soapenv:Body>
  <get:GetTravelCostMatrixRequest>
    <get:input_port>
      <get:RouteMatrixRequest>
        <get:StartPoints>
          <get:StartPoint>
            <get:Latitude>33.751748</get:Latitude>

            <get:Longitude>-84.364014</get:Longitude>
          </get:StartPoint>
          <get:StartPoint>
            <get:Latitude>33.870416</get:Latitude>
            <get:Longitude>-78.62915</get:Longitude>
          </get:StartPoint>
          <get:StartPoint>
            <get:Latitude>35.025498</get:Latitude>
            <get:Longitude>-80.864868</get:Longitude>
          </get:StartPoint>
        </get:StartPoints>
        <get:EndPoints>
          <get:EndPoint>
            <get:Latitude>33.664925</get:Latitude>
            <get:Longitude>-80.90332</get:Longitude>
          </get:EndPoint>
```

```

        <get:EndPoint>
            <get:Latitude>34.40691</get:Latitude>
            <get:Longitude>-80.062866</get:Longitude>
        </get:EndPoint>
        <get:EndPoint>
            <get:Latitude>34.921971</get:Latitude>
            <get:Longitude>-81.013184</get:Longitude>
        </get:EndPoint>
    </get:Endpoints>
</get:RouteMatrixRequest>
</get:input_port>
</get:GetTravelCostMatrixRequest>
</soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns5:GetTravelCostMatrixResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.g1.com/services/erm/types"
xmlns:ns4="http://www.g1.com/services/GetTravelCostMatrix"

xmlns:ns5="http://<server>:<port>/spectrum/services/GetTravelCostMatrix">
      <ns5:output_port>
        <ns5:RouteMatrixResponse>
          <ns5:TimeUnits>Minutes</ns5:TimeUnits>
          <ns5:DistanceUnits>Miles</ns5:DistanceUnits>
          <ns5:RouteCosts>
            <ns5:RouteCost>
              <ns5:StartPointRef>1</ns5:StartPointRef>
              <ns5:EndPointRef>1</ns5:EndPointRef>
              <ns5:Time>215.82</ns5:Time>
              <ns5:Distance>218.441</ns5:Distance>
            </ns5:RouteCost>
            <ns5:RouteCost>
              <ns5:StartPointRef>2</ns5:StartPointRef>
              <ns5:EndPointRef>2</ns5:EndPointRef>
              <ns5:Time>124.82</ns5:Time>
              <ns5:Distance>103.437</ns5:Distance>
            </ns5:RouteCost>
            <ns5:RouteCost>
              <ns5:StartPointRef>3</ns5:StartPointRef>
              <ns5:EndPointRef>3</ns5:EndPointRef>
              <ns5:Time>22.53</ns5:Time>
              <ns5:Distance>15.005</ns5:Distance>
            </ns5:RouteCost>
          </ns5:RouteCosts>
          <ns5:user_fields/>
        </ns5:RouteMatrixResponse>

```

```

    </ns5:output_port>
  </ns5:GetTravelCostMatrixResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

The input to Get Travel Cost Matrix is a list of start points and end points expressed as latitude/longitude coordinates. You can also include user-defined fields.

The order of start and end points in the input determines the order of the sequences in the response. For example, if you specify two start points and two end points in the request, the response will contain the following order of sequences (where S is start and E is end): S1 to E1, S1 to E2, S2 to E1, S2 to E2.

Note: The Get Travel Cost Matrix service is only available as a web service. The Get Travel Cost Matrix is not available through the Java, C++, C, .NET, or COM APIs.

Table 83: Get Travel Cost Matrix Input

| Parameter | Description |
|-----------|---|
| Latitude | The latitude for a start or end point. Specify the latitude using the format selected in the <code>CoordinateFormat</code> option. |
| Longitude | The longitude for a start or end point. Specify the longitude using the format selected in the <code>CoordinateFormat</code> option. |
| ID | An identification you assign to the point. Specify an ID comprised of alpha and/or numeric characters to represent a point. This ID corresponds to the <code>StartPointID</code> or <code>EndPointID</code> field in output. |
| TollRoad | This feature specifies whether you want a route with or without a toll road. This is a Boolean type parameter. The default value is <code>False</code> . If you set the value of <code>TollRoad</code> to <code>True</code> , the response contains a route without any toll roads. If the value of <code>TollRoad</code> is set to <code>False</code> , the route includes toll roads. |

| Parameter | Description |
|-----------------------|---|
| MatrixTransientUpdate | <p>A schema containing the update types for the transient update. Transient updates are updates made on a request that only apply to that particular request. Transient updates are similar to Persistent updates, except that Transient updates are only for a particular request and Persistent updates are for all the requests. You have the ability to set a speed for a point, a segment id, or a road class, as well as the ability to update the road class for a segment (specified by the segment id).</p> <p>For transient update options and example, see ../GetTravelCostMatrix/GetTravelCostMatrix_TransientOptions.dita.</p> |

Parameters for Options

Travel

This set of preferences allows you to set the desirability for each road type. For instance, you can request that the server attempt to avoid all of the major road types.

Table 84: Travel Preferences Options

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-----------------|--|
| RoadType_<type> | |
|-----------------|--|

| Parameter | Description |
|-----------|--|
| | <p data-bbox="548 369 1424 396">Specifies the priority to give to different types of roads when determining the route.</p> <ul style="list-style-type: none"> <li data-bbox="548 422 695 449">• access way <li data-bbox="548 453 678 480">• back road <li data-bbox="548 485 678 512">• connector <li data-bbox="548 516 623 543">• ferry <li data-bbox="548 548 662 575">• footpath <li data-bbox="548 579 862 606">• limited access dense urban <li data-bbox="548 611 776 638">• limited access rural <li data-bbox="548 642 829 669">• limited access suburban <li data-bbox="548 674 792 701">• limited access urban <li data-bbox="548 705 813 732">• local road dense urban <li data-bbox="548 737 732 764">• local road rural <li data-bbox="548 768 781 795">• local road suburban <li data-bbox="548 800 743 827">• local road urban <li data-bbox="548 831 878 858">• major local road dense urban <li data-bbox="548 863 797 890">• major local road rural <li data-bbox="548 894 846 921">• major local road suburban <li data-bbox="548 926 808 953">• major local road urban <li data-bbox="548 957 824 984">• major road dense urban <li data-bbox="548 989 743 1016">• major road rural <li data-bbox="548 1020 792 1047">• major road suburban <li data-bbox="548 1052 753 1079">• major road urban <li data-bbox="548 1083 878 1110">• minor local road dense Urban <li data-bbox="548 1115 797 1142">• minor local road rural <li data-bbox="548 1146 846 1173">• minor local road suburban <li data-bbox="548 1178 808 1205">• minor local road urban <li data-bbox="548 1209 846 1236">• normal road dense urban <li data-bbox="548 1241 753 1268">• normal road rural <li data-bbox="548 1272 753 1299">• normal road rural <li data-bbox="548 1304 764 1331">• normal road urban <li data-bbox="548 1335 878 1362">• primary highway dense urban <li data-bbox="548 1367 797 1394">• primary highway rural <li data-bbox="548 1398 846 1425">• primary highway suburban <li data-bbox="548 1430 808 1457">• primary highway urban <li data-bbox="548 1461 764 1488">• ramp dense urban <li data-bbox="548 1493 781 1520">• ramp limited access <li data-bbox="548 1524 748 1551">• ramp major road <li data-bbox="548 1556 808 1583">• ramp primary highway <li data-bbox="548 1587 683 1614">• ramp rural <li data-bbox="548 1619 829 1646">• ramp secondary highway <li data-bbox="548 1650 695 1677">• ramp urban <li data-bbox="548 1682 732 1709">• ramp suburban <li data-bbox="548 1713 911 1740">• secondary highway dense urban <li data-bbox="548 1745 829 1772">• secondary highway rural |

| Parameter | Description |
|-----------|---|
| | <ul style="list-style-type: none">• secondary highway suburban• secondary highway urban <p>For each road type you can specify one of the following:</p> <p>Avoid Exclude the road type from routes if possible.</p> <p>Note: It is not always possible to exclude a road type from the travel directions. Depending on the situation, the alternative to an avoided road type may be so poor that the software will choose a route that uses an avoided road type. Also, if the starting or ending point lies along a segment whose road type has been avoided, the software will still use that segment.</p> <p>High Prefer the road type over other road types.</p> <p>Low Prefer other road types over this road type.</p> <p>Medium Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium.</p> |

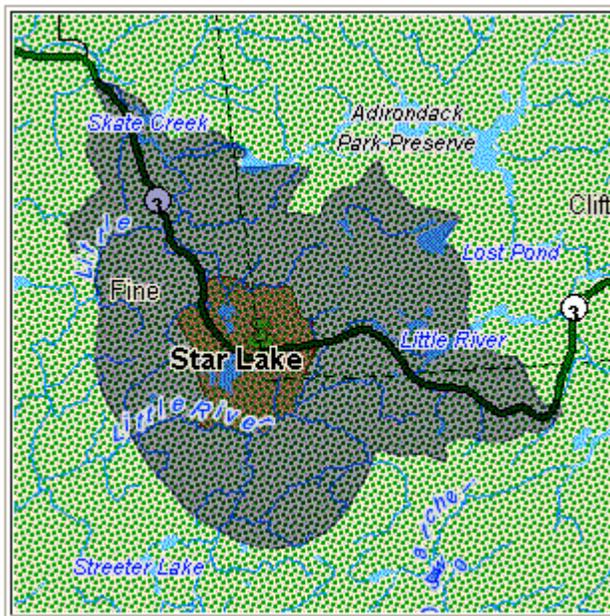
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|------------|--|
| MajorRoads | |
|------------|--|

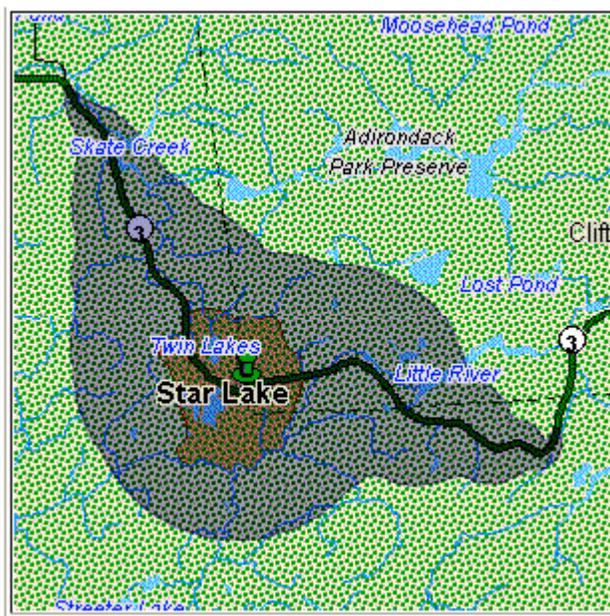
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Specifies whether to include all roads in the calculation or just major roads. If you choose to include only major roads, performance will improve but accuracy may decrease.

This map represents a travel boundary with travel allowed on all roads:



This map represents a travel boundary with travel restricted to major roads only:



| Parameter | Description |
|--------------------------------|---|
| <hr/> One of the following: | |
| Y | Include only major roads in the calculation. Default. |
| N | Include all roads in the calculation. |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Avoid

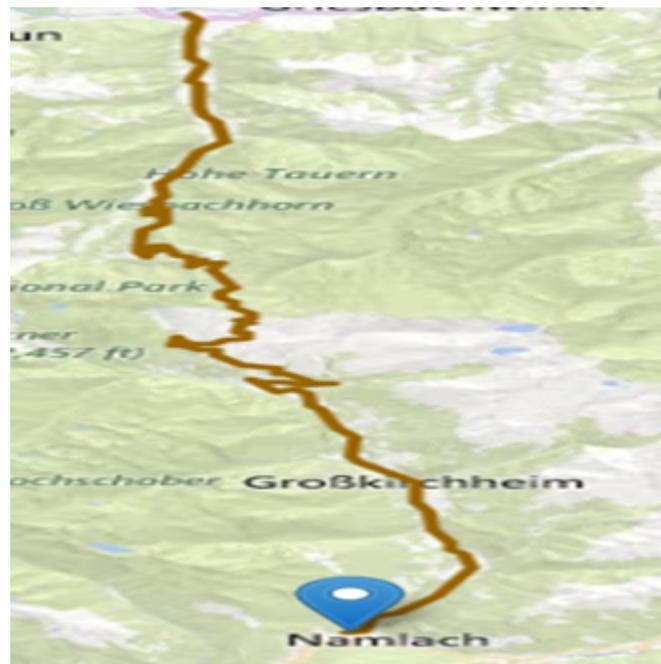
Toll Roads

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Specifies whether you want a route with or without a toll road. The stage GetTravelCostMatrix GetTravelDirections contains the "avoid Toll Roads" feature. There is a check box labeled "Toll Roads" on the UI. You can check it to avoid toll roads. You can also add or expose this parameter from input value as "TollRoad". The input value can contain Boolean values where False is the default value.

For Example:

Following route contains toll road information, which is mentioned in the following image:



Now, for the same points, if you check the toll Road or set the "TollRoad" parameter as True, the response contains a route without any toll roads. See the following response:

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|



Routing

Table 85: Routing Options

| Parameter | Description |
|---------------------|--|
| DataSetResourceName | The name of the database that contains the data to use in the search process. Use a valid routing database resource name defined in the Resources section of the Management Console. For more information, see the <i>Spectrum Technology Platform Spatial Guide</i> . |

| Parameter | Description |
|-------------------------|---|
| OptimizeBy | <p>Specifies whether to find the shortest distance or the shortest time.</p> <p>One of the following:</p> <p>Time Optimize by fastest time traveled. Default.</p> <p>Distance Optimize by shortest distance traveled.</p> |
| ReturnOptimalRoutesOnly | <p>Specifies whether to return only the optimized route for each start point/end point combination. The optimized route is either the fastest route or the shortest distance, depending on the selection you make for the OptimizeBy option.</p> <p>For example, if you have one start point (S1) and two end points (E1 and E2) and it is faster to go from S1 to E2 than from S1 to E1, and you choose to return only the optimal route, then only the route cost information for S1 to E2 will be returned.</p> <p>Y Yes, return only optimal routes. Default.</p> <p>N No, return all routes not just the optimal routes.</p> |
| RouteCostMatrixFormat | <p>Specifies the data format of the route cost matrix. One of the following:</p> <p>Hierarchy The route cost matrix is returned as list that you can view in Management Console or Enterprise Designer. Choose this option if you want to view the data in these tools or if you want to work with the data in tabular format. Default.</p> <p>Object The route cost matrix is returned as a data object that you cannot interact with directly. Only select this option if the output is going directly to another stage in a dataflow or a web service call, or if it will be used by another application to visualize the routes, such as input to mapping software.</p> |
| CoordinateSystem | <p>The coordinate system of the input coordinates.</p> <p>For more information on EPSG codes, see www.spatialreference.org. To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <a href="http://<server>:<port>/Spatial/GeometryService/DemoPage.html">http://<server>:<port>/Spatial/GeometryService/DemoPage.html.</p> |

| Parameter | Description | | | | | | | | | | |
|-----------------------|---|----------------|-----------------------------------|-----------------------|----------------------------------|------------------|----------------------------|---------------------|------------------------------------|-----------------------|----------------------------|
| CoordinateFormat | <p>Specifies the format of latitude/longitude coordinates in the input.</p> <p>Note: Use this option only if you specify a Latitude/Longitude coordinate system. If the coordinate system is not a Latitude/Longitude coordinate system, set the coordinate format to Decimal.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Decimal</td> <td>(90.000000, 180.000000). Default.</td> </tr> <tr> <td>DecimalAssumed</td> <td>(90000000, 180000000).</td> </tr> <tr> <td>DegMinSec</td> <td>(90 00 00N, 180 00 00W)</td> </tr> <tr> <td>PreZero</td> <td>(090000000N, 180000000W)</td> </tr> <tr> <td>PreZeroDecimal</td> <td>(090.000000N, 180.000000W)</td> </tr> </table> | Decimal | (90.000000, 180.000000). Default. | DecimalAssumed | (90000000, 180000000). | DegMinSec | (90 00 00N, 180 00 00W) | PreZero | (090000000N, 180000000W) | PreZeroDecimal | (090.000000N, 180.000000W) |
| Decimal | (90.000000, 180.000000). Default. | | | | | | | | | | |
| DecimalAssumed | (90000000, 180000000). | | | | | | | | | | |
| DegMinSec | (90 00 00N, 180 00 00W) | | | | | | | | | | |
| PreZero | (090000000N, 180000000W) | | | | | | | | | | |
| PreZeroDecimal | (090.000000N, 180.000000W) | | | | | | | | | | |
| DistanceUnits | <p>Specifies how GetTravelCostMatrix should return distance values. One of the following:</p> <table border="0"> <tr> <td>Feet</td> <td>Return distance in feet.</td> </tr> <tr> <td>Kilometers</td> <td>Return distance in kilometers.</td> </tr> <tr> <td>Meters</td> <td>Return distance in meters.</td> </tr> <tr> <td>Miles</td> <td>Return distance in miles. Default.</td> </tr> <tr> <td>Yards</td> <td>Return distance in yards.</td> </tr> </table> | Feet | Return distance in feet. | Kilometers | Return distance in kilometers. | Meters | Return distance in meters. | Miles | Return distance in miles. Default. | Yards | Return distance in yards. |
| Feet | Return distance in feet. | | | | | | | | | | |
| Kilometers | Return distance in kilometers. | | | | | | | | | | |
| Meters | Return distance in meters. | | | | | | | | | | |
| Miles | Return distance in miles. Default. | | | | | | | | | | |
| Yards | Return distance in yards. | | | | | | | | | | |
| TimeUnits | <p>Specifies how to return time. One of the following:</p> <table border="0"> <tr> <td>Hours</td> <td>Return time in hours.</td> </tr> <tr> <td>Minutes</td> <td>Return time in minutes. Default.</td> </tr> <tr> <td>Seconds</td> <td>Return time in seconds.</td> </tr> <tr> <td>Milliseconds</td> <td>Return time in milliseconds.</td> </tr> </table> | Hours | Return time in hours. | Minutes | Return time in minutes. Default. | Seconds | Return time in seconds. | Milliseconds | Return time in milliseconds. | | |
| Hours | Return time in hours. | | | | | | | | | | |
| Minutes | Return time in minutes. Default. | | | | | | | | | | |
| Seconds | Return time in seconds. | | | | | | | | | | |
| Milliseconds | Return time in milliseconds. | | | | | | | | | | |

| Parameter | Description |
|---------------------------|---|
| HistoricTrafficTimeBucket | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <p>None The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</p> <p>AMPeak Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</p> <p>PMPeak Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</p> <p>OffPeak Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</p> <p>Night Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</p> |

Transient Options

This set of preferences allows you to set transient updates for each request. For instance, you can request that the server attempt to avoid all of the major road types. Each request can contain one or more updates.

Note: The transient update functionality is only available through the SOAP API, and is not available through the Management Console or Enterprise Designer.

Table 86: Transient Update Options

| Parameter | Description |
|-------------|---|
| PointUpdate | <p>Point updates are changes applied to a corresponding point (Latitude, Longitude). For a particular point, you can: exclude the point, set the speed of the point or change (increase or decrease) the speed of the point by a value or percentage. You must specify a <code>Point</code> which consists of a <code>Latitude</code> and <code>Longitude</code>, then specify one of the following:</p> <p>Velocity This is a speed update where you define the new speed of the point by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedIncrease This is a speed update where you define an increase in the speed of the point by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedDecrease This is a speed update where you define a decrease in the speed of the point by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>Exclude This is a sting value to exclude the specified point from the route calculation. To exclude a point you need to specify the point and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|---------------|--|
| SegmentUpdate | |
|---------------|--|

| Parameter | Description |
|----------------------|--|
| | <p>Segment updates are changes applied to a corresponding segment ID (Latitude, Longitude). For a particular segment, you can: exclude the segment, set the speed of the segment, change (increase or decrease) the speed of the segment by a value or percentage, or change the road type of the segment. You must specify a valid <code>RoutingSegmentID</code>, then specify one of the following:</p> |
| Velocity | <p>This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| SpeedIncrease | <p>This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| SpeedDecrease | <p>This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| RoadType | <p>This is a string value to change the value of the road type for the segment for the route calculation.</p> <p>The <code>RoadType</code> can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural |

| Parameter | Description |
|----------------|---|
| | <ul style="list-style-type: none"> • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban • ramp suburban • secondary highway dense urban • secondary highway rural • secondary highway suburban • secondary highway urban |
| Exclude | <p>This is a sting value to exclude the specified segment from the route calculation. To exclude a segment you need to specify the segment ID and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Parameter | Description |
|----------------|---|
| RoadTypeUpdate | <p>Road type updates are changes applied to a corresponding road type. For a particular road type, you can: set the speed of the roadtype, or change (increase or decrease) the speed of the road type by a value or percentage. You must specify a <code>RoadType</code> to update (see the above road types in the segment update), then specify one of the following:</p> <p>Velocity This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedIncrease This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedDecrease This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |

GetTravelCostMatrix Transient Update SOAP Example

The following shows a standard GetTravelCostMatrix SOAP request with all of the transient update options available (not a working example, this is used to show all the syntax). Each request can have a `MatrixTransientUpdate` which will be used to calculate each route matrix. You can have multiple Update definitions within a `MatrixTransientUpdate`. You can only have a single update type (`PointUpdate`, `SegmentUpdate`, or `RoadTypeUpdate`) within an `Update`. You can also only have a single update within one of the update types (`PointUpdate`, `SegmentUpdate`, or `RoadTypeUpdate`).

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:get="http://<server><port>/spectrum/services/GetTravelCostMatrix"

xmlns:spec="http://spectrum.precisely.com/"
xmlns:get1="http://www.g1.com/services/GetTravelCostMatrix"
xmlns:typ="http://www.g1.com/services/erm/types">
<soapenv:Header/>
<soapenv:Body>
```

```

<get:GetTravelCostMatrixRequest>
  <get:input_port>
    <get:RouteMatrixRequest>
      <get:StartPoints>
        <get:StartPoint>
          <get:Latitude>33.751748</get:Latitude>

          <get:Longitude>-84.364014</get:Longitude>
        </get:StartPoint>
        <get:StartPoint>
          <get:Latitude>33.870416</get:Latitude>
          <get:Longitude>-78.62915</get:Longitude>
        </get:StartPoint>
        <get:StartPoint>
          <get:Latitude>35.025498</get:Latitude>
          <get:Longitude>-80.864868</get:Longitude>
        </get:StartPoint>
      </get:StartPoints>
      <get:EndPoints>
        <get:EndPoint>
          <get:Latitude>33.664925</get:Latitude>
          <get:Longitude>-80.90332</get:Longitude>
        </get:EndPoint>
        <get:EndPoint>
          <get:Latitude>34.40691</get:Latitude>
          <get:Longitude>-80.062866</get:Longitude>
        </get:EndPoint>
        <get:EndPoint>
          <get:Latitude>34.921971</get:Latitude>
          <get:Longitude>-81.013184</get:Longitude>
        </get:EndPoint>
      </get:EndPoints>
    </get:RouteMatrixRequest>
  </get:input_port>
</get:GetTravelCostMatrixRequest>
<get:MatrixTransientUpdate>
  <typ:Update>
    <typ:PointUpdate>
      <typ:Point>
        <typ:Latitude>?</typ:Latitude>
        <typ:Longitude>?</typ:Longitude>
      </typ:Point>
      <typ:SpeedUpdate>
        <typ:Velocity VelocityUnit=""/>
        <typ:SpeedIncrease>
          <typ:Velocity VelocityUnit=""/>
          <typ:Percentage>?</typ:Percentage>
        </typ:SpeedIncrease>
        <typ:SpeedDecrease>
          <typ:Velocity VelocityUnit="">?</typ:Velocity>
          <typ:Percentage>?</typ:Percentage>
        </typ:SpeedDecrease>
      </typ:SpeedUpdate>
      <typ:Exclude>?</typ:Exclude>
    </typ:PointUpdate>
    <typ:SegmentUpdate>

```

```

    <typ:RoutingSegmentID?></typ:RoutingSegmentID>
    <typ:SpeedUpdate>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:SpeedIncrease>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:Percentage?></typ:Percentage>
    </typ:SpeedIncrease>
    <typ:SpeedDecrease>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:Percentage?></typ:Percentage>
    </typ:SpeedDecrease>
    </typ:SpeedUpdate>
    <typ:RoadType?></typ:RoadType>
    <typ:Exclude?></typ:Exclude>
    </typ:SegmentUpdate>
    <typ:RoadTypeUpdate>
    <typ:RoadType?></typ:RoadType>
    <typ:SpeedUpdate>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:SpeedIncrease>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:Percentage?></typ:Percentage>
    </typ:SpeedIncrease>
    <typ:SpeedDecrease>
    <typ:Velocity VelocityUnit="?"></typ:Velocity>
    <typ:Percentage?></typ:Percentage>
    </typ:SpeedDecrease>
    </typ:SpeedUpdate>
    </typ:RoadTypeUpdate>
    </typ:Update>
  </get:MatrixTransientUpdate>
    </get:RouteMatrixRequest>
    </get:input_port>
  </get:GetTravelCostMatrixRequest>
</soapenv:Body>
</soapenv:Envelope>

```

Response

The output from `GetTravelCostMatrix` contains a number of route sequences, each containing the start and end points for the candidates used in the matrix route as well as the time and distance for each route that has been calculated. The output may contain all routes in the matrix, or only the optimal route between each start/end point, depending on how you configure the `OptimizeBy` option.

If `GetTravelCostMatrix` cannot find a route between a start and end point in the matrix, an error will be logged in the server log but the routes that could be determined will be returned. For example, if you have start point A and end points 1 and 2, and `GetTravelCostMatrix` can find a route from A to 1 but not from A to 2, `GetTravelCostMatrix` will return the route from A to 1 and log an error that the route from A to 2 could not be determined.

GetTravelCostMatrix output is either in list format or object format depending on how you configure the ReturnRouteCostMatrix option. You cannot interact directly with object output.

Table 87: Get Travel Cost Matrix Output

| Response Element | Format | Description | | | | | | | | | | |
|-------------------|------------------------------------|--|-------------|--------------------------|-------------------|--------------------------------|---------------|----------------------------|--------------|------------------------------------|--------------|---------------------------|
| Distance | String | The distance from the start point to the end point. The value is in the units indicated in the DistanceUnits element. | | | | | | | | | | |
| DistanceUnits | String | The unit of measurement used for distance. One of the following: <table border="0" data-bbox="633 735 1250 966"> <tr> <td>Feet</td> <td>The distance is in feet.</td> </tr> <tr> <td>Kilometers</td> <td>The distance is in kilometers.</td> </tr> <tr> <td>Meters</td> <td>The distance is in meters.</td> </tr> <tr> <td>Miles</td> <td>The distance is in miles. Default.</td> </tr> <tr> <td>Yards</td> <td>The distance is in yards.</td> </tr> </table> | Feet | The distance is in feet. | Kilometers | The distance is in kilometers. | Meters | The distance is in meters. | Miles | The distance is in miles. Default. | Yards | The distance is in yards. |
| Feet | The distance is in feet. | | | | | | | | | | | |
| Kilometers | The distance is in kilometers. | | | | | | | | | | | |
| Meters | The distance is in meters. | | | | | | | | | | | |
| Miles | The distance is in miles. Default. | | | | | | | | | | | |
| Yards | The distance is in yards. | | | | | | | | | | | |
| EndPointRef | String | A reference ID that corresponds to the order in which the end points were specified in the input. The first end point specified has a reference ID of 1, the second has an ID of 2, and so on. You must develop your own process for associating the latitude/longitude coordinates in the input with the reference ID returned by Get Travel Cost Matrix. | | | | | | | | | | |
| EndPointID | String | An identification you assigned to the corresponding end point in the ID field of the Input stage. For example, the first end point could have an EndPointID of N, the second end point could have an EndPointID of O, and so on. <p>Note: This field is active only when the Route Cost Matrix Format field is set to "Hierarchy."</p> | | | | | | | | | | |
| StartPointRef | String | A reference ID that corresponds to the order in which the start points were specified in the input. The first start point specified has a reference ID of 1, the second has an ID of 2, and so on. You must develop your own process for associating the latitude/longitude coordinates in the input with the reference ID returned by Get Travel Cost Matrix. | | | | | | | | | | |

| Response Element | Format | Description | | | | | | | | |
|---------------------|----------------------------------|--|--------------|-----------------------|----------------|----------------------------------|----------------|-------------------------|---------------------|------------------------------|
| StartPointID | String | An identification you assigned to the corresponding start point in the ID field of the Input stage. For example, the first start point could have a StartPointID of A, the second start point could have a StartPointID of B, and so on. Note: This field is active only when the Route Cost Matrix Format field is set to "Hierarchy." | | | | | | | | |
| Time | String | The total time from the start point to the end point. The value is in the units indicated in the TimeUnits element. | | | | | | | | |
| TimeUnits | String | The unit of measurement used for time. One of the following: <table border="0"> <tr> <td>Hours</td> <td>The time is in hours.</td> </tr> <tr> <td>Minutes</td> <td>The time is in minutes. Default.</td> </tr> <tr> <td>Seconds</td> <td>The time is in seconds.</td> </tr> <tr> <td>Milliseconds</td> <td>The time is in milliseconds.</td> </tr> </table> | Hours | The time is in hours. | Minutes | The time is in minutes. Default. | Seconds | The time is in seconds. | Milliseconds | The time is in milliseconds. |
| Hours | The time is in hours. | | | | | | | | | |
| Minutes | The time is in minutes. Default. | | | | | | | | | |
| Seconds | The time is in seconds. | | | | | | | | | |
| Milliseconds | The time is in milliseconds. | | | | | | | | | |
| Status | String [1] | Reports the success or failure of the match. <table border="0"> <tr> <td>null</td> <td>Success</td> </tr> <tr> <td>F</td> <td>Failure</td> </tr> </table> | null | Success | F | Failure | | | | |
| null | Success | | | | | | | | | |
| F | Failure | | | | | | | | | |
| Status.Code | String [100] | Reason for failure, if there is one. | | | | | | | | |
| Status.Description | String | A description of failure indicated in Status.Code. | | | | | | | | |

GetTravelDirections (Legacy)

Important: This stage has been deprecated in the 12.1 release. The **Route** stage should be used instead when creating new dataflows.

GetTravelDirections returns routing information for a set of two distinct points or for multiple points. It takes a starting latitude and longitude point and an ending latitude and longitude point as input and returns the route that is either fastest or shortest, depending on how you configure the stage.

Each country has its own database, named in this format: Spectrum Spatial Routing - <Country>. Each database also has its own country code. For example, the name of the Austrian database is "Spectrum Spatial Routing - Austria," and the Austrian batch country code is "A1T." Each database requires a separate license.

Note: Get Travel Directions is only available as a SOAP web service. Get Travel Directions is not available through REST. It is also not available through the Java, C++, C, .NET, or COM APIs.

GetTravelDirections is part of Spectrum Spatial Routing.

Resource URL

```
http://server:port/soap/GetTravelDirections
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://<server>:<port>/spectrum/services/GetTravelDirections"
xmlns:typ="http://www.gl.com/services/erm/types">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetTravelDirectionsRequest>
      <get:input_port>
        <get:PointToPointRequest>
          <get:RoutePoints>
            <get:RoutePoint>
              <get:Latitude>33.751748</get:Latitude>

              <get:Longitude>-84.364014</get:Longitude>
            </get:RoutePoint>
            <get:RoutePoint>
              <get:Latitude>33.664925</get:Latitude>
              <get:Longitude>-80.90332</get:Longitude>
            </get:RoutePoint>
          </get:RoutePoints>
        </get:PointToPointRequest>
      </get:input_port>
    </get:GetTravelDirectionsRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: Some of the directions have been removed from this example to shorten it.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns6:GetTravelDirectionsResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.g1.com/services/erm/types"
xmlns:ns4="http://www.g1.com/services/GetTravelDirections"
xmlns:ns5="http://www.mapinfo.com/midev/service/geometries/v1"
xmlns:ns6="http://<server>:<port>/spectrum/services/GetTravelDirections">

      <ns6:output_port>
        <ns6:PointToPointResponse>
          <ns6:Time>215.82</ns6:Time>
          <ns6:TimeUnits>Minutes</ns6:TimeUnits>
          <ns6:Distance>218.441</ns6:Distance>
          <ns6:DistanceUnits>Miles</ns6:DistanceUnits>
          <ns6:Format>Normal</ns6:Format>
          <ns6:Language>en</ns6:Language>
          <ns6:RouteDirections>
            <ns6:RouteDirection>
              <ns6:Instruction/>
              <ns6:Time>0.03</ns6:Time>
              <ns6:TimeUnits>Minutes</ns6:TimeUnits>
              <ns6:Distance>0.013</ns6:Distance>
              <ns6:DistanceUnits>Miles</ns6:DistanceUnits>
            </ns6:RouteDirection>
            <ns6:RouteDirection>
              <ns6:Instruction>Turn left on Short St SE and travel
South 0.10 mi (0.3 min).</ns6:Instruction>
              <ns6:Time>0.28</ns6:Time>
              <ns6:TimeUnits>Minutes</ns6:TimeUnits>
              <ns6:Distance>0.099</ns6:Distance>
              <ns6:DistanceUnits>Miles</ns6:DistanceUnits>
            </ns6:RouteDirection>
            ...
            <ns6:RouteDirection>
              <ns6:Instruction>Turn left on Un-named street and
travel East 0.11 mi (0.2 min).</ns6:Instruction>
              <ns6:Time>0.2</ns6:Time>
              <ns6:TimeUnits>Minutes</ns6:TimeUnits>
              <ns6:Distance>0.105</ns6:Distance>
              <ns6:DistanceUnits>Miles</ns6:DistanceUnits>
            </ns6:RouteDirection>
            <ns6:RouteDirection>
              <ns6:Instruction>Turn right to reach your destination
to the East.</ns6:Instruction>
              <ns6:Time>0.33</ns6:Time>
              <ns6:TimeUnits>Minutes</ns6:TimeUnits>
              <ns6:Distance>0.167</ns6:Distance>
              <ns6:DistanceUnits>Miles</ns6:DistanceUnits>
            </ns6:RouteDirection>
          </ns6:RouteDirections>
        </ns6:PointToPointResponse>
      </ns6:output_port>
    </ns6:GetTravelDirectionsResponse>
  </soap:Body>
</soap:Envelope>
```

```

        </ns6:RouteDirections>
        <ns6:user_fields/>
    </ns6:PointToPointResponse>
</ns6:output_port>
</ns6:GetTravelDirectionsResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

GetTravelDirections takes route points, which include starting and ending latitudes and longitudes as input. The following table provides information on the format and layout of the input.

Note: If you intend to interact with the GetTravelDirections service, note that it is only available as a web service. The Get Travel Directions service is not available through the Java, C++, C, .NET, or COM APIs.

Table 88: GetTravel Directions Input Data

| Parameter | Format | Description |
|-------------|--------|--|
| RoutePoints | List | A schema containing the latitude string and the longitude string. For information on creating a process list, see the <i>Spectrum Technology Platform Dataflow Designer's Guide</i> . |

| Parameter | Format | Description |
|-----------|--------|--|
| Language | String | <p>Specifies the language in which GetTravelDirections should return directions.</p> <p>One of the following:</p> <p>sq Return directions in Albanian.</p> <p>zh_CN Return directions in Chinese.</p> <p>zh_TW Return directions in Chinese (Taiwan).</p> <p>hr Return directions in Croatian.</p> <p>cs Return directions in Czech.</p> <p>da Return directions in Danish.</p> <p>nl Return directions in Dutch.</p> <p>en Return directions in English. Default</p> <p>en-US Return directions in American English.</p> <p>et Return directions in Estonian.</p> <p>fi Return directions in Finnish.</p> <p>fr Return directions in French.</p> <p>de Return directions in German.</p> <p>hu Return directions in Hungarian.</p> <p>it Return directions in Italian.</p> <p>ja Return directions in Japanese.</p> <p>lv Return directions in Latvian.</p> <p>lt Return directions in Lithuanian.</p> <p>no Return directions in Norwegian.</p> <p>pt Return directions in Portuguese.</p> <p>ro Return directions in Romanian.</p> <p>sk Return directions in Slovak.</p> <p>sl Return directions in Slovenian.</p> <p>es Return directions in Spanish.</p> <p>sv Return directions in Swedish.</p> <p>ru Return directions in Russian.</p> <p>tr Return directions in Turkish.</p> <p>Note: An entry in this input field will override an entry in the Default Language option.</p> |

| Parameter | Format | Description |
|--------------------------------|---------|--|
| TravelDirectionTransientUpdate | List | <p>A schema containing the update types for the transient update. Transient updates are updates made on a request that only apply to that particular request. Transient updates are similar to Persistent updates, except that Transient updates are only for a particular request and Persistent updates are for all the requests. You have the ability to set a speed for a point, a segment id, or a road class, as well as the ability to update the road class for a segment (specified by the segment id).</p> <p>For transient update options and example, see GetTravelDirections_TransientOptions.dita.</p> |
| TollRoad | Boolean | <p>This feature specifies whether you want a route with or without a toll road. This is a Boolean type parameter. The default value is False. If you set the value of <code>TollRoad</code> to True, the response contains a route without any toll roads. If the value of <code>TollRoad</code> is set to False, the route includes toll roads.</p> |

Parameters for Options

Routing

The following table lists the configuration options for GetTravelDirections.

Table 89: Get Travel Directions Configuration Options

| Parameter | Description | | | | |
|---------------------|--|-------------|--|-----------------|---|
| DataSetResourceName | <p>The name of the database that contains the data to use in the search process. Use a valid routing database resource name defined in the Resources section of the Management Console. For more information, see the <i>Spectrum Technology Platform Spatial Guide</i>.</p> | | | | |
| OptimizeBy | <p>Specifies whether GetTravelDirections should find the shortest distance or the shortest time.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Time</td> <td>Optimize by shortest time traveled. Default.</td> </tr> <tr> <td>Distance</td> <td>Optimize by shortest distance traveled.</td> </tr> </table> | Time | Optimize by shortest time traveled. Default. | Distance | Optimize by shortest distance traveled. |
| Time | Optimize by shortest time traveled. Default. | | | | |
| Distance | Optimize by shortest distance traveled. | | | | |

| Parameter | Description | | | | | | | | | | |
|------------------------------|---|----------------|--|-----------------------|---|------------------------------|----------------------------|----------------|------------------------------------|-----------------------|----------------------------|
| CoordinateSystem | <p>The coordinate system of the input coordinates.</p> <p>For more information on EPSG codes, see www.spatialreference.org. To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <a href="http://<server>:<port>/Spatial/GeometryService/DemoPage.html">http://<server>:<port>/Spatial/GeometryService/DemoPage.html.</p> | | | | | | | | | | |
| CoordinateFormat | <p>Specifies the format of latitude/longitude coordinates in the input.</p> <p>Note: Use this option only if you specify a Latitude/Longitude coordinate system. If the coordinate system is not a Latitude/Longitude coordinate system, set the coordinate format to Decimal.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Decimal</td> <td>(90.000000, 180.000000). Default.</td> </tr> <tr> <td>DecimalAssumed</td> <td>(90000000, 180000000).</td> </tr> <tr> <td>DegreesMinutesSeconds</td> <td>(90 00 00N, 180 00 00W)</td> </tr> <tr> <td>PreZero</td> <td>(090000000N, 180000000W)</td> </tr> <tr> <td>PreZeroDecimal</td> <td>(090.000000N, 180.000000W)</td> </tr> </table> | Decimal | (90.000000, 180.000000). Default. | DecimalAssumed | (90000000, 180000000). | DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | PreZero | (090000000N, 180000000W) | PreZeroDecimal | (090.000000N, 180.000000W) |
| Decimal | (90.000000, 180.000000). Default. | | | | | | | | | | |
| DecimalAssumed | (90000000, 180000000). | | | | | | | | | | |
| DegreesMinutesSeconds | (90 00 00N, 180 00 00W) | | | | | | | | | | |
| PreZero | (090000000N, 180000000W) | | | | | | | | | | |
| PreZeroDecimal | (090.000000N, 180.000000W) | | | | | | | | | | |
| DirectionsStyle | <p>Specifies how directions should be returned.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Normal</td> <td>Return directions in a normal format. Default.</td> </tr> <tr> <td>Terse</td> <td>Return directions in a terse format. Terse directions are more suitable for wireless devices.</td> </tr> </table> | Normal | Return directions in a normal format. Default. | Terse | Return directions in a terse format. Terse directions are more suitable for wireless devices. | | | | | | |
| Normal | Return directions in a normal format. Default. | | | | | | | | | | |
| Terse | Return directions in a terse format. Terse directions are more suitable for wireless devices. | | | | | | | | | | |
| DistanceUnits | <p>Specifies how GetTravelDirections should return distance values.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Feet</td> <td>Return distance in feet.</td> </tr> <tr> <td>Kilometers</td> <td>Return distance in kilometers.</td> </tr> <tr> <td>Meters</td> <td>Return distance in meters.</td> </tr> <tr> <td>Miles</td> <td>Return distance in miles. Default.</td> </tr> <tr> <td>Yards</td> <td>Return distance in yards.</td> </tr> </table> | Feet | Return distance in feet. | Kilometers | Return distance in kilometers. | Meters | Return distance in meters. | Miles | Return distance in miles. Default. | Yards | Return distance in yards. |
| Feet | Return distance in feet. | | | | | | | | | | |
| Kilometers | Return distance in kilometers. | | | | | | | | | | |
| Meters | Return distance in meters. | | | | | | | | | | |
| Miles | Return distance in miles. Default. | | | | | | | | | | |
| Yards | Return distance in yards. | | | | | | | | | | |

| Parameter | Description | | | | | | | | | | |
|---------------------------|---|--------------|---|----------------|--|----------------|--|---------------------|--|--------------|---|
| TimeUnits | <p>Specifies how GetTravelDirections should return time.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Hours</td> <td>Return time in hours.</td> </tr> <tr> <td>Minutes</td> <td>Return time in minutes. Default.</td> </tr> <tr> <td>Seconds</td> <td>Return time in seconds.</td> </tr> <tr> <td>Milliseconds</td> <td>Return time in milliseconds.</td> </tr> </table> | Hours | Return time in hours. | Minutes | Return time in minutes. Default. | Seconds | Return time in seconds. | Milliseconds | Return time in milliseconds. | | |
| Hours | Return time in hours. | | | | | | | | | | |
| Minutes | Return time in minutes. Default. | | | | | | | | | | |
| Seconds | Return time in seconds. | | | | | | | | | | |
| Milliseconds | Return time in milliseconds. | | | | | | | | | | |
| FocusOfRoute | <p>Specifies the focus of the route. A focused route is a subset of the whole route that concentrates on either the beginning or end of the route. A route focused at the start will route the user from their origin to (and including) the first major highway. A route focused at the end will route the user from the last major highway in the route (including that highway) to the destination. If there are no major highways in the route, the focused route will be the same as an unfocused route.</p> <p>One of the following:</p> <table border="0"> <tr> <td>Start</td> <td>Return just the start of the route.</td> </tr> <tr> <td>End</td> <td>Return just the end of the route.</td> </tr> <tr> <td>None</td> <td>Return the whole route. Default.</td> </tr> </table> | Start | Return just the start of the route. | End | Return just the end of the route. | None | Return the whole route. Default. | | | | |
| Start | Return just the start of the route. | | | | | | | | | | |
| End | Return just the end of the route. | | | | | | | | | | |
| None | Return the whole route. Default. | | | | | | | | | | |
| HistoricTrafficTimeBucket | <p>Specifies whether the routing calculation uses the historic traffic speeds. These speeds are based on different time-of-day buckets. The data must have historic traffic speeds included in order to use this feature. The data for each country/region has the same bucket definitions, where the speeds for these bucket values may vary. The options are:</p> <table border="0"> <tr> <td>None</td> <td>The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used.</td> </tr> <tr> <td>AMPeak</td> <td>Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day.</td> </tr> <tr> <td>PMPeak</td> <td>Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day.</td> </tr> <tr> <td>OffPeak</td> <td>Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day.</td> </tr> <tr> <td>Night</td> <td>Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day.</td> </tr> </table> | None | The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used. | AMPeak | Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day. | PMPeak | Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day. | OffPeak | Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day. | Night | Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day. |
| None | The default value. Historic traffic data is not used in the calculation. Instead an averaged speed value is used. | | | | | | | | | | |
| AMPeak | Calculate routes with the peak AM speeds. The AMPeak time bucket is from 07:00 to 10:00hr time of day. | | | | | | | | | | |
| PMPeak | Calculate routes with the peak PM speeds. The PMPeak time bucket is from 16:00 to 19:00hr time of day. | | | | | | | | | | |
| OffPeak | Calculate routes with the off peak (daytime) speeds. The OffPeak time bucket is from 10:00 to 16:00hr time of day. | | | | | | | | | | |
| Night | Calculate routes with the nighttime speeds. The Night time bucket is from 22:00 to 04:00hr time of day. | | | | | | | | | | |

| Parameter | Description |
|----------------------------|--|
| OptimizeIntermediatePoints | <p>Specifies whether GetTravelDirections re-orders the intermediate points in an optimal manner during route computation.</p> <p>Y Yes, optimize intermediate points. Directions optimally orders the intermediate points. Default.</p> <p>N No, do not optimize intermediate points. Directions preserves the specified order of the points.</p> |
| localRoadsLoadFactor | <p>Specifies the number of local roads that can be loaded into memory during route or matrix calculation. The number of roads that can load is directly proportional to the value selected in the parameter where 1 is the minimum and 3 is the maximum value. Valid values can be 1,2, or 3. The default is 1. See Local Roads Load Factor for a detailed description and impact of the parameter on routing or matrix calculation.</p> <p>Note: The parameter does not accept decimal values.</p> |

Directions

Table 90: Get Travel Directions Direction Options

| Parameter | Description |
|-----------------------|---|
| ShowDistance | <p>Specifies whether to return the distance for the route.</p> <p>Y Yes, return the distance for the route. Default.</p> <p>N No, do not return the distance for the route.</p> |
| ReturnRouteDirections | <p>Specifies whether to return turn-by-turn directions for the route. This option is enabled by default.</p> <p>Y Yes, return textual turn-by-turn directions. Default.</p> <p>N No, do not return textual turn-by-turn directions.</p> |
| ReturnSegmentGeometry | <p>Specifies whether to return a set of latitude/longitude points that represent a geometry for a segment within a route. The segment geometry is used to create the route geometry. The output of this field is shown within the RouteDirections output field. For more information about route geometry, see What is Route Geometry? on page 784.</p> |

| Parameter | Description | | | | | | |
|---------------------|---|------------|--|------------|--|-------------|--|
| ReturnRouteGeometry | <p>Specifies whether to return a set of latitude/longitude points that represent a route geometry. The route geometry can be used to create a route map and to perform analysis on the route.</p> <p>One of the following:</p> <table><tbody><tr><td>All</td><td>Return all points in the route geometry.</td></tr><tr><td>End</td><td>Return just the end points of each route segment geometry.</td></tr><tr><td>None</td><td>Do not return route geometry. Default.</td></tr></tbody></table> | All | Return all points in the route geometry. | End | Return just the end points of each route segment geometry. | None | Do not return route geometry. Default. |
| All | Return all points in the route geometry. | | | | | | |
| End | Return just the end points of each route segment geometry. | | | | | | |
| None | Do not return route geometry. Default. | | | | | | |

| Parameter | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------|--|-----------|--|--------------|---|--------------|--|-----------|--------------------------------|-----------|-----------------------------|-----------|------------------------------|-----------|-----------------------------|-----------|---------------------------------------|--------------|--|-----------|--------------------------------|-----------|-------------------------------|-----------|------------------------------|-----------|------------------------------|-----------|---------------------------------|-----------|-------------------------------|-----------|--------------------------------|-----------|-------------------------------|-----------|----------------------------------|-----------|---------------------------------|-----------|----------------------------------|-----------|--------------------------------|-----------|------------------------------|-----------|---------------------------------|-----------|-------------------------------|-----------|-------------------------------|-----------|-------------------------------|-----------|-------------------------------|
| DefaultLanguage | <p>Specifies the language in which GetTravelDirections should return directions.</p> <p>One of the following:</p> <table border="0"> <tr> <td>sq</td> <td>Return directions in Albanian.</td> </tr> <tr> <td>zh_CN</td> <td>Return directions in Chinese.</td> </tr> <tr> <td>zh_TW</td> <td>Return directions in Chinese (Taiwan).</td> </tr> <tr> <td>hr</td> <td>Return directions in Croatian.</td> </tr> <tr> <td>cs</td> <td>Return directions in Czech.</td> </tr> <tr> <td>da</td> <td>Return directions in Danish.</td> </tr> <tr> <td>nl</td> <td>Return directions in Dutch.</td> </tr> <tr> <td>en</td> <td>Return directions in English. Default</td> </tr> <tr> <td>en-US</td> <td>Return directions in American English.</td> </tr> <tr> <td>et</td> <td>Return directions in Estonian.</td> </tr> <tr> <td>fi</td> <td>Return directions in Finnish.</td> </tr> <tr> <td>fr</td> <td>Return directions in French.</td> </tr> <tr> <td>de</td> <td>Return directions in German.</td> </tr> <tr> <td>hu</td> <td>Return directions in Hungarian.</td> </tr> <tr> <td>it</td> <td>Return directions in Italian.</td> </tr> <tr> <td>ja</td> <td>Return directions in Japanese.</td> </tr> <tr> <td>lv</td> <td>Return directions in Latvian.</td> </tr> <tr> <td>lt</td> <td>Return directions in Lithuanian.</td> </tr> <tr> <td>no</td> <td>Return directions in Norwegian.</td> </tr> <tr> <td>pt</td> <td>Return directions in Portuguese.</td> </tr> <tr> <td>ro</td> <td>Return directions in Romanian.</td> </tr> <tr> <td>sk</td> <td>Return directions in Slovak.</td> </tr> <tr> <td>sl</td> <td>Return directions in Slovenian.</td> </tr> <tr> <td>es</td> <td>Return directions in Spanish.</td> </tr> <tr> <td>sv</td> <td>Return directions in Swedish.</td> </tr> <tr> <td>ru</td> <td>Return directions in Russian.</td> </tr> <tr> <td>tr</td> <td>Return directions in Turkish.</td> </tr> </table> | sq | Return directions in Albanian. | zh_CN | Return directions in Chinese. | zh_TW | Return directions in Chinese (Taiwan). | hr | Return directions in Croatian. | cs | Return directions in Czech. | da | Return directions in Danish. | nl | Return directions in Dutch. | en | Return directions in English. Default | en-US | Return directions in American English. | et | Return directions in Estonian. | fi | Return directions in Finnish. | fr | Return directions in French. | de | Return directions in German. | hu | Return directions in Hungarian. | it | Return directions in Italian. | ja | Return directions in Japanese. | lv | Return directions in Latvian. | lt | Return directions in Lithuanian. | no | Return directions in Norwegian. | pt | Return directions in Portuguese. | ro | Return directions in Romanian. | sk | Return directions in Slovak. | sl | Return directions in Slovenian. | es | Return directions in Spanish. | sv | Return directions in Swedish. | ru | Return directions in Russian. | tr | Return directions in Turkish. |
| sq | Return directions in Albanian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| zh_CN | Return directions in Chinese. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| zh_TW | Return directions in Chinese (Taiwan). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| hr | Return directions in Croatian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| cs | Return directions in Czech. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| da | Return directions in Danish. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| nl | Return directions in Dutch. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| en | Return directions in English. Default | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| en-US | Return directions in American English. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| et | Return directions in Estonian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fi | Return directions in Finnish. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fr | Return directions in French. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| de | Return directions in German. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| hu | Return directions in Hungarian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| it | Return directions in Italian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ja | Return directions in Japanese. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| lv | Return directions in Latvian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| lt | Return directions in Lithuanian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| no | Return directions in Norwegian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| pt | Return directions in Portuguese. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ro | Return directions in Romanian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| sk | Return directions in Slovak. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| sl | Return directions in Slovenian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| es | Return directions in Spanish. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| sv | Return directions in Swedish. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ru | Return directions in Russian. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| tr | Return directions in Turkish. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ShowTime | <p>Specifies whether GetTravelDirections should return the time it takes to follow a direction within the route.</p> <table border="0"> <tr> <td>Y</td> <td>Yes, return the time for the route. Default.</td> </tr> <tr> <td>N</td> <td>No, do not return the time for the route.</td> </tr> </table> | Y | Yes, return the time for the route. Default. | N | No, do not return the time for the route. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Y | Yes, return the time for the route. Default. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| N | No, do not return the time for the route. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| Parameter | Description |
|---------------------|--|
| ShowPrimaryNameOnly | Specifies whether GetTravelDirections should return all names for a given street in the directions or if it should return just the primary name for a street. Y Yes, return just the primary name. N Return all names for a street. Default. |

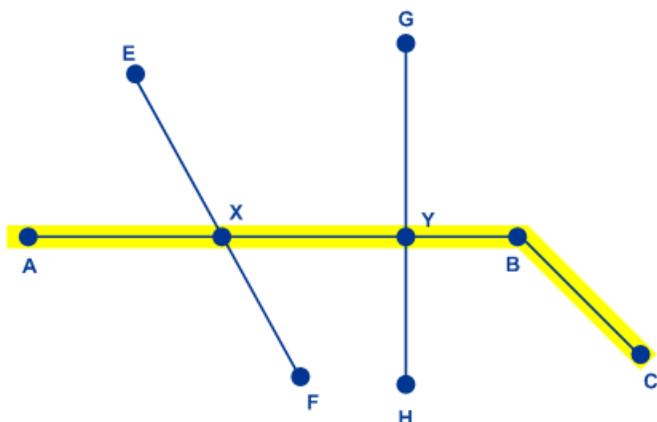
What is Route Geometry?

A route geometry is a series of latitude/longitude points that form a route. A route geometry can be as simple as a single point, such as a route that starts and ends on the same street segment:



Since the starting point is always known the route geometry in this very simple example could be just the end point. So if A is the starting point, the route geometry could be the latitude/longitude of point B.

In a more complex route geometry involving multiple route segments there can be several points in the route geometry. Consider the route highlighted below which starts at point A and ends at point C, traveling through intersections X, Y, and B:



In this route, the full route geometry would consist of the latitude/longitude of points A, X, Y, B, and C. Note that you can control which points to actually return and may choose to include all points in the route geometry or just the end points of each route segment. In the above example, the end points are B and C, since A to B is one route segment and B to C is another.

Travel

This set of preferences allows you to set the desirability for each road type. For instance, you can request that the server attempt to avoid all of the major road types.

Table 91: Travel Preferences Options

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-----------------|--|
| RoadType_<type> | |
|-----------------|--|

| Parameter | Description |
|-----------|--|
| | <p data-bbox="548 369 1424 396">Specifies the priority to give to different types of roads when determining the route.</p> <ul style="list-style-type: none"> <li data-bbox="548 422 695 449">• access way <li data-bbox="548 453 678 480">• back road <li data-bbox="548 485 678 512">• connector <li data-bbox="548 516 623 543">• ferry <li data-bbox="548 548 659 575">• footpath <li data-bbox="548 579 862 606">• limited access dense urban <li data-bbox="548 611 776 638">• limited access rural <li data-bbox="548 642 829 669">• limited access suburban <li data-bbox="548 674 792 701">• limited access urban <li data-bbox="548 705 813 732">• local road dense urban <li data-bbox="548 737 732 764">• local road rural <li data-bbox="548 768 781 795">• local road suburban <li data-bbox="548 800 743 827">• local road urban <li data-bbox="548 831 883 858">• major local road dense urban <li data-bbox="548 863 797 890">• major local road rural <li data-bbox="548 894 850 921">• major local road suburban <li data-bbox="548 926 813 953">• major local road urban <li data-bbox="548 957 829 984">• major road dense urban <li data-bbox="548 989 743 1016">• major road rural <li data-bbox="548 1020 797 1047">• major road suburban <li data-bbox="548 1052 753 1079">• major road urban <li data-bbox="548 1083 883 1110">• minor local road dense Urban <li data-bbox="548 1115 797 1142">• minor local road rural <li data-bbox="548 1146 850 1173">• minor local road suburban <li data-bbox="548 1178 813 1205">• minor local road urban <li data-bbox="548 1209 846 1236">• normal road dense urban <li data-bbox="548 1241 753 1268">• normal road rural <li data-bbox="548 1272 753 1299">• normal road rural <li data-bbox="548 1304 769 1331">• normal road urban <li data-bbox="548 1335 883 1362">• primary highway dense urban <li data-bbox="548 1367 797 1394">• primary highway rural <li data-bbox="548 1398 850 1425">• primary highway suburban <li data-bbox="548 1430 813 1457">• primary highway urban <li data-bbox="548 1461 769 1488">• ramp dense urban <li data-bbox="548 1493 781 1520">• ramp limited access <li data-bbox="548 1524 748 1551">• ramp major road <li data-bbox="548 1556 808 1583">• ramp primary highway <li data-bbox="548 1587 683 1614">• ramp rural <li data-bbox="548 1619 841 1646">• ramp secondary highway <li data-bbox="548 1650 699 1677">• ramp urban <li data-bbox="548 1682 737 1709">• ramp suburban <li data-bbox="548 1713 915 1740">• secondary highway dense urban <li data-bbox="548 1745 829 1772">• secondary highway rural |

| Parameter | Description |
|-----------|---|
| | <ul style="list-style-type: none">• secondary highway suburban• secondary highway urban <p>For each road type you can specify one of the following:</p> <p>Avoid Exclude the road type from routes if possible.</p> <p>Note: It is not always possible to exclude a road type from the travel directions. Depending on the situation, the alternative to an avoided road type may be so poor that the software will choose a route that uses an avoided road type. Also, if the starting or ending point lies along a segment whose road type has been avoided, the software will still use that segment.</p> <p>High Prefer the road type over other road types.</p> <p>Low Prefer other road types over this road type.</p> <p>Medium Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium.</p> |

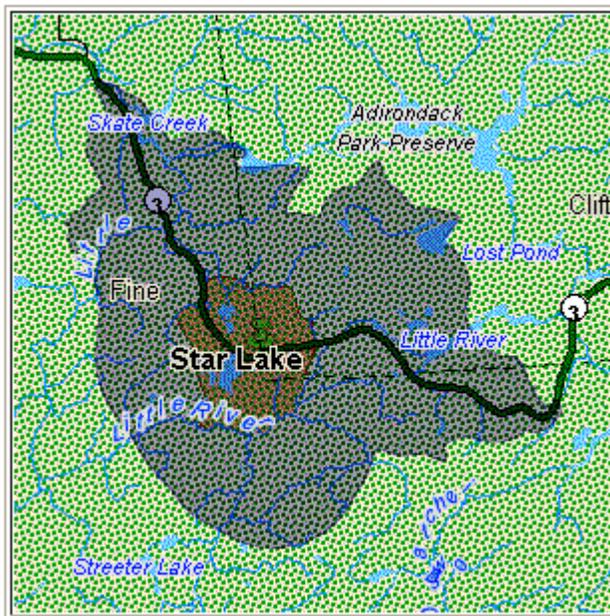
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|------------|--|
| MajorRoads | |
|------------|--|

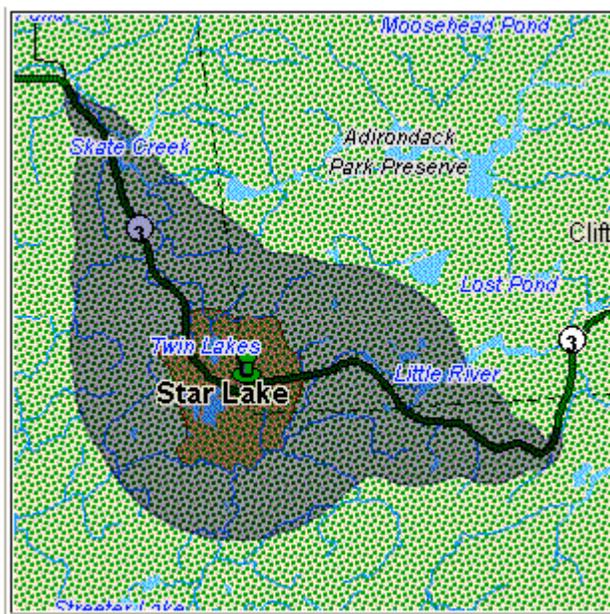
| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Specifies whether to include all roads in the calculation or just major roads. If you choose to include only major roads, performance will improve but accuracy may decrease.

This map represents a travel boundary with travel allowed on all roads:



This map represents a travel boundary with travel restricted to major roads only:



| Parameter | Description |
|--------------------------------|---|
| <hr/> One of the following: | |
| Y | Include only major roads in the calculation. Default. |
| N | Include all roads in the calculation. |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

Avoid

Toll Roads

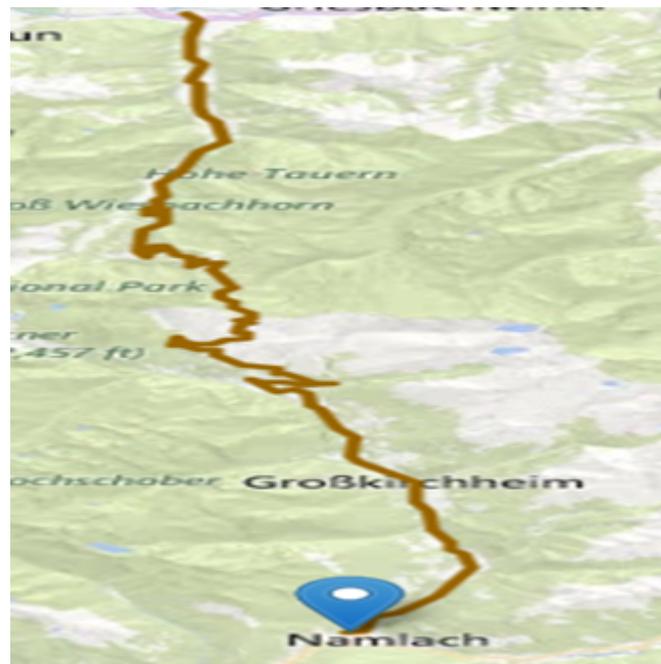
Parameter

Description

Specifies whether you want a route with or without a toll road. The stage GetTravelCostMatrix GetTravelDirections contains the "avoid Toll Roads" feature. There is a check box labeled "Toll Roads" on the UI. You can check it to avoid toll roads. You can also add or expose this parameter from input value as "TollRoad". The input value can contain Boolean values where False is the default value.

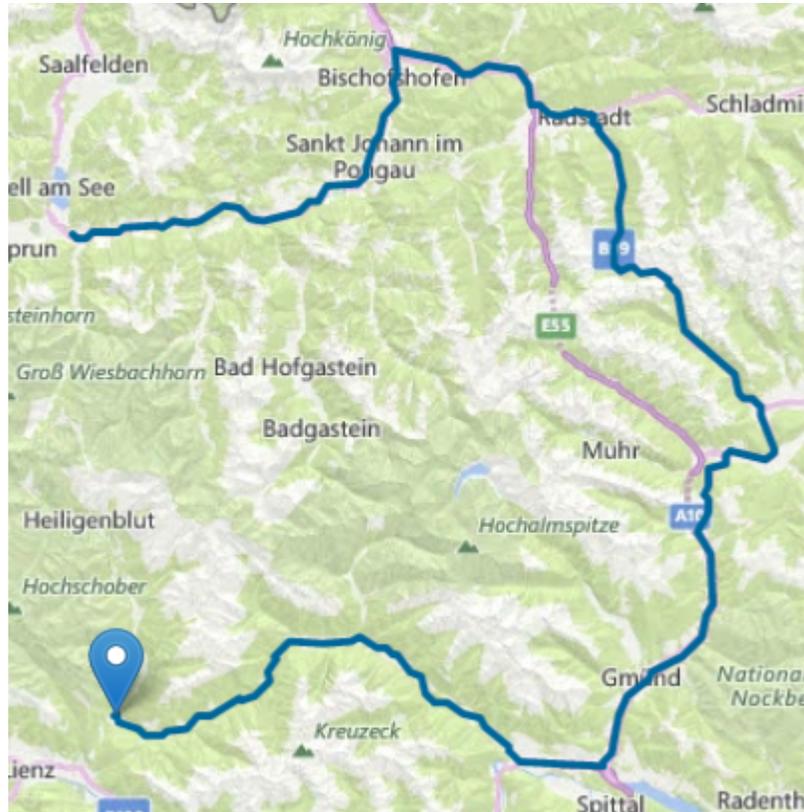
For Example:

Following route contains toll road information, which is mentioned in the following image:



Now, for the same points, if you check the toll Road or set the "TollRoad" parameter as True, the response contains a route without any toll roads. See the following response:

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|



Transient Options

This set of preferences allows you to set transient updates for each request. For instance, you can request that the server attempt to avoid all of the major road types. Each request can contain one or more updates.

Note: The transient update functionality is only available through the SOAP API, and is not available through the Management Console or Enterprise Designer.

Table 92: Transient Update Options

| Parameter | Description |
|-------------|---|
| PointUpdate | <p>Point updates are changes applied to a corresponding point (Latitude, Longitude). For a particular point, you can: exclude the point, set the speed of the point or change (increase or decrease) the speed of the point by a value or percentage. You must specify a <code>Point</code> which consists of a <code>Latitude</code> and <code>Longitude</code>, then specify one of the following:</p> <p>Velocity This is a speed update where you define the new speed of the point by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedIncrease This is a speed update where you define an increase in the speed of the point by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedDecrease This is a speed update where you define a decrease in the speed of the point by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>Exclude This is a sting value to exclude the specified point from the route calculation. To exclude a point you need to specify the point and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|---------------|--|
| SegmentUpdate | |
|---------------|--|

| Parameter | Description |
|----------------------|--|
| | <p>Segment updates are changes applied to a corresponding segment ID (Latitude, Longitude). For a particular segment, you can: exclude the segment, set the speed of the segment, change (increase or decrease) the speed of the segment by a value or percentage, or change the road type of the segment. You must specify a valid <code>RoutingSegmentID</code>, then specify one of the following:</p> |
| Velocity | <p>This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| SpeedIncrease | <p>This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| SpeedDecrease | <p>This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |
| RoadType | <p>This is a string value to change the value of the road type for the segment for the route calculation.</p> <p>The <code>RoadType</code> can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural |

| Parameter | Description |
|----------------|---|
| | <ul style="list-style-type: none"> • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban • ramp suburban • secondary highway dense urban • secondary highway rural • secondary highway suburban • secondary highway urban |
| Exclude | <p>This is a sting value to exclude the specified segment from the route calculation. To exclude a segment you need to specify the segment ID and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no).</p> |

| Parameter | Description |
|----------------|---|
| RoadTypeUpdate | <p>Road type updates are changes applied to a corresponding road type. For a particular road type, you can: set the speed of the roadtype, or change (increase or decrease) the speed of the road type by a value or percentage. You must specify a <code>RoadType</code> to update (see the above road types in the segment update), then specify one of the following:</p> <p>Velocity This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedIncrease This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> <p>SpeedDecrease This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute).</p> |

GetTravelDirections Transient Update SOAP Example

The following shows a standard GetTravelDirections SOAP request with all of the transient update options available (not a working example, this is used to show all the syntax). You can have multiple Update definitions within a TravelDirectionTransientUpdate. You can only have a single update type (PointUpdate, SegmentUpdate, or RoadTypeUpdate) within an Update. You can also only have a single update within one of the update types (PointUpdate, SegmentUpdate, or RoadTypeUpdate).

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://<server>:<port>/spectrum/services/GetTravelDirections"
xmlns:typ="http://www.g1.com/services/erm/types">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetTravelDirectionsRequest>
      <get:input_port>
        <get:PointToPointRequest>
          <get:RoutePoints>
            <get:RoutePoint>
              <get:Latitude>33.751748</get:Latitude>
```

```

        <get:Longitude>-84.364014</get:Longitude>
    </get:RoutePoint>
    <get:RoutePoint>
        <get:Latitude>33.664925</get:Latitude>
        <get:Longitude>-80.90332</get:Longitude>
    </get:RoutePoint>
</get:RoutePoints>
<get:Language>en-US</get:Language>
<get:TravelDirectionTransientUpdate>
    <typ:Update>
        <typ:PointUpdate>
            <typ:Point>
                <typ:Latitude>?</typ:Latitude>
                <typ:Longitude>?</typ:Longitude>
            </typ:Point>
            <typ:SpeedUpdate>
                <typ:Velocity VelocityUnit=""/>
                <typ:SpeedIncrease>
                    <typ:Velocity VelocityUnit=""/>
                    <typ:Percentage>?</typ:Percentage>
                </typ:SpeedIncrease>
                <typ:SpeedDecrease>
                    <typ:Velocity
VelocityUnit="">?</typ:Velocity>
                    <typ:Percentage>?</typ:Percentage>
                </typ:SpeedDecrease>
            </typ:SpeedUpdate>
            <typ:Exclude>?</typ:Exclude>
        </typ:PointUpdate>
    <typ:SegmentUpdate>

    <typ:RoutingSegmentID>?</typ:RoutingSegmentID>
        <typ:SpeedUpdate>
            <typ:Velocity
VelocityUnit="">?</typ:Velocity>
            <typ:SpeedIncrease>
                <typ:Velocity
VelocityUnit="">?</typ:Velocity>
            <typ:Percentage>?</typ:Percentage>
            </typ:SpeedIncrease>
            <typ:SpeedDecrease>
                <typ:Velocity
VelocityUnit="">?</typ:Velocity>
            <typ:Percentage>?</typ:Percentage>
            </typ:SpeedDecrease>
        </typ:SpeedUpdate>
        <typ:RoadType>?</typ:RoadType>
        <typ:Exclude>?</typ:Exclude>
    </typ:SegmentUpdate>
    <typ:RoadTypeUpdate>
        <typ:RoadType>?</typ:RoadType>
        <typ:SpeedUpdate>

```


| Response Element | Format | Description |
|--------------------|-----------------|--|
| RouteGeometry | Geometry object | A geometry object containing the coordinates of each point in the route. For more information, see What is Route Geometry? on page 784. |
| Status | String | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | String | Reason for failure, if there is one. One of the following: <ul style="list-style-type: none"> InsufficientInputData(missing lat/lon) MalformedInputData(wrong input format) InputOutOfRange (input is out of range) EngineError (engine generated error) |
| Status.Description | String | Description of failure indicated in Status.Code. |
| Time | String | Total time it takes to follow the route in minutes. |
| TimeUnits | String | The unit of measurement for time. |

GetRouteData

GetRouteData returns routing segment information for a point or segment ID. When you specify a point, the closest route segments are returned. When you specify a segment ID, the route segment for that segment ID is returned.

Note: Get Route Data is only available as a service (Management Console and SOAP web service). Get Route Data is not available through a stage or REST API. It is also not available through the Java, C++, C, .NET, or COM APIs.

GetRouteData is part of Spectrum Spatial.

Resource URL

```
http://server:port/soap/GetRouteData
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.g1.com/services/GetRouteData">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetRouteDataRequest>
      <get:options>
        <get:DataSetResourceName>US</get:DataSetResourceName>
        <get:CoordinateSystem>epsg:4326</get:CoordinateSystem>
      </get:options>
      <get:rows>
        <get:row>
          <get:RoutingData>
            <get:RouteDataPoint>
              <get:Longitude>-74.843</get:Longitude>
              <get:Lattitude>40.0077</get:Lattitude>
            </get:RouteDataPoint>
          </get:RoutingData>
        </get:row>
      </get:rows>
    </get:GetRouteDataRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: Some of the segment information has been removed from this example to shorten it.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <GetRouteDataResponse
xmlns="http://www.g1.com/services/GetRouteData">
      <rows>
        <row>
          <Segments>
            <SegmentDetails>
              <Segment>

<RoutingSegmentID>b81740d3:4b3526</RoutingSegmentID>
              <SegmentData>
                <PrimaryName>New Jersey Tpke S</PrimaryName>
```

```

    <PrimaryNameLanguage>en</PrimaryNameLanguage>

    <AlternateNameList>
      <AlternateName>
        <Name>New Jersey Tpke S</Name>
        <Language>en</Language>
      </AlternateName>
    </AlternateNameList>
    <SegmentLength>8.397</SegmentLength>
    <SegmentLengthUnit>Miles</SegmentLengthUnit>

    <TimeTaken>7.866666666666666</TimeTaken>
    <TimeUnit>Minutes</TimeUnit>
    <TurnAngle>0.0</TurnAngle>
    <TurnAngleUnit>degree</TurnAngleUnit>
    <CompassDirection/>

<speedOfTravel>64.01366022429013</speedOfTravel>
<speedOfTravelUnit>Miles/hour</speedOfTravelUnit>
  <RoadType>primary highway rural</RoadType>
  <SegmentDirection>from_to</SegmentDirection>

  <StartJunctionType>Other</StartJunctionType>

  <EndJunctionType/>
  <IsRoundabout>false</IsRoundabout>
  <IsTollRoad>true</IsTollRoad>
  <PointsInSegment>
    <RouteDataPoint>
      <Longitude>-74.823861</Longitude>
      <Latitude>40.024421</Latitude>
    </RouteDataPoint>
    <RouteDataPoint>
      <Longitude>-74.824133</Longitude>
      <Latitude>40.024149</Latitude>
    </RouteDataPoint>
    ...
  </PointsInSegment>
</SegmentData>
</Segment>
</SegmentDetails>
</Segments>
</row>
</rows>
</GetRouteDataResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input

GetRouteData takes either a point latitude and longitude or a route segment ID as input. You also need to specify the route data source and coordinate system for the route data. The following table provides information on the format and layout of the input.

Table 94: GetRouteData Input Data

| Parameter | Description |
|---------------------|---|
| RouteDataPoint | The point to query to return segment information. You must specify a point which consists of a Latitude and Longitude . |
| RoutingSegmentID | The route segment ID to return segment information. |
| DataSetResourceName | The name of the database that contains the data to query for segment information. Use the database name specified in the Spectrum Spatial Routing Database Resource tool. For more information, see the <i>Spectrum Technology Platform Administration Guide</i> . |
| CoordinateSystem | The coordinate system of the input coordinates. For more information on EPSG codes, see www.spatialreference.org . To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <code>http://<server>:<port>/Spatial/GeometryService/DemoPage.html</code> . |

Options

Table 95: Get Route Data Options

| Parameter | Description |
|------------------|---|
| CoordinateFormat | This is a string value to specify the format of the coordinates for the point data returned in the response. The options are <code>Decimal</code> , <code>DecimalAssumed</code> , <code>PreZero</code> , <code>PreZeroDecimal</code> , and <code>DegMinSec</code> . |

| Parameter | Description |
|-------------------------|--|
| DistanceUnits | This is a string value to specify the format of the segment distance values (length of segment) returned in the response. The options are Feet, Yards, Miles, Meters, and Kilometers. |
| TimeUnits | This is a string value to specify the format of the segment time values (time to travel segment) returned in the response. The options are Milliseconds, Seconds, Minutes and Hours. |
| VelocityUnits | This is a string value to specify the format of the segment speed values returned in the response. The options are KPH (kilometers per hour), MPH (miles per hour), MTPS (meters per second), and MTPM (meters per minute). |
| AngularUnits | This is a string value to specify the format of the segment turn values (turn angle units) returned in the response. The options are radian, degree, minute, second, and grad. |
| ReturnActualPreferences | This is a string value to specify if the list of route preferences are returned in the response. The options are Y or N. The default is no route preferences are returned. |
| ReturnSegmentGeometry | This is a string value to specify the amount of detail to be returned for the segment geometry. The options are NONE, ALL, or END. If ALL is specified, the entire segment geometry will be returned. If END is specified, only the end point for the segments will be returned. The default is ALL segment geometries are returned. |

Response

Response

GetRouteData returns the following fields:

Table 96: GetRouteData Outputs

| Response Element | Format | Description |
|------------------|--------|---|
| RoutingSegmentID | String | One or more route segment IDs |
| SegmentData | String | <p>All information for the route segment. This information includes:</p> <ul style="list-style-type: none"> • PrimaryName • PrimaryNameLanguage • AlternateNameList • SegmentLength • SegmentLengthUnit • TimeTaken • TimeUnit • TurnAngle • TurnAngleUnit • speedOfTravel • speedOfTravelUnit • RoadType • SegmentDirection • StartJunctionType • IsRoundabout • IsTollRoad • PointsInSegment <p>The route geometries (points) for the segment is returned in the <code>PointsInSegment</code> parameter.</p> |

PersistentUpdate

PersistentUpdate allows changes to the routing data that are made at the server level and apply to all route requests or stages. These updates remain intact even if the server is restarted. Updates can be based on four types:

1. Point Updates
2. Segment Updates
3. Road-Type Updates
4. Reset Updates

Using persistent updates to make these types of modifications, you have the ability to:

- Exclude a point
- Exclude a segment
- Set the speed of a point, segment, or road type

- Change (increase or decrease) the speed of a point, segment, or road type by a value
- Change (increase or decrease) the speed of a point, segment, or road type by a percentage

Note: Persistent Update is only available as a service (Management Console and SOAP web service). Persistent Update is not available through a stage or REST API. It is also not available through the Java, C++, C, .NET, or COM APIs.

Note: Since persistent updates are changes made on a system-wide basis for routing data and all updates will persist, they should be used with caution.

PersistentUpdate is part of Spectrum Spatial.

Resource URL

```
http://server:port/soap/PersistentUpdate
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:per="http://www.g1.com/services/PersistentUpdate"
xmlns:typ="http://www.g1.com/services/erm/types">
  <soapenv:Header/>
  <soapenv:Body>
    <per:PersistentUpdateRequest>
      <per:context>
        <per:account.id>admin</per:account.id>
        <per:account.password>admin</per:account.password>
      </per:context>
      <per:options>
        <per:DataSetResourceName>US</per:DataSetResourceName>
        <per:RestoreDefaults>N</per:RestoreDefaults>
      </per:options>
      <per:rows>
        <per:row>
          <per:PersistentUpdates>
            <typ:UpdateList>
              <typ:Update>
                <typ:PointUpdate>
                  <typ:Point>
                    <typ:Latitude>34.40691</typ:Latitude>
                    <typ:Longitude>-80.062866</typ:Longitude>
                  </typ:Point>
                  <typ:SpeedUpdate>
                    <typ:Velocity
VelocityUnit="mph">15</typ:Velocity>
```

```

        </typ:SpeedUpdate>
    </typ:PointUpdate>
</typ:Update>
</typ:UpdateList>
</per:PersistentUpdates>
</per:row>
</per:rows>
</per:PersistentUpdateRequest>
</soapenv:Body>
</soapenv:Envelope>

```

Request

Parameters for Options

Input Options

Table 97: PersistentUpdate Input Options

| Parameter | Description |
|---------------------|---|
| DataSetResourceName | The name of the database that contains the data to use in the update process. Use the database name specified in the Spectrum Spatial Routing Database Resource tool. For more information, see the <i>Spectrum Technology Platform Administration Guide</i> . |
| CoordinateSystem | The coordinate system of the input coordinates. For more information on EPSG codes, see www.spatialreference.org . To retrieve a list of supported codespaces for EPSG, you can submit the SOAP request List Supported CoordSys by Code Space from the Geometry Service Demo page at <code>http://<server>:<port>/Spatial/GeometryService/DemoPage.html</code> . |
| RestoreDefaults | All persistent updates made to the <code>DataSetResourceName</code> will be reverted to their original state. Specify <code>Y</code> to restore the defaults. |

Point Options

This set of preferences allows you to set the point updates for each persistent update. Point updates are changes applied to a corresponding point (Latitude, Longitude). For a particular point, you can: exclude the point, set the speed of the point or change (increase or decrease) the speed of the point by a value or percentage.

You must specify a `Point` which consists of a `Latitude` and `Longitude`, then specify the type of point update. Each request can contain one or more updates.

Table 98: Point Update Options

| Parameter | Description |
|---------------|---|
| Point | The point to perform the persistent update. You must specify a point which consists of a Latitude and Longitude . |
| Exclude | This is a string value to exclude the specified point from all route calculations. To exclude a point you need to specify the point and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no). |
| Velocity | This is a speed update where you define the new speed of the point by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedIncrease | This is a speed update where you define an increase in the speed of the point by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedDecrease | This is a speed update where you define a decrease in the speed of the point by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |

Segment Options

This set of preferences allows you to set the segment updates for each persistent update. Segment updates are changes applied to a corresponding route segment ID. For a particular segment, you can: exclude the segment, set the speed of the segment, change (increase or decrease) the speed of the segment by a value or percentage, or change the road type of the segment.

You must specify a `RoutingSegmentID`, then specify the type of segment update. Each request can contain one or more updates.

Table 99: Segment Update Options

| Parameter | Description |
|------------------|--|
| RoutingSegmentID | The route segment ID to perform the persistent update. |
| Exclude | This is a string value to exclude the specified segment from all route calculations. To exclude a segment you need to specify the segment ID and include the <code>Exclude</code> parameter defined as Y. Valid values are Y (yes) and N (no). |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|----------|--|
| RoadType | |
|----------|--|

| Parameter | Description |
|-----------|--|
| | <p>This is a string value to change the value of the road type for the segment for the route calculation.</p> <p>The <code>RoadType</code> can be one of the following:</p> <ul style="list-style-type: none"> • access way • back road • connector • ferry • footpath • limited access dense urban • limited access rural • limited access suburban • limited access urban • local road dense urban • local road rural • local road suburban • local road urban • major local road dense urban • major local road rural • major local road suburban • major local road urban • major road dense urban • major road rural • major road suburban • major road urban • minor local road dense Urban • minor local road rural • minor local road suburban • minor local road urban • normal road dense urban • normal road rural • normal road rural • normal road urban • primary highway dense urban • primary highway rural • primary highway suburban • primary highway urban • ramp dense urban • ramp limited access • ramp major road • ramp primary highway • ramp rural • ramp secondary highway • ramp urban • ramp suburban |

| Parameter | Description |
|---------------|---|
| | <ul style="list-style-type: none"> • secondary highway dense urban • secondary highway rural • secondary highway suburban • secondary highway urban |
| Velocity | This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedIncrease | This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedDecrease | This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |

Road-type Options

This set of preferences allows you to set the road type updates for each persistent update. Road type updates are changes applied to a corresponding road type. For a particular road type, you can: set the speed of the roadtype, or change (increase or decrease) the speed of the road type by a value or percentage.

You must specify a `RoadType`, then specify the type of update. Each request can contain one or more updates.

Table 100: Road Type Update Options

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|----------|--|
| RoadType | |
|----------|--|

| Parameter | Description |
|-----------|---|
| | <p>This is a string value to change the speed value of the road type for the route calculation.</p> <p>The road type can be one of the following:</p> <ul style="list-style-type: none"> • AccessWay • Backroad • Connector • Ferry • Footpath • LimitedAccessDenseUrban • LimitedAccessRural • LimitedAccessSuburban • LimitedAccessUrban • LocalRoadDenseUrban • LocalRoadRural • LocalRoadSuburban • LocalRoadUrban • MajorLocalRoadDenseUrban • MajorLocalRoadRural • MajorLocalRoadSuburban • MajorLocalRoadUrban • MajorRoadDenseUrban • MajorRoadRural • MajorRoadSuburban • MajorRoadUrban • MinorLocalRoadDenseUrban • MinorLocalRoadRural • MinorLocalRoadSuburban • MinorLocalRoadUrban • NormalRoadDenseUrban • NormalRoadRural • NormalRoadSuburban • NormalRoadUrban • PrimaryHighwayDenseUrban • PrimaryHighwayRural • PrimaryHighwaySuburban • PrimaryHighwayUrban • RampDenseUrban • RampLimitedAccess • RampMajorRoad • RampPrimaryHighway • RampRural • RampSecondaryHighway • RampSuburban • RampUrban |

| Parameter | Description |
|---------------|---|
| | <ul style="list-style-type: none"> • SecondaryHighwayDenseUrban • SecondaryHighwayRural • SecondaryHighwaySuburban • SecondaryHighwayUrban |
| Velocity | This is a speed update where you define the new speed of the segment by specifying the velocity unit and new velocity. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedIncrease | This is a speed update where you define an increase in the speed of the segment by specifying either the velocity (unit and value) or a percentage to increase the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |
| SpeedDecrease | This is a speed update where you define a decrease in the speed of the segment by specifying either the velocity (unit and value) or a percentage to decrease the speed. For speed updates, the velocity unit can have one of the following values: kph (kilometers per hour), mph (miles per hour), mtps (meters per second), mtpm (meters per minute). |

Reset Options

This set of preferences allows you to reset (undo) the updates for each point, segment, or road type update. This will simply clear the updates already applied for a point/segment/road type from the server and set them to their default values.

Table 101: Reset Update Options

| Parameter | Description |
|------------|---|
| PointReset | The point that has the persistent update to be reset. You must specify a point which consists of a <code>Latitude</code> and <code>Longitude</code> , and specify the type of update to be reset for the point. The options for the <code>Reset Type</code> are <code>Exclude</code> and <code>Speed</code> . |

| Parameter | Description |
|---------------|---|
| SegmentReset | The segment that has the persistent update to be reset. You must specify the <code>RoutingSegmentID</code> and specify the type of update to be reset for the segment. The options for the <code>Reset Type</code> are <code>Exclude</code> , <code>Road Type</code> , and <code>Speed</code> . |
| RoadTypeReset | The road type that has the persistent update to be reset. By specifying a road type, the speed update applied to that type will be reset. |

Persistent Update SOAP Example

The following shows a standard PersistentUpdate SOAP request with all of the update options available (not a working example, this is used to show all the request syntax). You can have multiple Update or Reset definitions within a PersistentUpdates. You can only have a single update type (PointUpdate, SegmentUpdate, or RoadTypeUpdate) within an Update. Similarly you can only have a single reset type (PointReset, SegmentReset, or RoadTypeReset) within an Reset. You can also only have a single update or reset within one of the update or reset types.

To perform a reset on all updates, you only need to specify the `DataSetResourceName` and set the `RestoreDefaults` parameter to `Y`.

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:per="http://www.g1.com/services/PersistentUpdate"
xmlns:typ="http://www.g1.com/services/erm/types">
  <soapenv:Header/>
  <soapenv:Body>
    <per:PersistentUpdateRequest>
      <per:options>
        <per:DataSetResourceName>US</per:DataSetResourceName>
        <per:CoordinateSystem?></per:CoordinateSystem>
        <per:RestoreDefaults>N</per:RestoreDefaults>
      </per:options>
      <per:rows>
        <per:row>
          <per:PersistentUpdates>
            <typ:UpdateList>
              <typ:Update>
                <typ:PointUpdate>
                  <typ:Point>
                    <typ:Latitude?></typ:Latitude>
                    <typ:Longitude?></typ:Longitude>
                  </typ:Point>
                  <typ:SpeedUpdate>
                    <typ:Velocity VelocityUnit=""/>
                    <typ:SpeedIncrease>
```

```

        <typ:Velocity VelocityUnit="" />
        <typ:Percentage?></typ:Percentage>
    </typ:SpeedIncrease>
    <typ:SpeedDecrease>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
        <typ:Percentage?></typ:Percentage>
    </typ:SpeedDecrease>
</typ:SpeedUpdate>
    <typ:Exclude?></typ:Exclude>
</typ:PointUpdate>
<typ:SegmentUpdate>

<typ:RoutingSegmentID?></typ:RoutingSegmentID>
    <typ:SpeedUpdate>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:SpeedIncrease>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:Percentage?></typ:Percentage>
</typ:SpeedIncrease>
    <typ:SpeedDecrease>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:Percentage?></typ:Percentage>
</typ:SpeedDecrease>
</typ:SpeedUpdate>
    <typ:RoadType?></typ:RoadType>
    <typ:Exclude?></typ:Exclude>
</typ:SegmentUpdate>
    <typ:RoadTypeUpdate>
        <typ:RoadType?></typ:RoadType>
    <typ:SpeedUpdate>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:SpeedIncrease>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:Percentage?></typ:Percentage>
</typ:SpeedIncrease>
    <typ:SpeedDecrease>
        <typ:Velocity
VelocityUnit="">?</typ:Velocity>
    <typ:Percentage?></typ:Percentage>
</typ:SpeedDecrease>
</typ:SpeedUpdate>
    </typ:RoadTypeUpdate>
</typ:Update>
</typ:UpdateList>
<typ:ResetList>
    <typ:Reset>
        <typ:PointReset ResetType="">

```

```

        <typ:Point>
            <typ:Latitude>?</typ:Latitude>
            <typ:Longitude>?</typ:Longitude>
        </typ:Point>
    </typ:PointReset>
    <typ:SegmentReset ResetType="?">
        <typ:RoutingSegmentID>?</typ:RoutingSegmentID>
        </typ:SegmentReset>
        <typ:RoadTypeReset>
            <typ:RoadType>?</typ:RoadType>
        </typ:RoadTypeReset>
    </typ:Reset>
</typ:ResetList>
</per:PersistentUpdates>
</per:row>
</per:rows>
</per:PersistentUpdateRequest>
</soapenv:Body>
</soapenv:Envelope>

```

Response

Response

PersistentUpdate returns the following field:

Table 102: PersistentUpdate Outputs

| Response Element | Format | Description |
|------------------|--------|--|
| SuccessMessage | String | Returns if the update or reset was completed successfully. |

Spectrum Universal Address

AutoCompleteLoqate

AutoCompleteLoqate offers real-time entry of address data for fast, accurate results. Users are returned instant results based on each character entered into the form, ensuring only accurate data is entered into the database. AutoCompleteLoqate also includes the Powersearch option, which reduces input time by up to 80% for 238 countries by using data in the form of an index file.

Resource URL

```
http://server:port/soap/AutoCompleteLoqate
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:aut="http://www.precisely.com/spectrum/services/AutoCompleteLoqate"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <aut:AutoCompleteLoqateRequest>
      <aut:input_port>
        <aut:Address>
          <aut:AddressLine1>1 Global</aut:AddressLine1>
        </aut:Address>
      </aut:input_port>
    </aut:AutoCompleteLoqateRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: To make the example easier to read, empty response elements have been removed and only the first three address matches are shown.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:AutoCompleteLoqateResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/AutoCompleteLoqate">
      <ns3:output_port>
        <ns3:Address>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:HouseNumber>1</ns3:HouseNumber>
          <ns3:AddressLine1>1 Global Vw</ns3:AddressLine1>
          <ns3:FirmName>Map Info</ns3:FirmName>
          <ns3:City>Troy</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>12180-8399</ns3:PostalCode>
          <ns3:PostalCode.AddOn>8399</ns3:PostalCode.AddOn>
          <ns3:Country>United States</ns3:Country>
        </ns3:Address>
        <ns3:Address>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:HouseNumber>1</ns3:HouseNumber>
```

```

        <ns3:AddressLine1>1 Global Pl</ns3:AddressLine1>
        <ns3:City>Glendale</ns3:City>
        <ns3:StateProvince>AZ</ns3:StateProvince>
        <ns3:PostalCode>85306-3216</ns3:PostalCode>
        <ns3:PostalCode.AddOn>3216</ns3:PostalCode.AddOn>
        <ns3:Country>United States</ns3:Country>
    </ns3:Address>
    <ns3:Address>
        <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
        <ns3:HouseNumber>1</ns3:HouseNumber>
        <ns3:AddressLine1>1 Global Dr</ns3:AddressLine1>
        <ns3:City>Olive Hill</ns3:City>
        <ns3:StateProvince>KY</ns3:StateProvince>
        <ns3:PostalCode>41164-6739</ns3:PostalCode>
        <ns3:PostalCode.AddOn>6739</ns3:PostalCode.AddOn>
        <ns3:Country>United States</ns3:Country>
    </ns3:Address>
</ns3:output_port>
</ns3:AutoCompleteLoqateResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

The following table lists the input for AutoCompleteLoqate.

Table 103: Input Format

| Parameter | Description |
|--------------|--------------------------|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| City | The city name. |

| Parameter | Description |
|---------------|---|
| Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| FirmName | The company or firm name. |
| PostalCode | The postal code for the address. |
| StateProvince | The state or province. |

Parameters for Options

Table 104: AutoCompleteLoqate Options

| Parameter | Description |
|-----------------|---|
| Database.Loqate | Specifies the database to be used for address processing. Only databases that have been defined in the Database Resources panel in the Management Console are available. |
| OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p>123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p>123 MAIN ST MYTOWN FL 12345</p> |

| Parameter | Description |
|----------------------------|---|
| HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent and the Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |
| Option.OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |

| Parameter | Description |
|---------------------|--|
| OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |
| MaximumResults | <p>The maximum number of addresses that AutoCompleteLoqate should return. The default is 10.</p> |
| isPowersearchEnable | <p>Reduces input time by up to 80% for 240 countries by using data in the form of an index file. When you conduct a search, the Loqate Engine will first look for the corresponding index. If present, the method will attempt to instantly return a list of candidate addresses. If the index is not present, or if the index does not return any results, the original search process will be triggered.</p> <p>Note: Powersearch can be performed when there are two and only two fields in the input file: the Country field and any one of the AddressLine fields. If you select this option and your input file contains additional fields, the original search process will automatically be triggered.</p> <p>To conduct its search, Auto Complete indexes use up to the first 10 characters for searches within the United States and up to the first 15 characters for searches within all other eligible countries. Spaces and punctuation are not factored into this count.</p> <p>Powersearch cannot be used for the following countries: Botswana, Ethiopia, India, Kazakhstan, Malaysia, Mongolia, Saint Kitts and Nevis, and San Marino.</p> <p>Note: You must have a valid license for Powersearch processing. If you select this option but are not licensed for Powersearch, or if your license has expired, you will receive an error.</p> |

| Parameter | Description |
|-------------------------------|--|
| IsDuplicateHandlingMaskEnable | <p>Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options:</p> <p>S Selected by default. Pre-process the input and remove duplicates that occur in a single field.</p> <p>C Selected by default. Preprocess the input and remove duplicates across all fields.</p> <p>T Preprocess the input and remove duplicates in fields that are not standard address fields.</p> <p>F Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.</p> |
| FailJobOnDataLicenseError | <p>Specifies how you want Spectrum Technology Platform to respond when a data license error occurs.</p> <p>Fail the job Fail the entire job if a data license error occurs.</p> <p>Fail the record Fail the record(s) for which the data license error occurs and continue processing.</p> |

Response

The output from AutoCompleteLoqate is optional and corresponds directly to the fields you selected in the Output Fields section of the AutoCompleteLoqate Options dialog box.

Table 105: AutoCompleteLoqate Output

| Response Element | Description |
|------------------|--------------------------|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |

| Response Element | Description |
|------------------|--|
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| FirmName | The firm name. |
| HouseNumber | The ending house number for the range in which the candidate address's house number falls. |
| PostalCode | The postal code. |
| PostalCode.AddOn | The last four digits of the ZIP + 4 [®] Code. |
| ProcessedBy | Indicates which address coder processed the address. LOQATE The Loqate coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. <ul style="list-style-type: none"> • DisabledCoder • RequestFailed • NoLookupAddressFound |

| Response Element | Description |
|--------------------|---|
| Status.Description | A description of the problem, if there is one. |
| | <p>Did not return multiples The input address matched only one address in the database. AutoCompleteLoqate returns data only if multiple possible matches were found.</p> <p>Not able to look up the address pattern AutoCompleteLoqate is not able to process the partial address.</p> |

AutoCompleteLoqate Sample Web Application

You can access a sample web application that demonstrates the Auto Complete Loqate functionality. When you enter a partial address, this application makes a call to the Auto Complete Loqate REST web service, which returns a suggested address.

Note: Prior to using this feature, you must add an Auto Complete Loqate database resource in Management Console and save the database resource in the Auto Complete Loqate Service.

1. Be sure the Spectrum Technology Platform server is running.
2. Open a web browser and go to: `http://<servername>:<port>/autocomplete`. For example, if your server is named "myserver" and it uses the default HTTP port 8080, you would go to: `http://myserver:8080/autocomplete`.

Note: This site is best viewed in Internet Explorer 8.0 or later, Chrome, or Mozilla Firefox.

3. When the login screen appears, enter "**guest**" as the user name and leave the password field blank.
4. Press **OK**.
5. Select a country from the drop-down list.
6. Begin typing your address in any of the fields provided.
7. Select from the list of suggested addresses.
8. To begin a new call, click **Reset**, which will clear the fields you used in your previous call.

GetCandidateAddresses

GetCandidateAddresses returns a list of addresses that are considered matches for a given input address. GetCandidateAddresses returns candidate addresses only if the input address matches multiple addresses in the postal database. If the input address matches only one address in the postal database, then no address data is returned.

For addresses outside the U.S. and Canada, you may notice inconsistent results between the multiple matches returned by ValidateAddress and the results for that same address returned by

GetCandidateAddresses. If you experience inconsistent results, it is likely because you set the performance tuning setting in ValidateAddress to a value other than 100. To obtain consistent results between GetCandidateAddresses and ValidateAddress, set the performance tuning option to 100.

Note: By default, GetCandidateAddresses does not match to individual house numbers. Rather, it uses house number ranges for each street. After GetCandidateAddresses has determined the street name, city name, state/province name, and postal code, it checks to make sure the input house number falls within one of the ranges of house numbers given for the matched street name. The same type of logic applies to unit numbers. If you want to determine that an individual house number is valid, you should use the ValidateAddress Delivery Point Validation (DPV) processing option. DPV processing is only available for U.S. addresses.

The Canadian coder contains a reverse lookup routine that takes as input a specific postal code and returns the street information stored in the database for that postal code. To use this function enter nothing but a Canadian postal code in the PostalCode field. See the second example to view the return from a sample postal code.

GetCandidateAddresses is part of Spectrum Universal Address.

Resource URL

```
http://server:port/soap/GetCandidateAddresses
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.precisely.com/spectrum/services/GetCandidateAddresses"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetCandidateAddressesRequest>
      <get:input_port>
        <get:Address>
          <get:AddressLine1>P.O. Box 1</get:AddressLine1>
          <get:City>New York</get:City>
          <get:StateProvince>NY</get:StateProvince>
        </get:Address>
      </get:input_port>
    </get:GetCandidateAddressesRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: Empty response elements have been removed from this example. Only the first two candidate address are shown.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:GetCandidateAddressesResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/GetCandidateAddresses">

      <ns3:output_port>
        <ns3:Address>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:RecordType>PostOfficeBox</ns3:RecordType>
          <ns3:MatchLevel>A</ns3:MatchLevel>
          <ns3:AddressLine1>PO Box 1</ns3:AddressLine1>
          <ns3:HouseNumberLow>1</ns3:HouseNumberLow>
          <ns3:HouseNumberHigh>60</ns3:HouseNumberHigh>
          <ns3:HouseNumberParity>B</ns3:HouseNumberParity>
          <ns3:City>New York</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>10002</ns3:PostalCode>
          <ns3:PostalCode.AddOn>0001</ns3:PostalCode.AddOn>
          <ns3:Country>USA</ns3:Country>
        </ns3:Address>
        <ns3:Address>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:RecordType>PostOfficeBox</ns3:RecordType>
          <ns3:MatchLevel>A</ns3:MatchLevel>
          <ns3:AddressLine1>PO Box 1</ns3:AddressLine1>
          <ns3:HouseNumberLow>1</ns3:HouseNumberLow>
          <ns3:HouseNumberHigh>9</ns3:HouseNumberHigh>
          <ns3:HouseNumberParity>B</ns3:HouseNumberParity>
          <ns3:City>New York</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>10008</ns3:PostalCode>
          <ns3:PostalCode.AddOn>0001</ns3:PostalCode.AddOn>
          <ns3:Country>USA</ns3:Country>
        </ns3:Address>
      </ns3:output_port>
    </ns3:GetCandidateAddressesResponse>
  </soap:Body>
</soap:Envelope>
```

Request

Parameters for Input Data

The following table lists the input for GetCandidateAddresses.

Table 106: Input Format

| Parameter | Description |
|---------------|---|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. Does not apply to U.S. and Canadian addresses. |
| AddressLine4 | The fourth address line. Does not apply to U.S. and Canadian addresses. |
| AddressLine5 | The fifth address line. Applies only to U.K. addresses. May contain street name, unit number, building number, and so on. |
| City | The city name. |
| StateProvince | The state or province. For U.S. addresses only, you may put the state in the City field instead of the StateProvince field. |
| PostalCode | The postal code for the address. For U.S. addresses this is the ZIP Code™ in one of the following formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 Note: For Canadian addresses you can complete just this field and have candidate address data returned. For other countries, AddressLine1 and AddressLine2 must also be completed. |

| Parameter | Description |
|-------------|---|
| Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name • French country name • German country name • Spanish country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| FirmName | The company or firm name. |
| USUrbanName | U.S. address urbanization name. Used primarily for Puerto Rico addresses. |

Parameters for Options

Table 107: GetCandidateAddresses Options

| Parameter | Description |
|---------------------|---|
| PerformUSProcessing | <p>Specifies whether or not to process U.S. addresses. If you enable U.S. address processing GetCandidateAddresses will attempt to retrieve candidate addresses for U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process U.S. addresses (default).</p> <p>N No, do not process U.S. addresses.</p> |

| Parameter | Description |
|---------------------------|---|
| Database.US | Specifies the database to be used for U.S. address processing. Only databases that have been defined in the US Database Resources panel in the Management Console are available. |
| PerformCanadianProcessing | <p>Specifies whether or not to process Canadian addresses. If you enable Canadian address processing GetCandidateAddresses will attempt to retrieve candidate addresses for Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process Canadian addresses (default).</p> <p>N No, do not process Canadian addresses.</p> |
| Database.Canada | Specifies the database to be used for Canadian address processing. Only databases that have been defined in the Canadian Database Resources panel in the Management Console are available. |

| Parameter | Description |
|--------------------------------|--|
| PerformInternationalProcessing | <p>Specifies whether or not to process international addresses (addresses outside the U.S. and Canada). If you enable international address processing GetCandidateAddresses will attempt to retrieve candidate addresses for international addresses. If you disable international address processing, international addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for international address processing you must disable international address processing in order for your jobs to complete successfully, regardless of whether or not they contain international addresses.</p> <p>Note: You must have a valid license for international address processing to successfully process international addresses. If you enable international address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process international addresses (default).</p> <p>N No, do not process international addresses.</p> |
| Database.International | <p>Specifies the database to be used for international address processing. Only databases that have been defined in the International Database Resources panel in the Management Console are available.</p> |
| OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p>123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p>123 MAIN ST MYTOWN FL 12345</p> |
| MaximumResults | <p>The maximum number of candidate addresses that GetCandidateAddresses should return. The default is 10. The maximum is 10.</p> |

| Parameter | Description |
|--------------------------|--|
| OutputShortCityName | <p>For U.S. addresses, specifies whether or not to return the USPS®-approved abbreviation for the city, if there is one. The USPS® provides abbreviations for city names that are 14 characters long or longer. City abbreviations are 13 characters or less and can be used when there is limited space on the mailing label. If there is no short city name for the city, then the full city name is returned.</p> <p>Y Yes, return the short city name.</p> <p>N No, do not return the short city name.</p> |
| DualAddressLogic | <p>(U.S. addresses only). Controls whether GetCandidateAddresses should return a street match or a PO Box/Rural Route/Highway Contract match when the address contains both street and PO Box/Rural Route/Highway Contract information. For more information, see About Dual Address Logic on page 465.</p> <p>N (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority:</p> <ol style="list-style-type: none"> 1. PO Box 2. Firm 3. Highrise 4. Street 5. Rural Route 6. General Delivery <p>S Return a street match, regardless of the address line.</p> <p>P Return a PO Box match, regardless of the address line.</p> |
| StreetMatchingStrictness | <p>The strictness of the street name match (U.S. addresses only).</p> <p>E The input street name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| FirmMatchingStrictness | <p>The strictness of the firm name match (U.S. addresses only).</p> <p>E The input firm name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |

| Parameter | Description |
|-------------------------------|---|
| DirectionalMatchingStrictness | <p>The strictness of the directional match.</p> <p>E The input directional must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| PerformESM | <p>Specifies whether or not to perform Enhanced Street Matching (ESM). ESM applies extra matching logic with additional data to any input address that is not matched through the regular address validation process. ESM applies to U.S. addresses only.</p> <p>Y Yes, perform ESM processing.</p> <p>N No, do not perform ESM processing (default).</p> |
| AddressLineSearchOnFail | <p>Specifies whether ValidateAddress will search address lines for the city, state/province, and postal code.</p> <p>This option enables ValidateAddress to search the AddressLine input fields for the city, state/province, postal code, and country when the address cannot be matched using the values in the City, StateProvince, and PostalCode input fields.</p> <p>Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields.</p> <p>Consider disabling this option if your input addresses use the City, State/Province and PostalCode fields. If you enable this option and these fields are used, there is an increased possibility that ValidateAddress will fail to correct values in these fields (for example a misspelled city name).</p> <p>Y Yes, search the address line fields (default).</p> <p>N No, do not search the AddressLine fields.</p> |

Response

GetCandidateAddresses returns the following output.

Table 108: GetCandidateAddresses Output

| Response Element | Description |
|---|--|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| AddressLine5 | For U.K. addresses only. If the address was validated, the fifth line of the validated and standardized address. If the address could not be validated, the fifth line of the input address without any changes. |
| CanadianDeliveryInstallation AreaName | Delivery installation name (Canadian addresses only) |
| CanadianDeliveryInstallation QualifierName | Delivery installation qualifier (Canadian addresses only) |
| CanadianDeliveryInstallation Type | Delivery installation type (Canadian addresses only) |
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| FirmName | The firm name. |

| Response Element | Description |
|-------------------|---|
| HouseNumberHigh | The ending house number for the range in which the candidate address's house number falls. |
| HouseNumberLow | The beginning house number for the range in which the candidate address's house number falls. |
| HouseNumberParity | <p>Indicates the numbering scheme for the house numbers between HouseNumberLow and HouseNumberHigh, as follows:</p> <p>E Only even values</p> <p>O Only odd values</p> <p>B Both</p> |
| MatchLevel | <p>For addresses outside the U.S. and Canada, identifies the match level for the candidate address. U.S. and Canadian addresses are always "A." One of the following:</p> <p>A The candidate matches the input address at the street level.</p> <p>B The candidate matches the input address at the state/province level.</p> |
| PostalCode | The postal code. In the U.S. this is the ZIP Code™. |
| PostalCode.AddOn | The last four digits of the ZIP + 4® Code. U.S. addresses only. |
| RecordType | <p>The type of address record, as defined by U.S. and Canadian postal authorities (U.S. and Canadian addresses only):</p> <ul style="list-style-type: none"> • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal |

| Response Element | Description |
|--------------------|--|
| RecordType.Default | Code indicating the "default" match: Y The address matches a default record. null The address does not match a default record. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. There is only one possible value: <ul style="list-style-type: none"> • DisabledCoder • RequestFailed |
| Status.Description | A description of the problem, if there is one. Did not return multiples The input address matched only one address in the database. GetCandidateAddresses only returns data if multiple possible matches were found. Number of candidates is not greater than 1 The input address matched more than one address in the database but no addresses were returned. PerformUSProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformCanadianProcessing disabled This value will appear if Status.Code=DisabledCoder. PerformInternationalProcessing disabled This value will appear if Status.Code=DisabledCoder. |
| UnitNumberHigh | The ending unit number for the range in which the candidate address's unit number falls. |

| Response Element | Description | | | | | | |
|------------------|---|----------|------------------|----------|-----------------|----------|------|
| UnitNumberLow | The beginning unit number for the range in which the candidate address's unit number falls. | | | | | | |
| UnitNumberParity | Indicates the numbering scheme for the unit numbers between UnitNumberLow and UnitNumberHigh, as follows: <table border="0" data-bbox="552 577 974 714"> <tr> <td>E</td> <td>Only even values</td> </tr> <tr> <td>O</td> <td>Only odd values</td> </tr> <tr> <td>B</td> <td>Both</td> </tr> </table> | E | Only even values | O | Only odd values | B | Both |
| E | Only even values | | | | | | |
| O | Only odd values | | | | | | |
| B | Both | | | | | | |
| USUrbanName | The validated city urbanization name. Urbanization names are used primarily for Puerto Rico addresses. | | | | | | |

GetCandidateAddressesLoqate

GetCandidateAddressesLoqate returns a list of addresses that are considered matches for a given input address. GetCandidateAddressesLoqate returns candidate addresses only if the input address matches multiple addresses in the postal database. If the input address matches only one address in the postal database, then no address data is returned. The Country input field is required; if this field is blank, no output will be returned.

Note: By default, GetCandidateAddressesLoqate does not match to individual house numbers. Rather, it uses house number ranges for each street. After GetCandidateAddressesLoqate has determined the street name, city name, state/province name, and postal code, it checks to make sure the input house number falls within one of the ranges of house numbers given for the matched street name. The same type of logic applies to unit numbers.

GetCandidateAddressesLoqate is part of Spectrum Universal Address.

Resource URL

```
http://server:port/soap/GetCandidateAddressesLoqate
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.precisely.com/spectrum/services/GetCandidateAddressesLoqate"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetCandidateAddressesLoqateRequest>
      <get:input_port>
        <get:Address>
          <get:AddressLine1>PO Box 1</get:AddressLine1>
          <get:City>New York</get:City>
          <get:StateProvince>NY</get:StateProvince>
        </get:Address>
      </get:input_port>
    </get:GetCandidateAddressesLoqateRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

Note: Empty response elements have been removed from this example. Only the first two candidate address are shown.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:GetCandidateAddressesLoqateResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/GetCandidateAddressesLoqate">

      <ns3:output_port>
        <ns3:Address>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:AddressLine1>PO Box 101</ns3:AddressLine1>
          <ns3:City>New York Mls</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>13417-0101</ns3:PostalCode>
          <ns3:PostalCode.AddOn>0101</ns3:PostalCode.AddOn>
          <ns3:Country>USA</ns3:Country>
        </ns3:Address>
        <ns3:Address>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:AddressLine1>PO Box 102</ns3:AddressLine1>
          <ns3:City>New York Mls</ns3:City>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:PostalCode>13417-0102</ns3:PostalCode>
          <ns3:PostalCode.AddOn>0102</ns3:PostalCode.AddOn>
          <ns3:Country>USA</ns3:Country>
```

```

        </ns3:Address>
    </ns3:output_port>
</ns3:GetCandidateAddressesLoqateResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

The following table lists the input for GetCandidateAddressesLoqate.

Table 109: Input Format

| Parameter | Description |
|--------------|---|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| City | The city name. |
| Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> <p>Note: This field is required. If this field is blank, no output will be returned.</p> |
| FirmName | The company or firm name. |

| Parameter | Description |
|---------------|--|
| PostalCode | The postal code for the address. For U.S. addresses this is the ZIP Code™ in one of the following formats: |
| StateProvince | The state or province. For U.S. addresses only, you may put the state in the City field instead of the StateProvince field. |

Parameters for Options

Table 110: GetCandidateAddressesLoqate Options

| Parameter | Description |
|------------------------|--|
| Database.Loqate | Specifies the database to be used for address processing. Only databases that have been defined in the Management Console are available. |
| OutputCasing | Specifies the casing of the output data. One of the following: <ul style="list-style-type: none"> M Returns the output in mixed case (default). For example: 123 Main St Mytown FL 12345 U Returns the output in upper case. For example: 123 MAIN ST MYTOWN FL 12345 |
| CandidateProcessOption | Specifies the method of searching for candidates. One of the following: <ul style="list-style-type: none"> S Enter a full or partial address as input and return as output a list of closely matching results (default). V Enter address information in address lines, address components, or a combination of both as input and return as output results that more closely match the input. |

| Parameter | Description |
|----------------------------|---|
| HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. GetCandidateAddressLoqate uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent And The Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |
| Option.OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |

| Parameter | Description |
|---------------------|--|
| Option.OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |
| MaximumResults | The maximum number of candidate addresses that GetCandidateAddressesLoqate should return. The default is 10. The maximum is 99. |

Response

GetCandidateAddressesLoqate returns the following output.

Table 111: GetCandidateAddressesLoqate Output

| Response Element | Description |
|------------------|--|
| AddressLine1 | The first address line. |
| AddressLine2 | The second address line. |
| AddressLine3 | The third address line. |
| AddressLine4 | The fourth address line. |
| City | The city name. |
| Country | The three-character ISO 3166-1 Alpha 3 code for the country. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |

| Response Element | Description |
|--------------------|--|
| FirmName | The firm name. |
| PostalCode | The postal code. In the U.S. this is the ZIP Code™. |
| PostalCode.AddOn | The last four digits of the ZIP + 4® Code. U.S. addresses only. |
| ProcessedBy | Indicates which address coder processed the address. LOQATE The Loqate coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. There is only one possible value: • RequestFailed |
| Status.Description | A description of the problem, if there is one. There is only one possible value: Did not return multiples The input address matched only one address in the database. GetCandidateAddressesLoqate only returns data if multiple possible matches were found. |

GetCityStateProvince

GetCityStateProvince returns a city and state/province for a given input postal code.

Note: GetCityStateProvince works with U.S. and Canadian addresses only.

GetCityStateProvince is part of Spectrum Universal Address.

Resource URL

```
http://server:port/soap/GetCityStateProvince
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.precisely.com/spectrum/services/GetCityStateProvince"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetCityStateProvinceRequest>
      <get:input_port>
        <get:Input>
          <get:PostalCode>60510</get:PostalCode>
        </get:Input>
      </get:input_port>
    </get:GetCityStateProvinceRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:GetCityStateProvinceResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/GetCityStateProvince">
      <ns3:output_port>
        <ns3:Result>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:PostalCode>60510</ns3:PostalCode>
          <ns3:City>BATAVIA</ns3:City>
          <ns3:City.Type>P</ns3:City.Type>
          <ns3:StateProvince>IL</ns3:StateProvince>
          <ns3:Country>USA</ns3:Country>
          <ns3:user_fields/>
        </ns3:Result>
      </ns3:output_port>
    </ns3:GetCityStateProvinceResponse>
  </soap:Body>
</soap:Envelope>
```

Request

Parameters for Input Data

The following table shows the input fields.

Table 112: GetCityStateProvince Input

| Parameter | Description |
|------------|---|
| PostalCode | A U.S. ZIP Code™ or Canadian postal code in one of the following formats: 99999 99999-9999 A9A9A9 A9A 9A9 |

Parameters for Options

Table 113: GetCityStateProvince Options

| Parameter Name | Description |
|---------------------|--|
| PerformUSProcessing | <p>Specifies whether or not to process U.S. addresses. If you enable U.S. address processing GetCityStateProvince will attempt to return the state for U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process U.S. addresses (default).</p> <p>N No, do not process U.S. addresses.</p> |
| Database.US | Specifies the database to be used for U.S. address processing. Only databases that have been defined in the US Database Resources panel in the Management Console are available. |

| Parameter Name | Description |
|---------------------------|---|
| PerformCanadianProcessing | <p>Specifies whether or not to process Canadian addresses. If you enable Canadian address processing GetCityStateProvince will attempt to return the province for Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>Y Yes, process Canadian addresses (default).</p> <p>N No, do not process Canadian addresses.</p> |
| Database.Canada | <p>Specifies the database to be used for Canadian address processing. Only databases that have been defined in the Canadian Database Resources panel in the Management Console are available.</p> |
| OutputVanityCity | <p>Specifies whether or not to include non-mailing city names in the output. A non-mailing city name is an alternate name for the primary city name. For example, Hollywood is a non-mailing city name for Los Angeles.</p> <p>Y Yes, include non-mailing city names.</p> <p>N No, do not include non-mailing city names (default).</p> |
| MaximumResults | <p>Specifies the maximum number of city-state/province pairs to return. The default value is 10.</p> |

Response

GetCityStateProvince returns the matching city and state/province for the input postal code as well as a code to indicate the success or failure of the match attempt. If more than one city/state or city/province matches the input postal code, multiple output records are returned.

Table 114: GetCityStateProvince Output

| Response Element | Description |
|------------------|--|
| City | The matched city name. |
| City.Type | The USPS® standardized city name type (U.S. addresses only). V Vanity (non-mailing) city name. P Primary. The city name is the primary mailing city name. S Secondary. The city name is an alternate city name but is acceptable. A city can have multiple secondary city names. |
| PostalCode | The input postal code. |
| ProcessedBy | Indicates which address coder processed the address. One of the following: USA The U.S. address coder processed the address. CAN The Canadian address coder processed the address. |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. null Success F Failure |
| Status.Code | The reason for failure, if there is one. The only valid value is: <ul style="list-style-type: none"> • DisabledCoder • UnrecognizedPostalCode |

| Response Element | Description | | | | | | |
|---|---|------------------------------|---|-------------------------------------|--|---|--|
| Status.Description | The description of the failure. The valid values are: <table border="0"> <tr> <td>Postal code not found</td> <td>This value will appear if Status.Code=UnrecognizedPostalCode.</td> </tr> <tr> <td>PerformUSProcessing disabled</td> <td>This value will appear if Status.Code=DisabledCoder.</td> </tr> <tr> <td>PerformCanadianProcessing disabled</td> <td>This value will appear if Status.Code=DisabledCoder.</td> </tr> </table> | Postal code not found | This value will appear if Status.Code=UnrecognizedPostalCode. | PerformUSProcessing disabled | This value will appear if Status.Code=DisabledCoder. | PerformCanadianProcessing disabled | This value will appear if Status.Code=DisabledCoder. |
| Postal code not found | This value will appear if Status.Code=UnrecognizedPostalCode. | | | | | | |
| PerformUSProcessing disabled | This value will appear if Status.Code=DisabledCoder. | | | | | | |
| PerformCanadianProcessing disabled | This value will appear if Status.Code=DisabledCoder. | | | | | | |

GetCityStateProvinceLoqate

GetCityStateProvinceLoqate returns a city and state/province for a given input postal code.

This stage is part of the Spectrum Universal Adresse.

Resource URL

```
http://server:port/soap/GetCityStateProvinceLoqate
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.precisely.com/spectrum/services/GetCityStateProvinceLoqate"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetCityStateProvinceLoqateRequest>
      <get:input_port>
        <get:Input>
          <get:PostalCode>60510</get:PostalCode>
          <get:Country>USA</get:Country>
        </get:Input>
      </get:input_port>
    </get:GetCityStateProvinceLoqateRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:GetCityStateProvinceLoqateResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/GetCityStateProvinceLoqate">

      <ns3:output_port>
        <ns3:Result>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:PostalCode>60510</ns3:PostalCode>
          <ns3:City>Batavia</ns3:City>
          <ns3:StateProvince>IL</ns3:StateProvince>
          <ns3:Country>United States</ns3:Country>
          <ns3:Status/>
          <ns3:Status.Code/>
          <ns3:Status.Description/>
          <ns3:user_fields/>
        </ns3:Result>
      </ns3:output_port>
    </ns3:GetCityStateProvinceLoqateResponse>
  </soap:Body>
</soap:Envelope>
```

Request

Parameters for Input Data

The following table shows the input fields.

Table 115: GetCityStateProvinceLoqate Input

| Parameter | Description |
|------------|---|
| Country | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • 2-digit ISO country code • 3-digit UPU Country code • English country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| PostalCode | The postal code for the address. |

Options

Table 116: GetCityStateProvinceLoqate Options

| | Description / Valid Values | | | | | | |
|------------------------|--|---------------------|--|------------------------|---|-------------|---------------------|
| | Specifies the database to be used for address processing. Only databases that have been defined in the Database Resources panel in the Management Console are available. | | | | | | |
| | The maximum number of addresses that GetCityStateProvinceLoqate should return. The default is 10. | | | | | | |
| OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <table border="0"> <tr> <td>Input</td> <td>Do not perform transliteration and provide output in the same script as the input (default).</td> </tr> <tr> <td>Native</td> <td>Output in the native script for the selected country wherever possible.</td> </tr> <tr> <td>Latn</td> <td>Use English values.</td> </tr> </table> | Input | Do not perform transliteration and provide output in the same script as the input (default). | Native | Output in the native script for the selected country wherever possible. | Latn | Use English values. |
| Input | Do not perform transliteration and provide output in the same script as the input (default). | | | | | | |
| Native | Output in the native script for the selected country wherever possible. | | | | | | |
| Latn | Use English values. | | | | | | |
| | <p>Specifies how you want Spectrum Technology Platform to respond when a data license error occurs.</p> <table border="0"> <tr> <td>Fail the job</td> <td>Fail the entire job if a data license error occurs.</td> </tr> <tr> <td>Fail the record</td> <td>Fail the record(s) for which the data license error occurs and continue processing.</td> </tr> </table> | Fail the job | Fail the entire job if a data license error occurs. | Fail the record | Fail the record(s) for which the data license error occurs and continue processing. | | |
| Fail the job | Fail the entire job if a data license error occurs. | | | | | | |
| Fail the record | Fail the record(s) for which the data license error occurs and continue processing. | | | | | | |

Response

GetCityStateProvinceLoqate returns the matching city and state/province for the input postal code as well as a code to indicate the success or failure of the match attempt. If more than one city/state or city/province matches the input postal code, multiple output records are returned.

Table 117: GetCityStateProvinceLoqate Output

| Response Element | Description |
|--------------------|---|
| City | The matched city name. |
| Country | The country in the format determined by what you selected in: <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| PostalCode | The input postal code. |
| ProcessedBy | Indicates which address coder processed the address. <p>LOQATE The Loqate coder processed the address.</p> |
| StateProvince | The validated state/province or its abbreviated value. |
| Status | Reports the success or failure of the match attempt. <p>null Success</p> <p>F Failure</p> |
| Status.Code | The reason for failure, if there is one. The only valid value is: <ul style="list-style-type: none"> • UnrecognizedPostalCode |
| Status.Description | The description of the failure. The only valid value is: <p>Postal code not found This value will appear if Status.Code=UnrecognizedPostalCode.</p> |

GetPostalCodes

GetPostalCodes allows you to look up the postal codes for a particular city. The service takes a city, state, and country as input and returns the postal codes for that city. The input must be exactly correct in order to return postal codes.

Note: GetPostalCodes only works with U.S. addresses.

GetPostalCodes is part of the Spectrum Universal Address.

Resource URL

```
http://server:port/soap/GetPostalCodes
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:get="http://www.precisely.com/spectrum/services/GetPostalCodes"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <get:GetPostalCodesRequest>
      <get:input_port>
        <get:Input>
          <get:City>Holland</get:City>
          <get:StateProvince>MI</get:StateProvince>
        </get:Input>
      </get:input_port>
    </get:GetPostalCodesRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:GetPostalCodesResponse
xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/GetPostalCodes">
      <ns3:output_port>
        <ns3:Result>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:PostalCode>49422</ns3:PostalCode>
          <ns3:City.Type></ns3:City.Type>
          <ns3:Status/>
        </ns3:Result>
      </ns3:output_port>
    </ns3:GetPostalCodesResponse>
  </soap:Body>
</soap:Envelope>
```

```

        <ns3:Status.Code/>
        <ns3:Status.Description/>
        <ns3:user_fields/>
    </ns3:Result>
    <ns3:Result>
        <ns3:ProcessedBy>USA</ns3:ProcessedBy>
        <ns3:PostalCode>49423</ns3:PostalCode>
        <ns3:City.Type></ns3:City.Type>
        <ns3:Status/>
        <ns3:Status.Code/>
        <ns3:Status.Description/>
        <ns3:user_fields/>
    </ns3:Result>
    <ns3:Result>
        <ns3:ProcessedBy>USA</ns3:ProcessedBy>
        <ns3:PostalCode>49424</ns3:PostalCode>
        <ns3:City.Type></ns3:City.Type>
        <ns3:Status/>
        <ns3:Status.Code/>
        <ns3:Status.Description/>
        <ns3:user_fields/>
    </ns3:Result>
</ns3:output_port>
</ns3:GetPostalCodesResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

GetPostalCodes takes a city, state/province, and country as input.

Table 118: GetPostalCodes Input

| Parameter | Description |
|---------------|---|
| City | <p>The city whose postal codes you want to look up.</p> <p>You may put the city and state in the City field. If you do this, you must leave the StateProvince field blank.</p> <p>The total length of the City and StateProvince fields cannot exceed 100 characters.</p> |
| StateProvince | <p>The state or province of the city whose postal codes you want to look up.</p> <p>You may also put the state in the City field instead of the StateProvince field.</p> <p>The total length of the City and StateProvince fields cannot exceed 100 characters.</p> |

| Parameter | Description |
|-----------|--|
| Country | The country code or name of the city whose postal codes you want to look up. The only valid value is US. |

Parameters for Options

Table 119: GetPostalCodes Options

| Parameter | Description |
|-------------------|--|
| Database.US | Specifies the database to be used for postal code look-ups. Only databases that have been defined in the US Database Resources panel in the Management Console are available. |
| IncludeVanityCity | Specifies whether or not to include postal codes for the city's non-mailing city names. A non-mailing city name is an alternate name for the primary city name. For example, Hollywood is a non-mailing city name for Los Angeles. Y Yes, include postal codes for non-mailing city names. N No, do not include postal codes for non-mailing city names (default). |
| OutputCityType | Specifies whether or not to return the city type in the output. If enabled, the city type is returned in the City.Type field. Y Yes, include the city type in the output. N No, do not include the city type in the output (default). |

Response

GetPostalCodes returns the postal codes for a specified city. Each postal code is returned in a separate record along with the data listed in the following table.

Table 120: GetPostalCodes Output

| Response Element | Description |
|--------------------|--|
| City.Type | <p>The USPS® city type (U.S. addresses only). The city type is determined by looking at the ZIP Code and the city name. For example, the city Lanham MD has the postal codes 20703, 20706, and 20784. Lanham is the primary city in 20703 and 20706 but is a vanity city in 20784.</p> <p>This field column is only populated if <code>OutputCityType=Y</code>. The possible values are:</p> <p>V Vanity (non-mailing) city name.</p> <p>P Primary. The city name is the primary mailing city name.</p> <p>S Secondary. The city name is an alternate city name but is acceptable. A city can have multiple secondary city names.</p> |
| PostalCode | A postal code in the specified city. |
| ProcessedBy | Because this service only works for U.S. addresses, <code>ProcessedBy</code> will always contain one value: USA. |
| Status | <p>Reports the success or failure of the match attempt.</p> <p>null Success</p> <p>F Failure</p> |
| Status.Code | <p>Reason for failure, if there is one. One of the following:</p> <ul style="list-style-type: none"> • CountryNotSupported • UnableToLookup |
| Status.Description | <p>Description of failure.</p> <ul style="list-style-type: none"> • Input country is not supported • Input city was blank • Input city & state / province was blank, or no match found • City-state mismatch (different spelling found, or city-state was a vanity name and vanity matching was not allowed, or city-state did not match ZIP Code) |

ValidateAddress

ValidateAddress standardizes and validates addresses using postal authority address data. It can correct information and format the address using the format preferred by the applicable postal authority. It also adds missing postal information, such as postal codes, city names, state/province names, and more.

ValidateAddress also returns result indicators about validation attempts, such as whether or not it validated the address, the level of confidence in the returned address, the reason for failure if the address could not be validated, and more.

During address matching and standardization, ValidateAddress separates address lines into components and compares them to the contents of the Universal Addressing Module databases. If a match is found, the input address is *standardized* to the database information. If no database match is found, it optionally *formats* the input addresses. The formatting process attempts to structure the address lines according to the conventions of the appropriate postal authority.

ValidateAddress is part of the Universal Addressing Module.

Resource URL

```
http://server:port/soap/ValidateAddress
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddress">

  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressRequest>
      <val:input_port>
        <val:Address>
          <val:AddressLine1>1 N. State St.</val:AddressLine1>
          <val:City>Chicago</val:City>
          <val:StateProvince>IL</val:StateProvince>
        </val:Address>
      </val:input_port>
    </val:ValidateAddressRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ValidateAddressResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/ValidateAddress">

      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>87</ns3:Confidence>
          <ns3:RecordType>HighRise</ns3:RecordType>
          <ns3:RecordType.Default>Y</ns3:RecordType.Default>
          <ns3:CountryLevel>A</ns3:CountryLevel>
          <ns3:ProcessedBy>USA</ns3:ProcessedBy>
          <ns3:MatchScore>0</ns3:MatchScore>
          <ns3:AddressLine1>1 N State St</ns3:AddressLine1>
          <ns3:City>Chicago</ns3:City>
          <ns3:StateProvince>IL</ns3:StateProvince>
          <ns3:PostalCode>60602-3302</ns3:PostalCode>
          <ns3:PostalCode.Base>60602</ns3:PostalCode.Base>
          <ns3:PostalCode.AddOn>3302</ns3:PostalCode.AddOn>
          <ns3:Country>United States Of America</ns3:Country>
          <ns3:AdditionalInputData/>
          <ns3:user_fields/>
        </ns3:Address>
      </ns3:output_port>
    </ns3:ValidateAddressResponse>
  </soap:Body>
</soap:Envelope>
```

Request

Parameters for Input Data

ValidateAddress takes an address as input. All addresses use this format regardless of the address's country. See [Address Line Processing for U.S. Addresses](#) on page 452 for important information about how address line data is processed for U.S. addresses.

Table 121: Input Format

| Parameter | Format | Description |
|--------------|-------------|--------------------------|
| AddressLine1 | String [50] | The first address line. |
| AddressLine2 | String [50] | The second address line. |

| Parameter | Format | Description |
|---------------|-------------|---|
| AddressLine3 | String [50] | The third address line. Does not apply to Canadian addresses. |
| AddressLine4 | String [50] | The fourth address line. Does not apply to Canadian addresses. |
| AddressLine5 | String [50] | The fifth address line. Applies only to U.K. addresses. May contain street name, unit number, building number, and so on. |
| City | String [50] | The city name. For U.S. addresses only, you may put the city, state, and ZIP Code™ in the City field. If you do this, you must leave the StateProvince and PostalCode fields blank. |
| StateProvince | String [50] | The state or province. For U.S. addresses only, you may put the state in the City field instead of the StateProvince field. |
| PostalCode | String [10] | The postal code for the address in one of the following formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 For U.S. addresses only, you may put the ZIP Code™ in the City field. For U.S. addresses only, if the city/state/ZIP Code™ is in the PostalCode field, ValidateAddress may parse the data and successfully process the address. For best results, put this data in the appropriate fields (City, StateProvince, and PostalCode). |

| Parameter | Format | Description |
|-------------|-------------|---|
| Country | String [50] | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • Two-character ISO 3166-1 Alpha 2 country code • Three-character ISO 3166-1 Alpha 3 country code • English country name • French country name • German country name • Spanish country name <p>For a list of ISO codes, see ISO Country Codes and Module Support on page 1011.</p> |
| FirmName | String [50] | The company or firm name. |
| USUrbanName | String [50] | The U.S. address urbanization name. This is used primarily for Puerto Rico addresses. |
| CustomerID | String [9] | If this mailpiece uses a generic barcode, specify your USPS®-assigned customer ID in this field. The ValidateAddress generic barcode is used for mailpieces that use the OneCode ACS® service. |
| CanLanguage | String | <p>For Canadian addresses only, indicates whether the address is in English or French, if the option <code>CanFrenchFormat=T</code> is used.</p> <p>If this field is blank, the address is formatted in English. If the field contains any non-blank value, the address is formatted in French. Note that addresses in Quebec are always formatted in French regardless of the value in this field.</p> |

Address Line Processing for U.S. Addresses

The input fields AddressLine1 through AddressLine4 are handled differently for U.S. addresses depending on whether the firm name extraction or urbanization code extraction options are enabled. If either of these options is enabled, ValidateAddress will look at the data in all four fields to validate the address and extract the requested data (firm name and/or urbanization code). If neither of these options is enabled, ValidateAddress uses only the first two non-blank address line fields in its validation attempt. The data in the other address line fields is returned in the output field AdditionalInputData. For example,

AddressLine1: A1 Calle A
AddressLine2:

AddressLine3: URB Alamar

AddressLine4: Precisely

In this address, if either firm name extraction or urbanization code extraction were enabled, `ValidateAddress` would examine all four address lines. If neither firm name extraction nor urbanization code extraction were enabled, `ValidateAddress` would examine `AddressLine1` and `AddressLine3` (the first two non-blank address lines) and attempt to validate the address using that data; the data in `AddressLine4` would be returned in the output field `AdditionalInputData`.

Parameters for Options

Output Data Options

The following table lists the options that control the type of information returned by `ValidateAddress`. Some of these options can be overridden for Canadian addresses. For more information, see [Canadian Address Options](#) on page 480.

Table 122: Output Data Options

| Parameter | Description |
|------------------|--|
| OutputRecordType | <p>Type of output record. For more than one, provide a list.</p> <ul style="list-style-type: none"> <li data-bbox="686 485 1422 890">A Returns 1 to 4 lines of address data plus city, state, postal code, firm name, and urbanization name information. Each address line represents an actual line of the address as it would appear on an envelope. For more information, see Standard Address Output on page 491. If the address is validated, the address lines contain the standardized address. When addresses are standardized, punctuation is removed, directionals are abbreviated, street suffixes are abbreviated, and address elements are corrected. If the address is not validated, the address lines contain the address as it appeared in the input ("pass through" data). Non-validated addresses are always included as pass through data in the address line fields even if you do not specify <code>OutputRecordType=A</code>. <li data-bbox="686 915 1422 1129">E Parsed address elements. Each part of the address, such as house number, street name, street suffix, directionals, and so on is returned in a separate field. For more information, see Parsed Address Elements Output on page 567. Note that if you specify "E" and specify <code>OutputFormattedOnFail=Y</code>, the parsed address elements will contain the input address for addresses that could not be validated. <li data-bbox="686 1155 1422 1436">I Parsed input. This option returns the input address in parsed form regardless of whether the address is validated. Each part of the input address, such as house number, street name, street suffix, directionals, and so on is returned in a separate field. Parsed input (value "I") differs from the combination of <code>OutputRecordType=E</code> and <code>OutputFormattedOnFail=Y</code> in that "I" returns all input address in parsed form, not just input that could not be validated. For more information, see Parsed Input on page 569. <li data-bbox="686 1461 1422 1549">P Postal data. Output addresses contain additional data for each validated address. For more information, see Postal Data Output on page 497. <p>Blank Do not return any address data or postal data.</p> |

| Parameter | Description |
|-----------------------------|--|
| OutputFieldLevelReturnCodes | <p data-bbox="678 380 1425 554">Specifies whether to include field-level result indicators. Field-level result indicators describe how each address element was handled. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in HouseNumber.Result. For a complete listing of result indicator output fields, see Field-Level Result Indicators on page 504.</p> <p data-bbox="683 575 1338 646">N No, do not output field-level return codes (default). Y Yes, output field-level return codes.</p> |

| Parameter | Description |
|-----------------------|---|
| OutputFormattedOnFail | <p data-bbox="677 380 1425 491">Specifies whether to return a formatted address when an address cannot be validated. The address is formatted using the preferred address format for the address's country. If this option is not selected, the output address fields are blank when the address cannot be validated.</p> <p data-bbox="761 512 1414 569">Note: This option applies only to U.S. and Canadian addresses. Formatted data will not be returned for any other address.</p> <p data-bbox="683 600 1276 674">N No, do not format failed addresses (default). Y Yes, format failed addresses.</p> <p data-bbox="677 705 1425 982">Formatted addresses are returned using the format specified by the <code>OutputRecordType</code> option. Note that if you specify <code>OutputRecordType=E</code>, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, specify <code>OutputRecordType=I</code>.</p> <p data-bbox="677 1010 1425 1287">Formatted addresses are returned using the format specified by the <code>Option.OutputRecordType</code> option. Note that if you specify <code>Option.OutputRecordType=E</code>, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, specify <code>Option.OutputRecordType=I</code>.</p> <p data-bbox="677 1314 1425 1602">Formatted addresses are returned using the format specified by the Include a standard address, Include address line elements, and Include postal information check boxes. Note that if you select Include address line elements, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not <code>ValidateAddress</code> could validate the address, select Include standardized input address elements.</p> <p data-bbox="677 1629 1425 1656">If you specify Y, you must specify "A" and/or "E" for <code>OutputRecordType</code>.</p> <p data-bbox="677 1671 1214 1728">If you specify Y, you must specify "A" and/or "E" for <code>Option.OutputRecordType</code>.</p> <p data-bbox="677 1755 1425 1797">If you check this option, you must select Include a standard address and/or Include address line elements.</p> |

| Parameter | Description |
|-----------------------|---|
| OutputStreetNameAlias | <p data-bbox="678 380 1429 464">For U.S. addresses only, specifies how to handle street name aliases used in the input. A street alias is an alternate name for a street and applies only to a specific range of addresses on the street.</p> <p data-bbox="678 485 1429 600">If you enable this option, street name aliases used in the input will appear in the output. If you do not enable this option, street name aliases in the input will be converted to the base street name in the output, with the following exceptions:</p> <ul data-bbox="678 621 1429 737" style="list-style-type: none"><li data-bbox="678 621 1429 674">• If a preferred alias is used in input the preferred alias will always be used in output.<li data-bbox="678 684 1429 737">• Changed aliases used in input are always converted to the base street name in output. <p data-bbox="678 758 1429 852">This is one of three options that control how <code>ValidateAddress</code> handles street name aliases. The other two are <code>OutputPreferredAlias</code> and <code>OutputAbbreviatedAlias</code>.</p> <p data-bbox="760 873 1429 968">Note: If <code>OutputAbbreviatedAlias</code> is enabled, the abbreviated alias will always appear in the output even if you have <code>OutputStreetNameAlias</code> disabled.</p> <p data-bbox="678 1010 1429 1041">N No, do not return street name aliases in the output.</p> <p data-bbox="678 1062 1429 1117">Y Yes, return street name aliases in the output if the input street name is an alias (default).</p> |

| Parameter | Description |
|---------------------|--|
| OutputAddressBlocks | <p>Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p>For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p>ValidateAddress formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option <code>OutputCountryFormat</code> does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p>For addresses outside the U.S. and Canada, if ValidateAddress is unable to validate the address, no address blocks are returned. For addresses in the U.S. and Canada, address blocks are returned even if validation fails.</p> <p>N No, do not return address blocks. Default.</p> <p>Y Yes, return address blocks.</p> |

| Parameter | Description |
|------------|--|
| OutputAMAS | <p>Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p>For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p>ValidateAddress formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option <code>OutputCountryFormat</code> does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p>For addresses outside the U.S. and Canada, if ValidateAddress is unable to validate the address, no address blocks are returned. For addresses in the U.S. and Canada, address blocks are returned even if validation fails.</p> <p>N No, do not return address blocks. Default. Y Yes, return address blocks.</p> |

Obtaining Congressional Districts

ValidateAddress can determine the U.S. congressional district for an address.

To obtain congressional districts, `OutputRecordType` must contain P. For more information on `OutputRecordType`, see [Output Data Options](#) on page 452.

Table 123: Congressional District Output

| Response Element | Description |
|-------------------------|--|
| USCongressionalDistrict | Congressional district number. If the address is a non-state address (for example Puerto Rico or Washington D.C.) this field is blank. |

Obtaining County Names

ValidateAddress can determine the county where a particular address is located and return the county name.

Note: County names are available for U.S. addresses only.

To obtain county names, `OutputRecordType` must contain P. For more information on `OutputRecordType`, see [Output Data Options](#) on page 452.

Table 124: County Name Output

| Response Element | Description |
|------------------|-------------|
| USCountyName | County name |

Obtaining FIPS County Numbers

Federal Information Processing Standards (FIPS) county numbers are numbers that identify each county in a state. Note that these numbers are only unique at the state level, not the national level. For more information, see <http://www.census.gov>.

Note: FIPS county numbers are available for U.S. addresses only.

To obtain FIPS county numbers, `OutputRecordType` must contain P. For more information on `OutputRecordType`, see [Output Data Options](#) on page 452.

Table 125: FIPS County Number Output

| | Description |
|--------------------|---|
| USFIPSCountyNumber | FIPS (Federal Information Processing Standards) county number |

Obtaining Carrier Route Codes

Carrier route codes are unique identifiers assigned to each mail carrier who delivers mail, allowing unique identification of each U.S. delivery route. ValidateAddress can return the code that represents an addressee's carrier route.

Note: Carrier route codes are available for U.S. addresses only.

To obtain carrier route codes, `OutputRecordType` must contain P. For more information on `OutputRecordType`, see [Output Data Options](#) on page 452.

Table 126: Carrier Route Code Output

| Response Element | Description |
|------------------|-------------|
|------------------|-------------|

| | |
|--------------------|--------------------|
| USCarrierRouteCode | Carrier route code |
|--------------------|--------------------|

Creating Delivery Point Barcodes

A Delivery Point Barcode (DPBC) is a POSTNET™ barcode representation of the address. It consists of 62 bars with beginning and ending frame bars and five bars each for the ZIP + 4® Code, a value calculated based on the street address number, and a correction digit. The DPBC allows automated sortation of letter mail to the carrier level in walk sequence. `ValidateAddress` generates the data you need to assemble a DPBC.

Note: Delivery Point Barcodes are available for U.S. addresses only. For more information on Delivery Point Barcodes, see <http://www.usps.com>.

To generate the data needed to assemble a DPBC, `OutputRecordType` must contain P. For more information on `OutputRecordType`, see [Output Data Options](#) on page 452.

Table 127: Delivery Point Barcode Output

| | Description |
|--|-------------|
|--|-------------|

| | |
|---------------|---|
| PostalBarCode | The delivery point portion of the delivery point barcode. |
|---------------|---|

| | |
|----------------|---|
| USBCCheckDigit | Check-digit portion of the 11-digit delivery point barcode. |
|----------------|---|

To assemble a DPBC you concatenate the values found in the `ValidateAddress` output as follows:

`PostalCode.Base` + `PostalCode.Addon` + `PostalBarcode` + `USBCCheckDigit`

For example, if you have the following:

- **PostalCode.Base** = 49423
- **PostalCode.Addon** = 4506
- **PostalBarcode** = 29
- **USBCCheckDigit** = 2

The assembled barcode would be:

494234506292

Default Options

The following table lists the options that control the format and processing of addresses. These are called "default options" because by default they apply to all addresses. Some of these options can be overridden for Canadian addresses. For more information, see [Canadian Address Options](#) on page 480.

Table 128: Default Options

| Parameter | Description |
|---------------------------|--|
| OutputCasing | <p>Specifies the casing of the output address. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p style="padding-left: 40px;">123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p style="padding-left: 40px;">123 MAIN ST MYTOWN FL 12345</p> |
| OutputPostalCodeSeparator | <p>Specifies whether to use separators (spaces or hyphens) in ZIP™ Codes or Canadian postal codes.</p> <p>For example, a ZIP + 4® Code with the separator would be 20706-1844 and without the separator it would be 207061844. A Canadian postal code with the separator would be P5E"1S7 and without the separator it would be P5E1S7.</p> <p>Y Yes, use separator (default).</p> <p>N No, do not use separator.</p> <p>Note: Spaces are used in Canadian postal codes and hyphens in U.S. ZIP + 4® Codes.</p> |

| Parameter | Description |
|-------------------------------|---|
| OutputMultinationalCharacters | <p>Specifies whether or not to return multinational characters, including diacritical marks such as umlauts or accents. (Not supported for U.S. addresses).</p> <p>N No, do not use multinational characters in the output (default). Only standard ASCII characters is returned.</p> <p>Y Yes, use multinational characters in the output.</p> |
| KeepMultimatch | <p>Indicates whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> <p>For more information, see Returning Multiple Matches on page 466.</p> |
| StandardAddressFormat | <p>Specifies where to place secondary address information for U.S. addresses. Secondary address information refers to apartment numbers, suite numbers, and similar designators. For example, in this address the secondary address information is "Apt 10E" and the primary address information is "424 Washington Blvd".</p> <p>Apt 10E 424 Washington Blvd Springfield MI 49423</p> <p>C Place both primary and secondary address information in AddressLine1 (default).</p> <p>S Place the primary address information in AddressLine1 and the secondary address information in AddressLine2.</p> <p>D Place both primary and secondary address information in AddressLine1 and place dropped information from dual addresses in AddressLine2. A dual address is an address that contains both street information and PO Box/Rural Route/Highway Contract information. For more information, see About Dual Address Logic on page 465.</p> |

| Parameter | Description |
|---------------------|--|
| OutputShortCityName | <p>Specifies how to format city names that have short city name or non-mailing city name alternatives. Applies to U.S. and Canadian addresses.</p> <ul style="list-style-type: none"> Y Returns the USPS®-approved abbreviation for the city, if there is one. The USPS® provides abbreviations for city names that are 14 characters long or longer. City abbreviations are 13 characters or less and can be used when there is limited space on the mailing label. If there is no short city name for the city, then the full city name is returned. N Returns the long city name (default). S Returns the abbreviated city name only if an abbreviated city name is used in the input address. If the input address does not use a short city name, either the long or short city name could be returned, depending on USPS® regulations for the particular city. Select this option if you are performing a CASS™ test. V Output the non-mailing city name (the vanity name) if the input city name is a non-mailing city name. For example, "Hollywood" is a non-mailing city name for "Los Angeles". If you do not select this option and the input city name is a non-mailing city name the long version of the mailing city is returned. |
| OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <ul style="list-style-type: none"> E Use English country names (default). S Use Spanish country names. F Use French country names. G Use German country names. I Use two-letter ISO abbreviation for the countries instead of country names. U Use Universal Postal Union abbreviation for the countries instead of country names. |

| Parameter | Description |
|-------------|---|
| HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Canada, specify Canada. ValidateAddress uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Samoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equatorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent and the Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Suriname, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |

| Parameter | Description |
|------------------|---|
| DualAddressLogic | <p>Indicates how to return a match if multiple non-blank address lines are present or multiple address types are on the same address line. (U.S. addresses only.)</p> <p>N (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority:</p> <ol style="list-style-type: none"> 1. PO Box 2. Firm 3. Highrise 4. Street 5. Rural Route 6. General Delivery <p>S Return a street match, regardless of the address line.</p> <p>P Return a PO Box match, regardless of the address line.</p> <p>For more information, see About Dual Address Logic on page 465.</p> |

About Dual Address Logic

For U.S. addresses only, the `DualAddressLogic` option controls whether `ValidateAddress` should return a street match or a PO Box/Rural Route/Highway Contract match when the address contains both street and PO Box/Rural Route/Highway Contract information in the same address line.

Note: The `DualAddressLogic` option has no effect if the street information is in a different address line input field than the PO Box/Rural Route/Highway Contract information.

For example, given the following input address:

AddressLine1: 401 N Main St Apt 1 POB 1
City: Kemp
StateProvince: TX
PostalCode: 75143

`ValidateAddress` would return one of the following:

- If `DualAddressLogic` is set to either N or P:

AddressLine1: PO Box 1
City: Kemp
StateProvince: TX
PostalCode: 75143-0001

- If `DualAddressLogic` is set to S:

AddressLine1: 401 N Main St Apt 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-4806

The address data that is not used to standardize the address can be returned in one of two places:

- **AddressLine2**—The address information not used to standardize the address is returned in the **AddressLine2** field if you specify `StandardAddressFormat=D`. For more information, see [Default Options](#) on page 461. For example, if you choose to return a street match for dual addresses,

AddressLine1: 401 N Main St Apt 1
 AddressLine2: PO Box 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-0001

- **AdditionalInputData**—If you do not specify `StandardAddressFormat=D` then the address information not used to standardize the address is returned in the **AdditionalInputData** field. For more information on this option, see [Default Options](#) on page 461. For example, if you choose to return a street match for dual addresses,

AddressLine1: 401 N Main St Apt 1
 City: Kemp
 StateProvince: TX
 PostalCode: 75143-0001
 AdditionalInputData: PO Box 1

Address information that is dropped can be retrieved by setting the `StandardAddressFormat` option to D. For more information, see [Default Options](#) on page 461 .

Returning Multiple Matches

If `ValidateAddress` finds multiple address in the postal database that are possible matches for the input address, you can have `ValidateAddress` return the possible matches. For example, the following address matches multiple addresses in the U.S. postal database:

PO BOX 1
 New York, NY

Options

To return multiple matches, use the options described in the following table.

Table 129: Multiple Match Option

| Parameter | Description |
|-----------------------------|---|
| KeepMultimatch | <p>Indicates whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> |
| MaximumResults | <p>A number between 1 and 10 that indicates the maximum number of addresses to return.</p> <p>The default value is 1.</p> <p>Note: The difference between Keepmultimatch=N and KeepMultimatch=Y/MaximumResults=1 is that a multiple match will return a failure if KeepMultimatch=N, whereas a multiple match will return one record if KeepMultimatch=Y and MaximumResults=1.</p> |
| OutputFieldLevelReturnCodes | <p>To identify which output addresses are candidate addresses, you must specify a value of Y for OutputFieldLevelReturnCodes. When you do this, records that are candidate addresses will have one or more "M" values in the field-level result indicators.</p> |

Output

When you choose to return multiple matches, the addresses are returned in the address format you specify. For information on specifying address format, see [Output Data Options](#) on page 452. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information, see [Field-Level Result Indicators](#) on page 504.

U.S. Address Options

| Parameter | Description |
|---------------------|---|
| PerformUSProcessing | <p>Specifies whether to process U.S. addresses. If you enable U.S. address processing ValidateAddress will attempt to validate U.S. addresses. If you disable U.S. address processing, U.S. addresses will fail, meaning they are returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for U.S. address processing you must disable U.S. address processing in order for your jobs to complete successfully, regardless of whether or not they contain U.S. addresses.</p> <p>Note: You must have a valid license for U.S. address processing to successfully process U.S. addresses. If you enable U.S. address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>N No, do not process U.S. addresses.</p> <p>Y Yes, process U.S. addresses. Default.</p> |
| Database.US | <p>Specifies which database to use for validating U.S. addresses. Only databases that have been defined in the US Database Resources panel in the Management Console are available.</p> |
| PerformLOT | <p>Enhanced Line of Travel (eLOT) processing assigns a Line of Travel sequence code to your addresses. Note that addresses are not sorted into eLOT sequence but they are assigned a Line of Travel sequence code that allows you to sort addresses into eLOT sequence.</p> <p>To perform eLOT processing you must have the eLOT database installed.</p> <p>N No, do not perform Line of Travel Processing. Default.</p> <p>Y Yes, perform Line of Travel processing.</p> <p>For a listing of the output fields returned by this option, see Enhanced Line of Travel Output on page 518.</p> |

| Parameter | Description |
|------------|---|
| PerformRDI | <p>Residential Delivery Indicator (RDI™) processing checks if an address is a residential address (not a business address). To perform RDI™ processing, you must have the RDI™ database installed.</p> <p>If you enable both DPV® and RDI™ processing, RDI™ information is only returned if the address is a valid delivery point. If DPV® does not validate the address no RDI™ data is returned.</p> <p>N No, do not perform Residential Delivery Indicator processing. Default.</p> <p>Y Yes, perform Residential Delivery Indicator processing.</p> |
| PerformESM | <p>Enhanced Street Matching (ESM) applies additional matching logic to correct misspelled or complex street names and obtain a match. ESM enables more addresses to be validated but it reduces performance. You cannot perform ESM when ASM is enabled.</p> <p>N No, do not perform enhanced street matching. Default.</p> <p>Y Yes, perform enhanced street matching.</p> |
| PerformASM | <p>All Street Matching (ASM) applies ESM processing as well as additional matching logic to correct errors in street names and obtain a match. It is effective at matching streets when the first letter of the street is incorrect. ASM provides the best address validation but reduces performance.</p> <p>N No, do not perform all street matching.</p> <p>Y Yes, perform all street matching. Default.</p> |
| PerformDPV | <p>Delivery Point Validation (DPV®) validates that a specific address exists, as opposed to validating that a specific address is within a range of valid addresses. CMRA processing checks if an address is for a mailbox rented from a private company, referred to as a Commercial Mail Receiving Agent (CMRA).</p> <p>To perform DPV and CMRA processing, you must have the DPV database installed. The DPV database contains both DPV and CMRA data.</p> <p>N No, do not perform Delivery Point Validation or CMRA processing. Default.</p> <p>Y Yes, perform Delivery Point Validation and CMRA processing.</p> <p>For a listing of the output fields returned by this option, see DPV and CMRA Output on page 521.</p> |

| Parameter | Description |
|-----------------|---|
| PerformLACSLink | <p>The USPS® Locatable Address Conversion System (LACS) allows you to correct addresses that have changed as a result of a rural route address converting to street-style address, a PO Box renumbering, or a street-style address changing. When enabled, LACS^{Link} processing is attempted for addresses that could not be validated, or addresses were validated and flagged for LACS^{Link} conversion.</p> <p>To perform LACS^{Link} processing, you must have the LACS^{Link} database installed.</p> <p>N No, do not attempt LACS^{Link} conversion. Default.</p> <p>Y Yes, attempt LACS^{Link} conversion.</p> <p>For a listing of the output fields returned by this option, see LACSLink Output on page 519</p> |
| PerformEWS | <p>The Early Warning System (EWS) uses the USPS® EWS File to validate addresses that are not in the ZIP + 4® database.</p> <p>To perform EWS processing, you must have the EWS database installed.</p> <p>If an input address matches an address in the EWS file, the following record-level result indicators are returned:</p> <ul style="list-style-type: none"> • Status="F" • Status.Code="EWSFailure" • Status.Description="Address found in EWS table" <p>N No, do not perform EWS processing. Default.</p> <p>Y Yes, perform EWS processing.</p> |

| Parameter | Description |
|-----------|-------------|
|-----------|-------------|

| | |
|-------------|--|
| ExtractFirm | |
|-------------|--|

| Parameter | Description |
|-----------|--|
| Y | Yes, extract the firm name. |
| N | No, do not extract the firm name. Default. |
| | <p>To identify firm names in address lines, the address lines are scanned for keywords and patterns that identify which fields are address lines and which are FirmName lines. Since this is done based on patterns, fields may be misidentified. The following tips can help ensure optimal firm extraction:</p> <ul style="list-style-type: none"> • If possible, place the primary address elements in AddressLine1, the secondary elements in AddressLine2, Urbanization in AddressLine3, and firm in AddressLine4. If the address has no urbanization code, then place the firm name in AddressLine3 and leave AddressLine4 blank. For example, <p data-bbox="570 852 984 879">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 884 846 911">AddressLine2: Suite 600</p> <p data-bbox="570 915 841 942">AddressLine3: Precisely</p> <p data-bbox="570 947 829 974">AddressLine4: <blank></p> • When you define just two address lines, AddressLine2 is assigned to the secondary address most of the time. If you want to increase the chance that AddressLine2 will be treated as a firm name, put the firm name in AddressLine3 and leave AddressLine2 blank. • Numbers in a firm name (such as the "1" in "1 Stop Software") will increase the likelihood that the field will be treated as an address line. <p>Here are some examples of firm name extraction:</p> <ul style="list-style-type: none"> • In this example, AddressLine2 would get extracted into the FirmName output field <p data-bbox="570 1293 789 1320">FirmName: <blank></p> <p data-bbox="570 1325 1089 1352">AddressLine1: 4200 Parliament Place Suite 600</p> <p data-bbox="570 1356 1094 1383">AddressLine2: International Goose Feathers inc.</p> • In this example, AddressLine3 would get extracted into the FirmName output field. <p data-bbox="570 1457 789 1484">FirmName: <blank></p> <p data-bbox="570 1488 984 1516">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 1520 846 1547">AddressLine2: Suite 600</p> <p data-bbox="570 1551 841 1579">AddressLine3: Precisely</p> • In this example, AddressLine3 would be placed in the AdditionalInputData output field. The firm name would not be extracted because the FirmName input field is not blank. <p data-bbox="570 1705 1057 1732">FirmName: International Goose Feathers Inc.</p> <p data-bbox="570 1736 984 1764">AddressLine1: 4200 Parliament Place</p> <p data-bbox="570 1768 846 1795">AddressLine2: Suite 600</p> <p data-bbox="570 1799 841 1827">AddressLine3: Precisely</p> • In this example, no firm name would be extracted because there is only one |

| Parameter | Description |
|------------|--|
| | <p>non-blank address line, which is always treated as the primary address element.</p> <p>FirmName: <blank> AddressLine1: 4200 Parliament Place Suite 600</p> <ul style="list-style-type: none"> In this example, AddressLine2 would be treated as a secondary address element because the numeral "1" causes that field to be treated as a secondary address element. <p>FirmName: <blank> AddressLine1: 4200 Parliament Place Suite 600 AddressLine2: 1 Stop Software</p> |
| ExtractUrb | <p>Specifies whether to extract the urbanization name from AddressLine1 through AddressLine4 and place it in the USUrbanName output field. This option works in cases where the input record's USUrbanName field is blank and there is more than one address line.</p> <p>Y Yes, extract the urbanization name.</p> <p>N No, do not extract the urbanization name. Default.</p> <p>To identify urbanization names, the address lines are scanned for keywords and patterns that identify which fields are address lines and which are urbanization name lines. Since this is done based on patterns, it is possible for fields to be incorrectly identified. To help ensure optimal urbanization extraction, place the primary address elements in AddressLine1, the secondary elements in AddressLine2, Urbanization in AddressLine3, and firm in AddressLine4, if possible. For example,</p> <p>AddressLine1: A1 Calle A AddressLine2: AddressLine3: URB Alamar AddressLine4: Precisely</p> |

| Parameter | Description |
|------------------|--|
| PerformSuiteLink | <p data-bbox="548 380 1089 407">Specifies whether to perform Suite^{Link™} processing.</p> <p data-bbox="548 422 1425 541">Suite^{Link} corrects secondary address information for U.S. business addresses whose secondary address information could not be validated. If Suite^{Link} processing is enabled, the firm name is matched to a database of known firm names and their secondary address information.</p> <p data-bbox="548 556 688 583">For example,</p> <p data-bbox="548 604 964 724">Firm Name: Precisely Address Line 1: 4200 Parliament Place Address Line 2: STE 1 Postal Code: 20706</p> <p data-bbox="548 739 1317 766">In this case, Suite^{Link} processing would provide the correct suite number:</p> <p data-bbox="548 793 924 913">Firm Name: Precisely Address Line 1: 4200 Parliament Pl Address Line 2: STE 500 Postal Code: 20706-1844</p> <p data-bbox="548 928 1414 955">To perform Suite^{Link™} processing, you must have the Suite^{Link™} database installed.</p> <p data-bbox="548 976 1024 1003">This option takes one of the following values:</p> <p data-bbox="558 1018 1076 1098">N No, do not use Suite^{Link™}. Default. Y Yes, use Suite^{Link™} processing.</p> <p data-bbox="548 1123 1401 1150">For a listing of fields returned by this option, see SuiteLink Output on page 523.</p> |

| Parameter | Description |
|----------------------|---|
| OutputPreferredAlias | <p data-bbox="548 380 1214 407">Specifies whether to use a street's preferred alias in the output.</p> <p data-bbox="548 426 1425 485">Street name aliases in the United States are alternative names given to sections of a street. There are four types of street name aliases:</p> <ul data-bbox="548 504 1425 840" style="list-style-type: none"> <li data-bbox="548 504 1425 562">• Preferred—A preferred alias is the street name preferred locally. It typically applies only to a specific range of addresses on the street. <li data-bbox="548 569 1425 682">• Abbreviated—An abbreviated alias is a variation of the street name that can be used in cases where the length of AddressLine1 is longer than 31 characters. For example, the street name 1234 BERKSHIRE VALLEY RD APT 312A could be abbreviated to 1234 BERKSHIRE VLLY RD APT 312A. <li data-bbox="548 688 1425 781">• Changed—There has been an official street name change and the alias reflects the new name. For example if SHINGLE BROOK RD is changed to CANNING DR, then CANNING DR would be a changed alias type. <li data-bbox="548 787 1425 840">• Other—The street alias is made up of other names for the street or common abbreviations of the street. <p data-bbox="548 863 1308 890">The non-alias version of the street name is called the base street name.</p> <p data-bbox="548 909 1401 968">If the preferred alias is used in the input then the preferred alias will be the street name in the output regardless of whether you enable this option.</p> <p data-bbox="548 987 1414 1045">This is one of three options that control how ValidateAddress handles street name aliases. The other two are OutputStreetNameAlias and OutputAbbreviatedAlias.</p> <p data-bbox="548 1064 1425 1203">In most cases, if you select both OutputPreferredAlias and OutputAbbreviatedAlias, and ValidateAddress finds both a preferred and an abbreviated alias in the postal database, the abbreviated alias will be used in the output. The exception to this rule is if the input street name is a preferred alias. In this case, the preferred alias will be used in the output.</p> <p data-bbox="548 1222 1114 1249">Y Yes, perform preferred alias processing.</p> <p data-bbox="548 1268 1268 1295">N No, do not perform preferred alias processing. Default.</p> <p data-bbox="631 1314 1414 1400">Note: If the input address contains a street name alias of type "changed" the output address will always contain the base street name regardless of the options you specify.</p> |

| Parameter | Description |
|------------------------|---|
| OutputAbbreviatedAlias | <p>Specifies whether to use a street's abbreviated alias in the output if the output address line is longer than 31 characters.</p> <p>This is one of three options that control how ValidateAddress handles street name aliases. The other two are OutputStreetNameAlias and OutputPreferredAlias.</p> <p>Note: If a preferred alias is specified in the input, the output street name will always be the preferred alias, even if you enable abbreviated street name alias processing.</p> <p>Y Yes, perform abbreviated alias processing.</p> <p>N No, do not perform abbreviated alias processing. Default.</p> <p>Note: If the input address contains a street name alias of type "changed" the output address will always contain the base street name regardless of the options you specify.</p> |
| DPVDetermineNoStat | <p>Determines the "no stat" status of an address. An address is considered "no stat" if it exists but cannot receive mail, and therefore is not counted as a delivery statistic on a carrier's route (hence the term "no stat"). Examples include buildings under construction or those that the letter carrier has identified as not likely to receive mail.</p> <p>N No, do not determine "no stat" status. Default.</p> <p>Y Yes, determine "no stat" status.</p> <p>Note: You must enable DPV processing to use this option.</p> <p>The result is returned in the DPVNoStat field. For more information see LACSLink Output on page 519</p> |
| DPVDetermineVacancy | <p>Determines if the location has been unoccupied for at least 90 days.</p> <p>N No, do not determine vacancy. Default.</p> <p>Y Yes, determine vacancy.</p> <p>Note: You must enable DPV processing to use this option.</p> <p>The result is returned in the DPVVacant field. For more information see LACSLink Output on page 519</p> |
| ReturnVerimove | <p>Returns VeriMove detail data in output.</p> <p>N No, do not return VeriMove detail data. Default.</p> <p>Y Yes, return VeriMove detail data.</p> |

| Parameter | Description |
|---------------------------------|--|
| SuppressZplusPhantomCarrierR777 | <p>Specifies whether to suppress addresses with Carrier Route R777. These addresses are phantom routes and are not eligible for street delivery. Since these addresses are assigned a ZIP + 4[®] code by the USPS[®], Validate Address marks these addresses as deliverable. Select this option if you do not want addresses with Carrier Route R777 marked as deliverable. This will cause the following actions:</p> <ul style="list-style-type: none"> • No ZIP + 4 code is assigned • Address is not counted on the USPS Form 3553 (CASS Summary Report) • DPV Footnote of R7 is returned <p>N No, do not suppress addresses with Carrier Route R777.</p> <p>Y Yes, suppress addresses with Carrier Route R777.</p> |
| StreetMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input street name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| FirmMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input firm name must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium" (default).</p> <p>L The matching algorithm is "loose."</p> |
| DirectionalMatchingStrictness | <p>Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following:</p> <p>E The input directionals, such as the "N" in 123 N Main St., must match the database exactly.</p> <p>T The matching algorithm is "tight."</p> <p>M The matching algorithm is "medium". Default.</p> <p>L The matching algorithm is "loose."</p> |

| Parameter | Description |
|------------------------------|---|
| DPVSuccessfulStatusCondition | <p>Select the match condition where a DPV result does NOT cause a record to fail.</p> <p>F Full match</p> <p>P Partial match</p> <p>A Always. Default.</p> <p>Note: You must enable DPV processing to use this option.</p> |
| FailOnCMRAMatch | <p>Treat Commercial Mail Receiving Agency (CMRA) matches as failures?</p> <p>N No, do not treat CMRA matches as failures. Default.</p> <p>Y Yes, treat CMRA matches as failures.</p> <p>Note: You must enable DPV processing to use this option.</p> |
| StandardAddressPMBLine | <p>Specifies where Private Mailbox (PMB) information is placed.</p> <p>N Do not include the PMB information in Standard Address output (default).</p> <p>1 Place the PMB information in AddressLine1. If you specify 1, you must set StandardAddressFormat to either C or D.</p> <p>2 Place the PMB information in AddressLine2.</p> |
| PreferredCity | <p>Specifies whether the preferred last line city name should be stored.</p> <p>Z Store the Preferred Last Line City Name from the USPS ZIP+4 File (Override City Name).</p> <p>Note: If you select this option, Validate Address generates a CASS-certified configuration and the USPS 3553 Report.</p> <p>C Store the USPS-preferred City Name from USPS City/State File.</p> <p>Note: If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.</p> <p>P Store the Primary City Name from the USPS City/State File.</p> <p>Note: If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.</p> |

CASS Certified Processing

CASS Certified™ processing also generates the USPS CASS Detailed Report, which contains some of the same information as the 3553 report but provides much greater detail about DPV, LACS, and SuiteLink statistics. The USPS CASS Detailed Report is not required for postal discounts and does not need to be submitted with your mailing.

1. Validate Address must be in CASS Certified™ mode. If **(Not CASS Certified)** appears at the top of the window, click the **Enable CASS** button. The **Enforce CASS rules** check box will appear.
2. Click **Configure CASS 3553**. The **CASS Report Fields** dialog box appears.
3. Type the **List Processor** company name, **List Name or ID#**, and the **Number of Lists** being processed for this job.
4. Type the **Mailer Name, Address, and City, State, ZIP**.
5. Click **OK**.

The List information will appear in Section B and the Mailer information in Section D of the generated USPS® CASS Form 3553.

6. In Enterprise Designer, drag **CASS3553** from the Reports pallet to the canvas.
7. Double-click the **CASS3553** icon on the canvas.
8. On the **Stages** tab, check the **Validate Address** check box. Note that if you have renamed the Validate Address stage to something else, you should check the box with the name you have given the address validation stage.
9. On the **Parameters** tab, select the format for the report. You can create the report in PDF, HTML, or plain text format.
10. Click **OK**.
11. Repeat steps 6-10 for **CASSDetail** if you want to produce the CASS Detail Report.

Canadian Address Options

| Parameter | Description |
|---------------------------|---|
| PerformCanadianProcessing | <p>Specifies whether to process Canadian addresses. If you enable Canadian address processing ValidateAddress will attempt to validate Canadian addresses. If you disable Canadian address processing, Canadian addresses will fail, meaning they is returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for Canadian address processing you must disable Canadian address processing in order for your jobs to complete successfully, regardless of whether or not they contain Canadian addresses.</p> <p>Note: You must have a valid license for Canadian address processing to successfully process Canadian addresses. If you enable Canadian address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>N No, do not process Canadian addresses.</p> <p>Y Yes, process Canadian addresses (default).</p> |
| Database.Canada | <p>Specifies which database you want to use for validating Canadian addresses. To specify a database for Canadian address validation, select a database in the Database drop-down list. Only databases that have been defined in the CAN Database Resources panel in the Management Console are available.</p> |

| Parameter | Description |
|--------------------------|--|
| CanFrenchFormat | <p>Specifies how to determine the language (English or French) to use to format the address and directional. The following example shows an address formatted in English and French:</p> <p>English: 123 Main St W French: 123 Rue Main O</p> <p>The parameter controls the formatting of the address. It also affects the spelling of the directional but not spelling of the suffix.</p> <ul style="list-style-type: none"> C Use the street suffix returned by the matching process to determine the language. The street suffix returned by the matching process, which is used internally by <code>ValidateAddress</code> during processing, may be different from that in the input address. Ambiguous records are formatted like the input. Default. All addresses in Quebec are formatted using French. S Use the Canadian database to determine the language. The Canadian database contains data from the Canada Post Corporation (CPC). All addresses in Quebec are formatted using French. T Use the <code>CanLanguage</code> input field to determine the language. If there is a non-blank value in this field the address are formatted using French. |
| CanEnglishApartmentLabel | <p>For English addresses, specifies the default apartment label to use in the output if there is no apartment label in the input address. This setting is ignored if you specify <code>CanStandardAddressFormat=F</code>.</p> <ul style="list-style-type: none"> Apt Use "Apt" as the label. Default. Apartment Use "Apartment" as the label. Suite Use "Suite" as the label. Unit Use "Unit" as the label. |

| Parameter | Description |
|-------------------------|---|
| CanFrenchApartmentLabel | <p>For French addresses, specifies the default apartment label to use in the output if there is no apartment label in the input address. This setting is ignored if you specify <code>CanStandardAddressFormat=F</code>.</p> <p>App Use "App" as the label. Default.</p> <p>Appartement Use "Appartement" as the label.</p> <p>Bureau Use "Bureau" as the label.</p> <p>Suite Use "Suite" as the label.</p> <p>Unite Use "Unite" as the label.</p> |
| ForceCorrectionLVR | <p>Changes the civic and/or suite information to match the Large Volume Receiver (LVR) or single-single record (used when there is only one record for that postal code/street name/street type).</p> <p>N Do not change the civic and/or suite information to match the LVR or single-single record. The LVR record will be marked as a valid but non-correctable record (VN). The single-single record will be corrected, if possible, or processed as a non-correctable record..</p> <p>Y Change the civic and/or suite information to match the LVR or single-single record.</p> <p>Note: If you check this box, the Statement of Address Accuracy will not be printed because this is not a SERP-recognized setting.</p> |
| CanPreferHouseNum | <p>In cases where the house number and postal code are both valid but in conflict, you can force the postal code to be corrected based on the house number by specifying <code>CanPreferHouseNum=Y</code>. If you do not select this option the house number is changed to match the postal code.</p> <p>N Change the house number to match the postal code. Default.</p> <p>Y Change the postal code to match the house number.</p> |

| Parameter | Description |
|--------------------------|---|
| CanOutputCityAlias | <p>Specifies whether or not to return the city alias when the alias is in the input address. This option is disabled when you specify <code>CanOutputCityFormat=D</code>.</p> <p>Y Output the city alias when the city alias is in the input. Default.</p> <p>N Never output the city alias even if it is in the input.</p> |
| CanNonCivicFormat | <p>Specifies whether or not non-civic keywords are abbreviated in the output. For example, Post Office Box vs. PO Box.</p> <p>A Abbreviate non-civic keywords. Default.</p> <p>F Do not abbreviate non-civic keywords. The full keyword is used.</p> |
| EnableSERP | <p>Specifies whether or not to use SERP options.</p> <p>Y Enable SERP options.</p> <p>N Do not enable SERP options. Default.</p> |
| CanStandardAddressFormat | <p>Specifies where to place secondary address information in the output address. Secondary address information refers to apartment numbers, suite numbers, and similar designators.</p> <p>D Place apartment information in the location specified in the Default.</p> <p>B Place apartment information at the at the end of the AddressLine1 field.</p> <p>F Place the apartment number only (no label) at the beginning of the AddressLine1 field. For example, 400-123 Rue Main</p> <p>E Place the apartment number and label at the beginning of the AddressLine1 field. For example, Apt 400 123 Rue Main</p> <p>S Place apartment information on a separate line.</p> <p>S Place apartment information in the same location as the input address.</p> |

| Parameter | Description |
|---------------------|--|
| CanOutputCityFormat | <p>Specifies whether to use the long, medium, or short version of the city if the city has a long name. For example,</p> <p>Long: BUFFALO HEAD PRAIRIE Medium: BUFFALO-HEAD-PR Short: BUFFALO-HD-PR</p> <p>D Use the default option specified by the <code>OutputShortCityName</code> option. Default. If you specify <code>OutputShortCityName=V</code>, the city is formatted as if you select for this option (see below) and <code>Y</code> for CanOutputCityAlias.</p> <p>S Output short city name.</p> <p>L Output the long city name.</p> <p>M Output the medium city name.</p> <p>I Use the same city format as used in the input address. Output is L, M, or S.</p> |
| CanRuralRouteFormat | <p>Specifies where to place rural route delivery information. An example of an address with rural route delivery information is:</p> <p>36 GRANT RD RR 3 ANTIGONISH NS</p> <p>In this address, "RR 3" is the rural route delivery information.</p> <p>A Place rural route delivery information on the same line as the address, after the address information. Default. For example,</p> <p>36 GRANT RD RR 3</p> <p>S Place rural route delivery information on a separate address line. For example,</p> <p>36 GRANT RD RR 3</p> |

| Parameter | Description |
|-------------------------|---|
| CanDeliveryOfficeFormat | <p data-bbox="805 380 1425 436">Specifies where to place station information. An example of an address with station information is:</p> <p data-bbox="805 457 1032 514">PO BOX 8625 STN A ST. JOHN'S NL</p> <ul data-bbox="805 535 1425 898" style="list-style-type: none"><li data-bbox="805 535 1425 592">I Place station information in the same location as it is in the input address. Default.<li data-bbox="805 613 1425 724">A Place station information on the same line as the address, after the address information. For example, PO BOX 8625 STN A<li data-bbox="805 745 1425 898">S Place station information on a separate address line. For example, PO BOX 8625 STN A |

| Parameter | Description |
|--------------------------------------|---|
| CanDualAddressLogic | <p>Specifies whether ValidateAddress should return a street match or a PO Box/non-civic match when the address contains both civic and non-civic information. One of the following:</p> <ul style="list-style-type: none"> D Use DualAddressLogic Global Option. Default. P Match to PO Box or other non-street data. S Match to street. <p>For example, given the following input address:</p> <p>AddressLine1: 36 GRANT RD AddressLine2: RR 4 City: ANTIGONISH StateProvince: NS</p> <p>ValidateAddress would return one of the following:</p> <ul style="list-style-type: none"> • If CanDualAddressLogic is set to S, ValidateAddress returns the following: <p>AddressLine1: 36 GRANT RD AddressLine2: RR 3 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L1</p> • If CanDualAddressLogic is set to P, ValidateAddress returns the following: <p>AddressLine1: RR 4 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L2</p> <p>The address data that is not used to standardize the address is returned in the AdditionalInputData field. For more information, see Output Data Options on page 452.</p> |
| canSwitchManagedPostalCodeConfidence | <p>Select this check-box to lower the output Confidence score of the record. This is useful for the non-correctable (VN) records. You can identify such records by the DefaultValidPostalCode value. It is Y for the VN type of records and blank for others.</p> |

SERP Processing

1. Validate Address must be in SERP Certified™ mode. If **(Not SERP Certified)** appears at the top of the window, click the **Enable SERP settings** button. The **Configure SERP** box will appear.
2. Click **Configure SERP**. The **SERP Report Fields** dialog box appears.
3. Type your merchant **CPC number**.
4. Type the mailer **Name, Address, and City, State, ZIP**.
5. Click **OK**.
6. In Enterprise Designer, drag **SERPReport** from the Reports pallet to the canvas.
7. Double-click the **SERPReport** icon on the canvas.
8. On the **Stages** tab, ensure that the **Validate Address** check box is checked. Note that if you have renamed the Validate Address stage to something else, you should check the box with the name you have given the address validation stage.
9. On the **Parameters** tab, select the format for the report. You can create the report in PDF, HTML, or plain text format. PDF format is the default.
10. Click **OK**.

Obtaining SERP Return Codes

SERP return codes indicate the quality of the input address as determined by the Canada Post's Software Evaluation and Recognition Program regulations.

To obtain SERP return codes, specify OutputRecordType=P. For more information on OutputRecordType, see [Output Data Options](#) on page 452.

SERP return codes are provided in the following output field.

Table 130: SERP Return Code Output

| Response Element | Description |
|------------------|--|
| CanadianSERPCode | <p>Validation/correction return code (Canadian addresses only):</p> <p>V The input was valid. Canada Post defines a "valid" address as an address that meets all the following requirements:</p> <p style="padding-left: 40px;">Note: There are exceptions. For further information, contact the CPC.</p> <ul style="list-style-type: none"> • The address must contain all required components as found in CPC's Postal Code Data Files. • The address must provide an exact match on all components for only one address in CPC's Postal Code Data Files, allowing for acceptable alternate words and names listed in the CPC Postal Code Data Files. • Address components must be in a form that allows recognition without ambiguity. Certain components may require "qualifiers" to identify them. For instance, a Route Service address requires the key words "Rural Route" or "RR" for differentiation from a "Suburban Service" or "SS" address with the same number. <p>I The input was invalid. An "invalid" address is one that does not meet CPC requirements for a valid address (see above). Examples of this include address components that are missing, invalid, or inconsistent.</p> <p>C The input was correctable. A "correctable" address is one that can be corrected to match one, and only one, address.</p> <p>N The input was non-correctable. A "non-correctable" address is one that could be corrected a number of different ways such that ValidateAddress cannot identify a single correct version.</p> <p>F The input address was foreign (outside of Canada).</p> |

International Address Options

Addresses outside of the U.S. and Canada are referred to as "international" addresses. The following options control international address processing:

| Parameter | Description |
|--------------------------------|--|
| PerformInternationalProcessing | <p>Specifies whether to process international addresses (addresses outside the U.S. and Canada). If you enable international address processing ValidateAddress will attempt to validate international addresses. If you disable international address processing, international addresses will fail, meaning they is returned with an "F" in the Status output field. The output field Status.Code will say "DisabledCoder." If you are not licensed for international address processing you must disable international address processing in order for your jobs to complete successfully, regardless of whether or not they contain international addresses.</p> <p>Note: You must have a valid license for international address processing to successfully process international addresses. If you enable international address processing but are not licensed for this feature, or your license has expired, you will receive an error.</p> <p>N No, do not process international addresses.</p> <p>Y Yes, process international addresses (default).</p> |
| Database.International | <p>Specifies which database you want to use for validating international addresses. To specify a database for international address validation, select a database in the Database drop-down list. Only databases that have been defined in the INTL Database Resources panel in the Management Console are available.</p> |

| Parameter | Description |
|----------------------------------|---|
| InternationalCityStreetSearching | <p>By default, ValidateAddress provides a balance of good address matching accuracy with good performance. If you are willing to trade matching accuracy for faster performance, use the InternationalCityStreetSearching option to increase processing speed. When you do this, some accuracy is lost. This option only controls performance for addresses outside the U.S. and Canada. This setting affects a small percentage of records, mostly addresses in the U.K. There is no performance control for U.S. and Canadian address processing.</p> <p>If you use GetCandidateAddresses, the candidate addresses returned by GetCandidateAddresses may differ from the multiple matches returned by ValidateAddress if you set the performance tuning option for international addresses to any value other than 100.</p> <p>To control performance, specify a value from 0 to 100. A setting of 100 maximizes accuracy while a setting of 0 maximizes speed. The default is 100.</p> |
| AddressLineSearchOnFail | <p>This option enables ValidateAddress to search the AddressLine input fields for the city, state/province, postal code, and country when the address cannot be matched using the values in the City, StateProvince, and PostalCode input fields.</p> <p>Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields.</p> <p>Consider disabling this option if your input addresses use the City, State/Province and PostalCode fields. If you enable this option and these fields are used, there is an increased possibility that ValidateAddress will fail to correct values in these fields (for example a misspelled city name).</p> <p>N No, do not search the AddressLine fields.</p> <p>Y Yes, search the address line fields. Default.</p> |

Response

The output from ValidateAddress contains different information depending on the output categories you select.

Standard Address Output

Standard address output consists of four lines of the address which correspond to how the address would appear on an address label. City, state/province, postal code, and other data is also included in standard address output. Standard address output is returned for validated addresses if you set `OutputRecordType=A`. Standard address fields are always returned for addresses that could not be validated. For non-validated addresses, the standard address output fields contain the address as it appeared in the input ("pass through" data). If you want addresses to be standardized according to postal authority standards when validation fails, specify `OutputFormattedOnFail=Y` in your request.

Table 131: Standard Address Output

| Response Element | Description |
|---------------------|--|
| AdditionalInputData | Input data not used by the address validation process. For more information, see About Additional Input Data . |
| AddressLine1 | If the address was validated, the first line of the validated and standardized address. If the address could not be validated, the first line of the input address without any changes. |
| AddressLine2 | If the address was validated, the second line of the validated and standardized address. If the address could not be validated, the second line of the input address without any changes. |
| AddressLine3 | If the address was validated, the third line of the validated and standardized address. If the address could not be validated, the third line of the input address without any changes. |
| AddressLine4 | If the address was validated, the fourth line of the validated and standardized address. If the address could not be validated, the fourth line of the input address without any changes. |
| AddressLine5 | For U.K. addresses only. If the address was validated, the fifth line of the validated and standardized address. If the address could not be validated, the fifth line of the input address without any changes. |
| City | The validated city name. |

| Response Element | Description |
|------------------------|---|
| Country | The country in the format determined by what you selected in OutputCountryFormat: <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| DepartmentName | For U.K. addresses only, a subdivision of a firm. For example, Engineering Department. |
| FirmName | The validated firm or company name. |
| PostalCode | The validated ZIP Code™ or postal code. |
| PostalCode.AddOn | The 4-digit add-on part of the ZIP Code™. For example, in the ZIP Code™ 60655-1844, 1844 is the 4-digit add-on. (U.S. addresses only.) |
| PostalCode.Base | The 5-digit ZIP Code™; for example 20706 (U.S. addresses only). |
| StateProvince | The validated state/province or its abbreviated value. |
| USUrbanName | The validated urbanization name. (U.S. addresses only.) This is used primarily for Puerto Rico addresses. |
| DefaultValidPostalCode | This field is generated only for Canadian addresses. A value of Y indicates that the record is non-correctable (VN) type. In such cases, you have the option of lowering the output Confidence score for the record. To do this, select the Switch default valid postal code confidence check-box in the Input Options. For more information, see section Canadian Address Options on page 480. Note: For all other records, the field value is blank. |

Parsed Address Elements Output

Output addresses are formatted in the parsed address format if you set `OutputRecordType=E`. If you want formatted data in the Parsed Address format to be returned when validation fails (that is, a normalized address), specify `OutputFormattedOnFail=Y`.

Note: If you always want parsed input data returned regardless of whether or not validation is successful, specify `OutputRecordType=I`. For more information, see [Parsed Input](#) on page 569.

Table 132: Parsed Address Output

| Response Element | Description |
|--|--|
| <code>AdditionalInputData</code> | Input data not used by <code>ValidateAddress</code> . For more information, see Additional Input Data on page 524. |
| <code>AdditionalInputData.Base</code> | Input data that was not output to the standardized address by <code>ValidateAddress</code> . For more information, see Additional Input Data on page 524. |
| <code>AdditionalInputData.Unmatched</code> | Input data passed to the matcher but not used by <code>ValidateAddress</code> for validation. For more information, see Additional Input Data on page 524. |
| <code>ApartmentLabel</code> | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| <code>ApartmentLabel2</code> | Secondary apartment designator, for example: 123 E Main St APT 3, 4th Floor Note: In this release, this field will always be blank. |
| <code>ApartmentNumber</code> | Apartment number. For example: 123 E Main St APT 3 |
| <code>ApartmentNumber2</code> | Secondary apartment number. For example: 123 E Main St APT 3, 4th Floor Note: In this release, this field will always be blank. |

| Response Element | Description |
|---|---|
| CanadianDeliveryInstallationAreaName | Delivery installation name (Canadian addresses only) |
| CanadianDeliveryInstallationQualifierName | Delivery installation qualifier (Canadian addresses only) |
| CanadianDeliveryInstallationType | Delivery installation type (Canadian addresses only) |
| City | Validated city name |
| Country | <p>Country. Format is determined by what you selected in <code>OutputCountryFormat</code>:</p> <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| DepartmentName | For U.K. addresses only, a subdivision of a firm. For example, Engineering Department. |
| FirmName | The validated firm or company name |
| HouseNumber | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional | Leading directional, for example: 123 E Main St Apt 3 |
| POBox | Post office box number. If the address is a rural route address, the rural route box number will appear here. |
| PostalCode | Validated postal code. For U.S. addresses, this is the ZIP Code. |

| Response Element | Description |
|---------------------|---|
| PrivateMailbox | Private mailbox indicator. |
| PrivateMailbox.Type | <p>The type of private mailbox. Possible values include:</p> <ul style="list-style-type: none"> • Standard • Non-Standard <p>Note: This replaces PrivateMailboxType (no period in field name). Please modify your API calls accordingly.</p> |
| RRHC | Rural Route/Highway Contract indicator |
| StateProvince | Validated state or province name |
| StreetName | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix | Street suffix, for example: 123 E Main St Apt 3 |
| TrailingDirectional | Trailing directional, for example: 123 Pennsylvania Ave NW |
| USUrbanName | USPS® urbanization name. Puerto Rican addresses only. |

Parsed Input

The output can include the input address in parsed form. This type of output is referred to as "parsed input." Parsed input fields contain the address data that was used as input regardless of whether or not ValidateAddress validated the address. Parsed input is different from the "parsed address elements" output in that parsed address elements contain the validated address if the address could be validated, and, optionally, the input address if the address could not be validated. Parsed input always contains the input address regardless of whether or not ValidateAddress validated the address.

To include parsed input fields in the output, set `OutputRecordType=I`.

Table 133: Parsed Input

| Response Element | Description |
|---|---|
| ApartmentLabel.Input | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| ApartmentNumber.Input | Apartment number, for example: 123 E Main St APT 3 |
| CanadianDeliveryInstallationAreaName.Input | Delivery installation name (Canadian addresses only) |
| CanadianDeliveryInstallationQualifierName.Input | Delivery installation qualifier (Canadian addresses only) |
| CanadianDeliveryInstallationType.Input | Delivery installation type (Canadian addresses only) |
| City.Input | Validated city name |
| Country.Input | Country. Format is determined by what you selected in OutputCountryFormat: <ul style="list-style-type: none"> • ISO Code • UPU Code • English • French • German • Spanish |
| FirmName.Input | The validated firm or company name |
| HouseNumber.Input | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional.Input | Leading directional, for example: 123 E Main St Apt 3 |
| POBox.Input | Post office box number. If the address is a rural route address, the rural route box number will appear here. |

| Response Element | Description |
|---------------------------|--|
| PostalCode.Input | Validated postal code. For U.S. addresses, this is the ZIP Code. |
| PrivateMailbox.Input | Private mailbox indicator |
| PrivateMailbox.Type.Input | The type of private mailbox. Possible values include: <ul style="list-style-type: none"> • Standard • Non-Standard |
| RRHC.Input | Rural Route/Highway Contract indicator |
| StateProvince.Input | Validated state or province name |
| StreetName.Input | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix.Input | Street suffix, for example: 123 E Main St Apt 3 |
| TrailingDirectional.Input | Trailing directional, for example: 123 Pennsylvania Ave NW |
| USUrbanName.Input | USPS® urbanization name |

Postal Data Output

If `OutputRecordType` contains P then the following fields are returned in the output.

Table 134: Postal Data Output

| Response Element | Description |
|------------------|---|
| CanadianSERPCode | Validation/correction return code (Canadian addresses only). For more information, see Obtaining SERP Return Codes on page 487. |

| Response Element | Description |
|-------------------------|---|
| IntHexaviaCode | For addresses in France only, a numeric code that represents the street. For information about Hexavia codes, see www.laposte.fr . |
| IntINSEECODE | For addresses in France only, a numeric code that represents the city. For a listing of INSEE codes, see www.insee.fr . |
| PostalBarCode | The two-digit delivery point portion of the delivery point barcode (U.S. addresses only). For more information, see Creating Delivery Point Barcodes on page 460. |
| USAltAddr | Indicates whether or not alternate address matching logic was used, and if so which logic was used (U.S. addresses only). One of the following: <ul style="list-style-type: none"> null No alternate address scheme used. D Delivery point alternate logic was used. E Enhanced highrise alternate match logic was used. S Small town default logic was used. U Unique ZIP Code logic was used. |
| USBCCheckDigit | Check-digit portion of the 11-digit delivery point barcode (U.S. addresses only). For more information, see Creating Delivery Point Barcodes on page 460. |
| USCarrierRouteCode | Carrier route code (U.S. addresses only). For more information, see Obtaining Carrier Route Codes on page 459. |
| USCongressionalDistrict | Congressional district (U.S. addresses only). For more information, see Obtaining Congressional Districts on page 458. |
| USCountyName | County name (U.S. addresses only). For more information, see Obtaining County Names on page 459. |
| USFinanceNumber | The finance number in which the address resides (U.S. addresses only). The finance number is a number assigned by the USPS to an area that covers multiple ZIP Codes. An address is validated only if its finance number matches the finance number of the candidate address in the U.S. Database. |

| Response Element | Description |
|--------------------|--|
| USFIPSCountyNumber | FIPS (Federal Information Processing Standards) county number (U.S. addresses only). For more information, see Obtaining FIPS County Numbers on page 459. |
| USLACS | <p>Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following:</p> <p>Y Yes, the address is a candidate for LACS^{Link} processing. If LACS^{Link} is enabled, an attempt is made to convert the address using the LACS^{Link} database. If the conversion attempt is successful, the output address is the new address obtained from the LACS^{Link} database. If the attempt is not successful, the address will not be converted.</p> <p>N No, the address is not a candidate for LACS^{Link} processing. LACS^{Link} processing may still be attempted if LACS^{Link} processing is requested, the LACS^{Link} database is installed, and one of the following is true:</p> <ul style="list-style-type: none"> • The address matches to a Rural Route address and the RecordType.Default field returns a Y. • The input address could not be matched to any address in the U.S. Postal Database (Failures due to multiple matches are not LACS^{Link} candidates.) |
| USLastLineNumber | <p>A six-character alphanumeric value that groups together ZIP Codes that share the same primary city. For example, addresses with the following two last lines would have the same last line number:</p> <p>Chantilly VA 20151 Chantilly VA 20152</p> |

Result Indicators

Result indicators provide information about the kinds of processing performed on an address. There are two types of result indicators:

Record-Level Result Indicators

Record-level result indicators provide data about the results of ValidateAddress processing for each record, such as the success or failure of the match attempt, which coder processed the address, and other details. The following table lists the record-level result indicators returned by ValidateAddress.

Table 135: Record Level Indicators

| Response Element | Description |
|------------------|--|
| AddressFormat | <p>The type of address data being returned:</p> <p>F French format (for example: 123 Rue Main)</p> <p>E English format (for example: 123 Main St)</p> |
| Confidence | <p>The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct. For multiple matches, the confidence level is 0. For details about how this number is calculated, see Introduction to the Validate Address Confidence Algorithm on page 1036.</p> |
| CouldNotValidate | <p>If no match was found, which address component could not be validated:</p> <ul style="list-style-type: none"> • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber • RuralRoute <p>Note: More than one component may be returned, in a comma-separated list.</p> |

| Response Element | Description |
|------------------|--|
| CountryLevel | <p>The category of address matching available. This is always "A" for U.S. and Canadian addresses. One of the following:</p> <p>A The address is in a country for which there is highly detailed postal data available. Addresses in this match level can have the following address elements validated and corrected, and added if missing from the input:</p> <ul style="list-style-type: none">• Postal code• City name• State/county name• Street address elements• Country name <p>B The address is in a country for which there is a medium level of postal data available. Addresses in this match level can have the following address elements validated and corrected, and added if missing from the input:</p> <ul style="list-style-type: none">• Postal code• City name• State/county name• Country name <p>C The address is in a country for which the postal data is least detailed. Addresses in this match level can have the following actions performed on them:</p> <ul style="list-style-type: none">• Validate and correct country name (cannot supply missing country name)• Validate the format of the postal code (cannot supply missing postal code or validate the code) |

| Response Element | Description |
|------------------|---|
| MatchScore | <p>MatchScore provides an indication of the degree to which the output address is correct. It is significantly different from Confidence in that Confidence indicates how much the input address changed to obtain a match, whereas the meaning of Match Score varies between U.S. and non-U.S. addresses.</p> <p>For U.S. addresses, MatchScore is a one-digit score on a scale of 0 to 9 that reflects the closeness of the street-name match (after transformations by ValidateAddress, if any). Zero indicates an exact match and 9 indicates the least likely match. If no match was found, this field is blank.</p> <p>For non-U.S. and non-Canadian addresses, MatchScore is a five-digit score, with a maximum value of 00999. Higher numbers indicates a closer match.</p> <p>This field does not apply to Canadian addresses.</p> <p>Note that you cannot equate match scores from U.S. addresses with those of non-U.S. addresses. For example, a match score of 4 for a U.S address does not indicate the same level of match as a 00004 for a non-U.S. address.</p> <p>Note: The Validate Address and Advanced Matching Module components both use the MatchScore field. The MatchScore field value in the output of a dataflow is determined by the last stage to modify the value before it is sent to an output stage. If you have a dataflow that contains Validate Address and Advanced Matching Module components and you want to see the MatchScore field output for each stage, use a Transformer stage to copy the MatchScore value to another field. For example, Validate Address produces an output field called MatchScore and then a Transformer stage copies the MatchScore field from Validate Address to a field called AddressMatchScore. When the matcher stage runs it populates the MatchScore field with the value from the matcher and passes through the AddressMatchScore value from Validate Address.</p> |
| MultimatchCount | If multiple matches were found, indicates the number of records that are possible matches. |
| MultipleMatches | <p>Indicates which address component had multiple matches, if multiple matches were found:</p> <ul style="list-style-type: none"> • Firm • LeadingDirectional • PostalCode • StreetName • StreetSuffix • TrailingDirectional • Urbanization <p>Note: More than one component may be returned, in a comma-separated list.</p> |

| Response Element | Description | | | | | | |
|---------------------|--|-------------|---------------------------------------|-------------|--|------------|-----------------------------|
| ProbableCorrectness | <p>Indicates the accuracy of a match on the scale of 0 to 9. The result can be:</p> <ul style="list-style-type: none"> • <i>Blank</i> - No Match Found • 0 - Most likely to be correct (Exact Match) • 1-8 - Intermediate probability of being correct • 9 - Match least likely to be correct | | | | | | |
| ProcessedBy | <p>Which address coder processed the address:</p> <table> <tr> <td>USA</td> <td>U.S. address coder</td> </tr> <tr> <td>CAN</td> <td>Canadian address coder</td> </tr> <tr> <td>INT</td> <td>International address coder</td> </tr> </table> | USA | U.S. address coder | CAN | Canadian address coder | INT | International address coder |
| USA | U.S. address coder | | | | | | |
| CAN | Canadian address coder | | | | | | |
| INT | International address coder | | | | | | |
| RecordType | <p>Type of address record, as defined by U.S. and Canadian postal authorities (supported for U.S. and Canadian addresses only):</p> <ul style="list-style-type: none"> • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal | | | | | | |
| RecordType.Default | <p>Code indicating the "default" match:</p> <table> <tr> <td>Y</td> <td>The address matches a default record.</td> </tr> <tr> <td>null</td> <td>The address does not match a default record.</td> </tr> </table> | Y | The address matches a default record. | null | The address does not match a default record. | | |
| Y | The address matches a default record. | | | | | | |
| null | The address does not match a default record. | | | | | | |
| Status | <p>Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches.</p> <table> <tr> <td>null</td> <td>Success</td> </tr> <tr> <td>F</td> <td>Failure</td> </tr> </table> | null | Success | F | Failure | | |
| null | Success | | | | | | |
| F | Failure | | | | | | |

| Response Element | Description |
|--------------------|--|
| Status.Code | Reason for failure, if there is one. For multiple matches, all possible matches is "MultipleMatchesFound." <ul style="list-style-type: none"> • DisabledCoder • InsufficientInputData • MultipleMatchesFound • UnableToValidate |
| Status.Description | Description of the problem, if there is one. <p>Possible Multiple Addresses Found This value will appear if Status.Code=MultipleMatchesFound.</p> <p>Address Not Found This value will appear if Status.Code=UnableToValidate.</p> <p>PerformUSProcessing disabled This value will appear if Status.Code=DisabledCoder.</p> <p>PerformCanadianProcessing disabled This value will appear if Status.Code=DisabledCoder.</p> <p>PerformInternationalProcessing disabled This value will appear if Status.Code=DisabledCoder.</p> |

Field-Level Result Indicators

Field-level result indicators describe how ValidateAddress handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in **HouseNumber.Result**.

To enable field-level result indicators, specify `OutputFieldLevelReturnCodes=Y`. For more information, see [Output Data Options](#) on page 452.

The following table lists the field-level result indicators. If a particular field does not apply to an address, the result indicator may be blank.

Table 136: Field-Level Result Indicators

| Response Element | Description |
|-----------------------|--|
| AddressRecord.Result | <p>These result codes apply to international addresses only.</p> <ul style="list-style-type: none"> M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. S Standardized. This option includes any standard abbreviations. U Unmatched. V Validated. The data was confirmed correct and remained unchanged from input. |
| ApartmentLabel.Result | <ul style="list-style-type: none"> A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. C Corrected. U.S. and Canadian addresses only. D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data. F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. R The apartment label is required but is missing from the input address. U.S. addresses only. S Standardized. This option includes any standard abbreviations. U Unmatched. Does not apply to Canadian addresses. V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|------------------------|---|
| ApartmentNumber.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. addresses that are an EWS match will have a value of P. U.S. and Canadian addresses only.</p> <p>R The apartment number is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|------------------|--|
| City.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>F Hyphens missing or punctuation errors. Canadian addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R The city is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| Country.Result | <p>These result codes do not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|--------------------|--|
| FirmName.Result | <p>C Corrected. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only.</p> <p>U Unmatched. U.S. and Canadian addresses only.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. U.S. addresses only.</p> |
| HouseNumber.Result | <p>A Appended. The field was added to a blank input field. Canadian addresses only.</p> <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>O Out of range. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The house number is required but is missing from the input address. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|---------------------------|--|
| LeadingDirectional.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. Non-blank input was corrected to a non-blank value. U.S. addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. Does not apply to Canadian addresses.</p> |

| Response Element | Description |
|------------------|---|
| POBox.Result | <p>A Appended. The field was added to a blank input field. Canadian addresses only.</p> <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The P.O. Box number is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> <p>Blank Not applicable.</p> |

| Response Element | Description |
|-----------------------|--|
| PostalCode.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses.</p> <p>R The postal code is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses.</p> <p>U Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| PostalCodeCity.Result | <p>These result codes apply to international addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|-------------------|---|
| PostalCode.Source | <p>These result codes apply to U.S. addresses only.</p> <p>FinanceNumber The ZIP Code™ in the input was verified by using USPS® Finance Number groupings.</p> <p>ZIPMOVE The ZIP Code™ in the input address was corrected because the USPS® redrew ZIP Code™ boundaries and the address is now in a different ZIP Code™.</p> |
| PostalCode.Type | <p>P The ZIP Code™ contains only PO Box addresses. U.S. addresses only.</p> <p>U The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.</p> <p>M The ZIP Code™ is for military addresses. U.S. addresses only.</p> <p>null The ZIP Code™ is a standard ZIP Code™.</p> |
| RRHC.Result | <p>C Corrected. Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data.</p> <p>M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>R The rural route/highway contract is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only.</p> <p>U Unmatched. U.S. and Canadian addresses only.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input. U.S. and Canadian addresses only.</p> |

| Response Element | Description |
|----------------------|--|
| RRHC.Type | <p>These result codes apply to U.S. addresses only.</p> <p>HC The address is a Highway Contract address.</p> <p>RR The address is a Rural Route address.</p> |
| StateProvince.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R The state is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| Street.Result | <p>These result codes apply to international addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output.</p> <p>R Street corrected. House number is out of range. Applies to French, UK, and Japanese records only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

| Response Element | Description |
|------------------------------------|---|
| StreetName.AbbreviatedAlias.Result | <p>Indicates the result of abbreviated alias processing. One of the following:</p> <ul style="list-style-type: none"> null No abbreviated alias processing attempted. B The StreetName field contains the base street name. L The standardized address length is less than 31 characters so the StreetName field contains the base name. N No abbreviated alias found. Y An abbreviated alias was found for input address. The StreetName field contains the abbreviated alias. |
| StreetName.Alias.Type | <p>This result code applies to U.S. addresses only.</p> <p>Note: In previous releases this field was named StreetName.AliasType with no "." between "Alias" and "Type." This old name is obsolete. Please update your processes to use the new name StreetName.Alias.Type.</p> <ul style="list-style-type: none"> Abbreviated The alias is an abbreviation of the street name. For example, HARTS-NM RD is an abbreviated alias for HARTSVILLE NEW MARLBORO RD. Changed There has been an official street name change and the alias reflects the new name. For example if SHINGLE BROOK RD is changed to CANNING DR, then CANNING DR would be a changed alias type. Other The street alias is made up of other names for the street or common abbreviations of the street. Preferred The street alias is the locally preferred alias. For example, a street is named "South Shore Dr." because it runs along the southern shore of a lake, not because it is south of a municipal demarcation line. So, "South" is not a predirectional in this case and should not be shorted to "S". So, "South Shore Dr." would be the preferred alias. |

| Response Element | Description |
|----------------------------------|--|
| StreetName.PreferredAlias.Result | <p>Indicates the result of preferred alias processing. One of the following:</p> <ul style="list-style-type: none"> null No preferred alias processing attempted. A Preferred alias processing was not attempted because the input address matched to an alias. Preferred alias processing is only attempted for base addresses. N No preferred alias found. Y A preferred alias was found for the input address. The StreetName field contains the preferred alias. |
| StreetName.Result | <ul style="list-style-type: none"> A Appended. The field was added to a blank input field. Canadian addresses only. C Corrected. U.S. and Canadian addresses only. D Dropped. The field provided on input was removed. U.S. addresses only. For more information, see About Additional Input Data. F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only. U Unmatched. V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|---------------------|---|
| StreetSuffix.Result | <ul style="list-style-type: none">A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.C Corrected. U.S. and Canadian addresses only.D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data.F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.S Standardized. This option includes any standard abbreviations.U Unmatched. Does not apply to U.S. addresses.V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|----------------------------|--|
| TrailingDirectional.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. and Canadian addresses only.</p> <p>D Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see About Additional Input Data.</p> <p>F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only.</p> <p>S Standardized. This option includes any standard abbreviations.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| USUrbanName.Result | <p>These result codes apply to U.S. addresses only.</p> <p>A Appended. The field was added to a blank input field.</p> <p>C Corrected.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field.</p> <p>U Unmatched.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |

Output from Options

ValidateAddress returns additional data depending on the options you select. For information on the output generated by each option, see the options listed in the following sections:

Enhanced Line of Travel Output

Enhanced Line of Travel processing produces the following output.

| Response Element | Description |
|------------------|--|
| USLOTCode | <p>Line of Travel sequence code and an indicator denoting USPS® LOT sequence. This field is in the format nnnnY where:</p> <p>nnnn The four-digit LOT code.</p> <p>Y One of the following:</p> <ul style="list-style-type: none"> • A—Ascending LOT sequence • D—Descending LOT sequence |
| USLOTHex | A hexadecimal value that allows you to sort your file in ascending order only. The hexadecimal values range from 0 to FF ascending, then FF through 0 descending. |
| USLOTSquence | A two-byte value used for final sortation in place of the DPC add-on. It consists of an uppercase letter followed by a digit 0 through 9. Values range from A0 (99 descending) through J9 (00 descending), and K0 (00 ascending) through T9 (99 ascending). |

LACS^{Link} Output

| Response Element | Description |
|-------------------|--|
| USLACS | <p>Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following:</p> <p>Y Yes, the address is a candidate for LACS^{Link} processing. If LACS^{Link} is enabled, ValidateAddress will attempt to convert the address using the LACS^{Link} database. If the conversion attempt is successful, the output address is the new address obtained from the LACS^{Link} database. If the attempt is not successful, the address will not be converted.</p> <p>N No, the address is not a candidate for LACS^{Link} processing. LACS^{Link} processing may still be attempted if LACS^{Link} processing is requested, the LACS^{Link} database is installed, and one of the following is true:</p> <ul style="list-style-type: none"> • The address matches to a Rural Route address and the RecordType.Default field returns a Y. • The input address could not be matched to any address in the U.S. Postal Database (Failures due to multiple matches are not LACS^{Link} candidates.) |
| USLACS.ReturnCode | <p>Indicates the success or failure of LACS^{Link} processing. (U.S. addresses only.)</p> <p>A LACS^{Link} processing successful. Record matched through LACS^{Link} processing.</p> <p>00 LACS^{Link} processing failed. No matching record found during LACS^{Link} processing.</p> <p>09 LACS^{Link} processing matched the input address to an older highrise default address. The address has been converted. Rather than provide an imprecise address, LACS^{Link} processing does not provide a new address.</p> <p>14 LACS^{Link} processing failed. Match found during LACS^{Link} processing but conversion did not occur due to other USPS® regulations.</p> <p>92 LACS^{Link} processing successful. Record matched through LACS^{Link} processing. Unit number dropped on input.</p> <p>null LACS^{Link} did not process the record, or LACS^{Link} processing was not attempted.</p> |

RDI Output

| Response Element | Description |
|------------------|---|
| RDI | Return values indicating address type. B The address is a business address. R The address is a residential address. M The address is both a residential and a business address. null Not checked because the address did not code at a ZIP + 4 [®] level, or RDI™ was not performed. |

DPV and CMRA Output

| Response Element | Description |
|------------------|--|
| DPV | <p>Indicates the results of Delivery Point Validation (DPV) processing.</p> <p>Y DPV confirmed. Mail can be delivered to the address.</p> <p>N Mail cannot be delivered to the address.</p> <p>S The building number was validated but the unit number could not be confirmed. A building number is the primary address number for a building. A unit number is a number of a distinct mailing address within a building such as an apartment, suite, floor, and so on. For example, in this address 424 is the building number and 12 is the unit number:</p> <p>424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA</p> <p>D The building number was validated but the unit number was missing from input. A building number is the primary address number for a building. A unit number is a number of a distinct mailing address within a building such as an apartment, suite, floor, and so on. For example, in this address 424 is the building number and 12 is the unit number:</p> <p>424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA</p> <p>M The address matches multiple valid delivery points.</p> <p>U The address could not be confirmed because the address did not code at the ZIP + 4[®] level.</p> <p>V The address caused a false-positive violation.</p> |
| CMRA | <p>Indicates if the address is a Commercial Mail Receiving Agency (CMRA)</p> <p>Y Yes, the address is a CMRA.</p> <p>N No, the address is not a CMRA.</p> <p>U Unconfirmed.</p> |

| Response Element | Description |
|------------------|--|
| DPVFootnote | <p>DPV footnote codes.</p> <p>AA Input address matched to the ZIP + 4[®] file.</p> <p>A1 Input address not matched to the ZIP + 4[®] file.</p> <p>BB Input address matched to DPV (all components).</p> <p>CC Input address primary number matched to DPV but secondary number not match (present but not valid).</p> <p>F1 Input address is military; DPV bypassed.</p> <p>G1 Input address is general delivery; DPV bypassed.</p> <p>M1 Input address primary number missing.</p> <p>M3 Input address primary number invalid.</p> <p>N1 Input address primary number matched to DPV but high rise address missing secondary number.</p> <p>P1 Input address missing RR or HC Box number.</p> <p>P3 Input address missing PO, RR, or HC Box number</p> <p>RR Input address matched to CMRA.</p> <p>R1 Input address matched to CMRA but secondary number not present.</p> <p>R7 Input address matched to phantom carrier route R777 (not eligible for street delivery).</p> <p>U1 Input address is unique ZIP; DPV bypassed.</p> |
| DPVVacant | <p>Indicates whether the building is vacant (unoccupied for 90 days). One of the following:</p> <p>Y Yes, the building is vacant.</p> <p>N No, the building is not vacant.</p> <p>null The <code>DPVDetermineVacancy</code> option was not turned on.</p> |
| DPVNoStat | <p>Indicates whether the building is a "no stat" building and therefore unable to receive mail. One of the following:</p> <p>Y Yes, the building is a "no stat" building, which means the building is not receiving mail.</p> <p>N No, the building is not a "no stat" building, which means the building does receive mail.</p> <p>null The <code>DPVDetermineNoStat</code> option was not turned on.</p> |

Suite^{Link} Output

| Response Element | Description |
|---------------------|---|
| SuiteLinkReturnCode | <p>Indicates whether or not ValidateAddress corrected the secondary address information (U.S. addresses only). One of the following:</p> <p>A ValidateAddress corrected the secondary address information.</p> <p>00 ValidateAddress did not correct the secondary address information.</p> <p>null Suite^{Link} was not performed.</p> <p>XX Suite^{Link} processing encountered an error. For example, an error would occur if the Suite^{Link} database is expired.</p> |
| SuiteLinkMatchCode | <p>Provides additional information on the Suite^{Link} match attempt. (U.S. addresses only)</p> <p>A ValidateAddress corrected the secondary address information.</p> <p>B ValidateAddress did not correct the secondary address information. No additional detail about the match attempt is available.</p> <p>C The words in the FirmName field are all "noise" words. Noise words are defined by the USPS[®] and are ignored when attempting to match the firm name. Examples of noise words are "company" and "corporation". ValidateAddress is not able to correct secondary address information for firm names that consist entirely of noise words. For example "Company and Corporation" is all noise words.</p> <p>D The address is not a high-rise default address. Suite^{Link} matching is only done for high-rise default addresses. A high-rise default is a default to use when the address does not contain valid secondary information (the apartment number or apartment type is missing).</p> <p>E Suite^{Link} processing failed because the Suite^{Link} database is expired.</p> <p>null Suite^{Link} was not performed or there was an error.</p> |

| Response Element | Description |
|-------------------|--|
| SuiteLinkFidelity | <p>Indicates how well ValidateAddress matched the firm name to the firm names in the Suite^{Link} database.</p> <p>1 The firm name matches the Suite^{Link} database exactly.</p> <p>2 Good match. All words in the firm name except one matched the firm name in the Suite^{Link} database.</p> <p>3 Poor match. More than one word in the firm name did not match the firm name in the Suite^{Link} database.</p> <p>null Suite^{Link} could not match the firm name, or was not performed, or there was an error.</p> |

VeriMove Output

| Response Element | Description |
|-------------------|---|
| VeriMoveDataBlock | <p>Indicates whether or not ValidateAddress should return a 250-byte field containing input data to pass to VeriMove Express. This field contains the Detail Results Indicator data required by VeriMove. For more information about the contents of this field, see the VeriMove User's Guide. One of the following:</p> <p>Y Yes, return the field VeriMoveDataBlock.</p> <p>N No, do not return the field VeriMoveDataBlock.</p> |

Additional Input Data

Some input data is ignored during the address standardization process. This extraneous data (sometimes referred to as "dropped data") is returned in the AdditionalInputData field. Some examples of dropped data include:

- Delivery instructions (for example, "Leave at back door")
- Phone numbers (for example, "555-135-8792")
- Attention lines (for example, "Attn: John Smith")

Data such as this is generally not embedded in an address. If it is embedded, the extraneous data can usually be identified and returned in the AdditionalInputData field.

Note: Dropped data from split indicia addresses is not returned. A split indicia address is one where a primary address is split between multiple address lines. For example, if the primary

address is "1 Green River Valley Rd" then the following would be a split indicia version of this address:

1 Green River
Valley Rd
01230

If there is more than one piece of dropped data in an address, each piece of data is separated by a semicolon and a space ("; ") for U.S. addresses and a space for addresses outside the U.S. The order of dropped data in `AdditionalInputData` is:

1. Care of, mail stop (U.S. addresses only)
2. Other extraneous data found on address lines
3. Entire unused data lines

For example, if this is the input address:

123 Main St C/O John Smith
Apt 5 Drop at back dock
jsmith@example.com
555-123-4567
05674

Then `AdditionalInputData` would contain:

C/O John Smith; Apt 5 Drop At Back Dock; 555-123-4567; Jsmith@example.com; 555-123-4567

Care of Data

For U.S. addresses only, "care of" data is returned in `AdditionalInputData`. The following addresses contain examples of "care of" data:

123 Main St C/O John Smith
Apt 5
05674

123 Main St
Apt 5 ATTN John Smith
05674

123 Main St Apt 5
MailStop 2
05674

Extraneous Data on Its Own Address Line

`ValidateAddress` returns extraneous data on its own address line for U.S. and Canadian addresses.

For U.S. addresses, `ValidateAddress` uses the first two non-blank address lines to perform address standardization, unless either the firm name extraction or urbanization code extraction options are enabled (see [Address Line Processing for U.S. Addresses](#) on page 452 for more information).

Data on other address lines is returned in `AdditionalInputData`. In the following address, "John Smith" would be returned in `AdditionalInputData` because it is in the third non-blank address line and `ValidateAddress` only uses the first two non-blank address lines for U.S. addresses.

```
123 Main St  
Apt 5  
John Smith  
05674
```

If one of either of the first two non-blank address lines contains extraneous data, that data is returned in `AdditionalInputData`. For example, in the following addresses "John Smith" would be returned in `AdditionalAddressData`.

```
123 Main St  
John Smith  
05674
```

```
John Smith  
123 Main St  
05674
```

In the following address both "John Smith" and "Apt 5" would both be returned in `AdditionalInputData`. "John Smith" would be returned because it is extraneous data in one of the first two address lines and "Apt 5" would be returned because U.S. address data must be in the first two non-blank address lines.

```
John Smith  
123 Main St  
Apt 5  
05674
```

Extraneous Data Within an Address Line

Extraneous data that is within an address line is returned in `AdditionalInputData`. For example, in the following addresses "John Smith" would be returned in `AdditionalInputData`.

```
123 Main St John Smith  
05674
```

```
123 Main St Apt 5 John Smith  
05674
```

```
123 Main St John Smith  
Apt 5  
05674
```

```
123 Main St  
Apt 5 John Smith  
05674
```

For U.S. addresses, only extraneous data at the end of the address line is returned in `AdditionalInputData`. Extraneous data that is not at the end of an address line is not returned for U.S. addresses. For example, in the following addresses "John Smith" is not returned.

John Smith 123 Main St
05674

123 Main John Smith St
05674

The `AdditionalInputData` will sometimes contain the original street name or suffix if the street name was changed to obtain a match and the street name or suffix was at the end of a line. For example this address:

Precisely
4200 Parliament
Lanham MD

`ValidateAddress` would correct the spelling of the street name and add the suffix, returning "4200 Parliament PI" as the corrected street address and "Parlament" in `AdditionalInputData`.

Dual Addresses

A dual address is an address that contains both street and PO Box/Rural Route/Highway Contract information. Depending on the processing options you select, the portion of the dual address that is not used for address standardization may be returned in `AdditionalInputData`. For more information, see [About Dual Address Logic](#) on page 465.

ValidateAddressGlobal

`ValidateAddressGlobal` provides enhanced address standardization and validation for addresses outside the U.S. and Canada. `ValidateAddressGlobal` can also validate addresses in the U.S. and Canada but its strength is validation of addresses in other countries. If you process a significant number of addresses outside the U.S. and Canada, you should consider using `ValidateAddressGlobal`.

`ValidateAddressGlobal` is part of the Universal Addressing Module.

`ValidateAddressGlobal` performs several steps to achieve a quality address, including transliteration, parsing, validation, and formatting.

Character Set Mapping and Transliteration

`ValidateAddressGlobal` handles international strings and their complexities. It uses fully Unicode enabled string processing which enables the transliteration of non-roman characters into the Latin character set and mapping between different character sets.

Character set mapping and transliteration features include:

- Support for over 30 different character sets including UTF-8, ISO 8859-1, GBK, BIG5, JIS, EBCDIC
- Proper "elimination" of diacritics according to language rules
- Transliteration for various alphabets into Latin Script
- Greek (BGN/PCGN 1962, ISO 843 - 1997)
- Cyrillic (BGN/PCGN 1947, ISO 9 - 1995)

- Hebrew
- Japanese Katakana, Hiragana and Kanji
- Chinese Pinyin (Mandarin, Cantonese)
- Korean Hangul

Address Parsing, Formatting, and Standardization

Restructuring incorrectly fielded address data is a complex and difficult task especially when done for international addresses. People introduce many ambiguities as they enter address data into computer systems. Among the problems are misplaced elements (such as company or personal names in street address fields) or varying abbreviations that are not only language, but also country specific. `ValidateAddressGlobal` identifies address elements in address lines and assigns them to the proper fields. This is an important precursor to the actual validation. Without restructuring, "no match" situations might result.

Properly identified address elements are also important when addresses have to be truncated or shortened to fit specific field length requirements. With the proper information in the right fields, specific truncation rules can be applied.

- Parses and analyzes address lines and identifies individual address elements
- Processes over 30 different character sets
- Formats addresses according to the postal rules of the country of destination
- Standardizes address elements (such as changing AVENUE to AVE)

Global Address Validation

Address validation is the correction process where properly parsed address data is compared against reference databases supplied by postal organizations or other data providers. `ValidateAddressGlobal` validates individual address elements to check for correctness using sophisticated fuzzy matching technology and produces standardized and formatted output based on postal standards and user preferences. `FastCompletion` validation type can be used in quick address entry applications. It allows input of truncated data in several address fields and generates suggestions based on this input.

In some cases, it is not possible to fully validate an address. Here `ValidateAddressGlobal` has a unique deliverability assessment feature that classifies addresses according to their probable deliverability.

Resource URL

```
http://server:port/soap/ValidateAddressGlobal
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddressGlobal">

  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressGlobalRequest>
      <val:input>
        <val:Address>
          <val:Country>USA</val:Country>
          <val:AddressLine1>1 Global View</val:AddressLine1>
          <val:City>Troy</val:City>
          <val:StateProvince>NY</val:StateProvince>
          <val:PostalCode></val:PostalCode>
        </val:Address>
      </val:input>
    </val:ValidateAddressGlobalRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ValidateAddressGlobalResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/ValidateAddressGlobal">

      <ns3:output>
        <ns3:Address>
          <ns3:Country>UNITED STATES</ns3:Country>
          <ns3:AddressLine1>1 GLOBAL VW</ns3:AddressLine1>
          <ns3:HouseNumber>1</ns3:HouseNumber>
          <ns3:StreetName>GLOBAL</ns3:StreetName>
          <ns3:StreetSuffix>VW</ns3:StreetSuffix>
          <ns3:City>TROY</ns3:City>
          <ns3:PostalCode>12180-8371</ns3:PostalCode>
          <ns3:PostalCode.Base>12180</ns3:PostalCode.Base>
          <ns3:PostalCode.AddOn>8371</ns3:PostalCode.AddOn>
          <ns3:StateProvince>NY</ns3:StateProvince>
          <ns3:County>RENSSELAER</ns3:County>
          <ns3:LastLine>TROY NY 12180-8371</ns3:LastLine>
          <ns3:AddressBlock1>1 GLOBAL VW</ns3:AddressBlock1>
          <ns3:AddressBlock2>TROY NY 12180-8371</ns3:AddressBlock2>

          <ns3:ProcessStatus>C4</ns3:ProcessStatus>
          <ns3:ProcessStatus.Description>
```

```

        Corrected - all elements have been checked
    </ns3:ProcessStatus.Description>
    <ns3:ModeUsed>BATCH</ns3:ModeUsed>
    <ns3:CountOverflow>NO</ns3:CountOverflow>
    <ns3:MailabilityScore>5</ns3:MailabilityScore>
    <ns3:Confidence>85.09</ns3:Confidence>

<ns3:ElementResultStatus>88F0F8E0F000000000E0</ns3:ElementResultStatus>

<ns3:ElementInputStatus>00606050600000000060</ns3:ElementInputStatus>

<ns3:ElementRelevance>11101010100000000010</ns3:ElementRelevance>
    <ns3:AddressType>S</ns3:AddressType>
    <ns3:AMAS.Status>EAM0</ns3:AMAS.Status>
    <ns3:user_fields/>
  </ns3:Address>
</ns3:output>
</ns3:ValidateAddressGlobalResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

ValidateAddressGlobal takes a standard address as input. All addresses use this format no matter what country the address is from.

Table 137: ValidateAddressGlobal Input

| Parameter | Format | Description |
|-----------------------------------|-------------|--|
| AddressLine1 through AddressLine6 | String [79] | <p>These fields contain address line data. AddressLine1 contains the first address line, AddressLine2 contains the second address line, and so forth. Note that the city, state/province, and postal code information should be placed in their respective fields, not address line fields. For example:</p> <p>AddressLine1: 17413 Blodgett Road AddressLine2: PO Box 123 City: Mount Vernon StateProvince: WA PostalCode: 97273 Country: USA</p> <p>If the input address is not already parsed into the appropriate address line and City, StateProvince, and PostalCode fields, use the UnformattedLine fields instead of the address line fields.</p> |

| Parameter | Format | Description |
|-----------------|--|---|
| City | String [79] | City name |
| StateProvince | String [79] | State or province. |
| PostalCode | String [79]: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 | The postal code for the address. In the U.S. this is the ZIP Code®. |
| Contact | String [79] | The name of the addressee. For example, "Mr. Jones". |
| Country | String [79] | The name of the country. If no value is specified in the or option, you must specify a country. |
| FirmName | String [79] | The company or firm name. |
| Street | String [79] | Street |
| Number | Building [79] | Number |
| Building | String [79] | Building |
| SubBuilding | String [79] | SubBuilding |
| DeliveryService | String [79] | DeliveryService |

| Parameter | Format | Description |
|--|-------------|---|
| UnformattedLine1 through UnformattedLine10 | String [79] | <p>Use these fields if the input address is completely unparsed and you want ValidateAddressGlobal to attempt to parse the address into the appropriate fields. For example:</p> <p>UnformattedLine1: 17413 Blodgett Road UnformattedLine2: PO Box 123 UnformattedLine3: Mount Vernon WA 97273 UnformattedLine4: USA</p> <p>This address would be parsed into these output fields:</p> <p>AddressLine1: 17413 Blodgett Road AddressLine2: PO Box 123 City: Mount Vernon StateProvince: WA PostalCode: 97273 Country: USA</p> <p>Note: If you specify input in the unformatted line fields you must specify the entire address using only unformatted line fields. Do not use other fields such as City or StateProvince in combination with unformatted line fields.</p> |

Parameters for Options

Input Options

Table 138: ValidateAddressGlobal Input Options

| Parameter | Description/Valid Values |
|--------------------------|---|
| Database.AddressGlobal | Specifies the database resource containing the postal data to use for address validation. Only databases that have been defined in the Global Database Resources panel in the Management Console are available. For more information, see the <i>Spectrum Technology Platform Administration Guide</i> . |
| Input.DefaultCountryISO3 | Specifies a default country to use when the input record does not contain explicit country information. Specify the country using the ISO3 country code. If you do not specify a default country each input record must have the country specified in the Country input field. For a list of ISO codes see ISO Country Codes and Module Support on page 1011. |

| Parameter | Description/Valid Values |
|------------------------|---|
| Input.ForceCountryISO3 | Causes address records to be always treated as originating from the country specified here, overriding the country in the address record and the default country. Specify the country using the ISO3 country code. For a list of ISO codes, see ISO Country Codes and Module Support on page 1011. |
| Input.FormatDelimiter | <p>Enables you to use non-standard formatting for multiline addresses in input files. Acceptable values for this field include the following:</p> <ul style="list-style-type: none"> • CRLF (default) • LF • CR • SEMICOLON (2101 MASSACHUSETTS AVE NW ; WASHINGTON DC 20008) • COMMA (2101 MASSACHUSETTS AVE NW , WASHINGTON DC 20008) • TAB (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • PIPE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • SPACE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) <p>Note: The same value must be selected for both the input option and output option.</p> |

Output Options

Table 139: ValidateAddressGlobal Output Options

| Parameter | Description | | | | |
|-----------------------|--|-------------|---------------------------------------|--------------|--|
| Result.MaximumResults | This option specifies the maximum number of candidate addresses to return. This field is disabled for batch processing; for all other processing modes the default is 1 and the maximum is 99. If you are using FastCompletion mode, you may want to enter a number greater than 1 to ensure you are provided with multiple options for completing a field. | | | | |
| Result.IncludeInputs | <p>Specifies whether to include the input data in the output. If enabled, the output will contain fields that end with .Input containing the corresponding input field. For example, the output field AddressLine1.Input would contain the data specified in the input field AddressLine1.</p> <table border="0"> <tr> <td>TRUE</td> <td>Include the input data in the output.</td> </tr> <tr> <td>FALSE</td> <td>Do not include the input data in the output (default).</td> </tr> </table> | TRUE | Include the input data in the output. | FALSE | Do not include the input data in the output (default). |
| TRUE | Include the input data in the output. | | | | |
| FALSE | Do not include the input data in the output (default). | | | | |

| Parameter | Description |
|--------------------------|--|
| Result.StateProvinceType | <p>Specifies the format for the StateProvince field. One of the following.</p> <p>ABBREVIATION Return the abbreviation for the state or province. For example, North Carolina would be returned as "NC".</p> <p>COUNTRY_STANDARD Return either the abbreviation or the full name depending on the format used by the country's postal authority. (Default)</p> <p>EXTENDED Return the full name of the state or province, not the abbreviation. For example "North Carolina".</p> |
| Result.CountryType | <p>Specifies the language or code to use for the country name returned by ValidateAddressGlobal.</p> <p>ISO2 The two-character ISO code for the country</p> <p>ISO3 The three-character ISO code for the country</p> <p>ISO_NUMBER The ISO country number</p> <p>NAME_CN Chinese</p> <p>NAME_DA Danish</p> <p>NAME_DE German</p> <p>NAME_EN English (default)</p> <p>NAME_ES Spanish</p> <p>NAME_FI Finnish</p> <p>NAME_FR French</p> <p>NAME_GR Greek</p> <p>NAME_HU Hungarian</p> <p>NAME_IT Italian</p> <p>NAME_JP Japanese</p> <p>NAME_KR Korean</p> <p>NAME_NL Dutch</p> <p>NAME_PL Polish</p> <p>NAME_PT Portuguese</p> <p>NAME_RU Russian</p> <p>NAME_SA Sanskrit</p> <p>NAME_SE Swedish</p> |

| Parameter | Description |
|--------------------------|--|
| Result.PreferredScript | <p>Specifies the alphabet in which the output should be returned. The alphabet in which the data is returned differs from country to country. For most countries the output will be Latin I regardless of the selected preferred language.</p> <p>ASCII_Extended ASCII characters with expansion of special characters (for example, Ã– = OE)</p> <p>ASCII_Simplified ASCII characters</p> <p>Database (default) Latin I or ASCII characters (as per reference database standard)</p> <p>Latin Latin I characters</p> <p>Latin_Alt Latin I characters (alternative transliteration)</p> <p>Postal_Admin_Alt Latin I or ASCII characters (local postal administration alternative)</p> <p>Postal_Admin_Pref Latin I or ASCII characters (as preferred by local postal administration)</p> <p>For countries that use an alphabet other than Latin I, the returned alphabet differs from country to country. For more information, see Alphabets for Non-Latin 1 Countries on page 947.</p> |
| Result.PreferredLanguage | <p>Specifies the language in which the output should be returned. The alphabet in which the data is returned differs from country to country, but for most countries the output will be Latin, regardless of the selected preferred language.</p> <p>DATABASE Language derived from reference data for each address. Default.</p> <p>ENGLISH English locality and state/province names output, if available.</p> |
| Result.FormatDelimiter | <p>Enables you to use non-standard formatting for multiline addresses in the output. Acceptable values for this field include the following:</p> <ul style="list-style-type: none"> • CRLF (default) • LF • CR • SEMICOLON (2101 MASSACHUSETTS AVE NW ; WASHINGTON DC 20008) • COMMA (2101 MASSACHUSETTS AVE NW , WASHINGTON DC 20008) • TAB (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • PIPE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) • SPACE (2101 MASSACHUSETTS AVE NW WASHINGTON DC 20008) <p>Note: The same value must be selected for both the input option and output option.</p> |

| Parameter | Description |
|---------------|--|
| Result.Casing | Specifies the casing of the output. |
| | NATIVE Output will be based on the reference database standard. |
| | UPPER Output will be in upper case for all countries. |
| | LOWER Output will be in lower case for all countries. |
| | MIXED Casing determined by country-specific rules. |
| | NOCHANGE For parse mode, returns the data the way it was entered. For validation mode, uses the casing found in the reference data and according to postal rules. Values that could not be checked against the reference data will retain their input casing. |

Alphabets for Non-Latin 1 Countries

For countries that use an alphabet other than Latin I, the returned alphabet differs from country to country. The following table shows how the output is returned for specific countries. All countries that are not listed use the value specified in the field option.

| Country | Database | Latin |
|---------|----------|--|
| RUS | Cyrillic | Cyrillic |
| | | CYRILLIC_ISO CYRILLIC_BGN CYRILLIC_ISO + LATIN_SIMPLE CYRILLIC_ISO + LATIN |
| JPN | Kanji | Kana |
| | | JAPANESE JAPANESE JAPANESE + LATIN_SIMPLE JAPANESE + LATIN |
| CHN | Hanzi | Hanzi |
| | | CHINESE_MANDARIN CHINESE_CANTONESE CHINESE_MANDARIN + LATIN_SIMPLE CHINESE_MANDARIN + LATIN |
| HKG | Hanzi | Hanzi |
| | | CHINESE_CANTONESE CHINESE_MANDARIN CHINESE_CANTONESE + LATIN_SIMPLE CHINESE_CANTONESE + LATIN |
| TWN | Hanzi | Hanzi |
| | | CHINESE_CANTONESE CHINESE_MANDARIN CHINESE_CANTONESE + LATIN_SIMPLE CHINESE_CANTONESE + LATIN |

| Country Database | | | | Latin | | | |
|------------------|---------|---------|---------|-----------|-----------|-----------------------------|----------------------|
| GRC | Greek | Greek | Greek | GREEK_ISO | GREEK_BGN | GREEK_ISO + LATIN_SIMPLE | GREEK_ISO + LATIN |
| KOR | Latin | Hangul | Hanja | KOREAN | KOREAN | KOREAN + LATIN_SIMPLE | KOREAN + LATIN |
| ISR | Latin | Hebrew | Hebrew | HEBREW | HEBREW | HEBREW + LATIN_SIMPLE | HEBREW + LATIN |
| ROM | Latin-3 | Latin-3 | Latin-3 | Latin-3 | Latin-3 | LATIN_SIMPLE | LATIN |
| POL | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| CZE | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| CRI | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| HUN | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| MDA | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| SVK | Latin-2 | Latin-2 | Latin-2 | Latin-2 | Latin-2 | LATIN_SIMPLE | LATIN |
| LAT | Latin-7 | Latin-7 | Latin-7 | Latin-7 | Latin-7 | LATIN_SIMPLE | LATIN |

Process Options

Table 140: ValidateAddressGlobal Process Options

| Parameter | Description |
|---------------------------|---|
| Process.OptimizationLevel | <p>Use this option to set the appropriate balance between processing speed and quality. One of the following:</p> <p>NARROW The parser will honor input assignment strictly, with the exception of separation of House Number from Street information.</p> <p>STANDARD The parser will separate address element more actively as follows:</p> <ul style="list-style-type: none"> • Province will be separated from Locality information • PostalCode will be separated from Locality information • House Number will be separated from Street information • SubBuilding will be separated from Street information • DeliveryService will be separated from Street information • SubBuilding will be separated from Building information • Locality will be separated from PostalCode information <p>WIDE Parser separation will happen similarly to Standard, but additionally up to 10 parsing candidates will be passed to validation for processing. Validation will widen its search tree and take additional reference data entries into account for matching.</p> <p>Please note that adjusting the optimization level might have no effect for countries that lack the postal reference data information required for the kind of separation described above.</p> <p>Increasing separation granularity from Narrow to Standard consumes some processing power, but the major impact on processing speed is from validation processing a larger search tree, thus increasing the number of data accesses and comparisons for the optimization level Wide, in an attempt to make the most out of the input data given.</p> |

| Parameter | Description |
|--------------|--|
| Process.Mode | <p>Specifies the type of processing to perform on the addresses. One of the following:</p> <p>BATCH Use this mode in batch processing environments when no human input or selection is possible. It is optimized for speed and will terminate its attempts to correct an address when ambiguous data is encountered that cannot be corrected automatically. The Batch processing mode will fall back to Parse mode when the database is missing for a specific country.</p> <p>Note: When the Process Status returns a value of I3, the attempt is considered a failure and the Status will return a value of F.</p> <p>CERTIFIED Use this mode in batch processing environments for Australian mail. Validate Address Global is certified by Australia Post's Address Matching Approval System (AMAS). It will standardize and validate your mail against the Postal Address File, providing postal discounts and allowing for the least amount of undeliverable pieces.</p> <p>FASTCOMPLETION Use this mode if you want to use FastCompletion mode to enter truncated data in address fields and have Validate Address Global generate suggestions. For example, if you work in a call center or point-of-sale environment, you can enter just part of an address element and the FastCompletion feature will provide valid options for the complete element.</p> <p>INTERACTIVE Use this mode when working in interactive environments to generate suggestions when an address input is ambiguous. This validation type is especially useful in data entry environments when capturing data from customers or prospects. It requires the input of an almost-complete address and will attempt to validate or correct the data provided. If ambiguities are detected, this validation type will generate up to 20 suggestions that can be used for pick lists. The Interactive processing mode will fall back to Parse mode when the respective database is missing for a specific country.</p> <p>PARSE Use this mode for separating address input into tokens for subsequent processing in other systems, bypassing validation. For example, you could use this mode when address data of already high quality simply needs to be tokenized quickly for export to an external system or for use by a downstream stage.</p> |

| Parameter | Description |
|-----------------------|---|
| Process.MatchingScope | <p>Specifies how closely an address must match the reference data in order for the address to be validated. One of the following:</p> <p>Note: These settings may not have an effect for countries lacking the necessary level of detail in the postal reference data.</p> <p>ALL All address elements must match.</p> <p>DELIVERYPOINT_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, street, house number, and sub building.</p> <p>STREET_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, and street.</p> <p>LOCALITY_LEVEL Validate Global Address must achieve a match on StateProvince, PostalCode, and City/Locality/Suburb.</p> |

Response

Address Data

Table 141: Parsed Address Elements

| Response Element | Description |
|------------------|---|
| AddressBlock1-9 | <p>The AddressBlock output fields contain a formatted version of the standardized or normalized address as it would be printed on a physical mailpiece. Validate Address Global formats the address into address blocks using postal authority standards. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9. For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882</p> |

| Response Element | Description |
|------------------------|---|
| AddressLine1-6 | <p>If the address was validated, the address line fields contain the validated and standardized address lines. If the address could not be validated, the address line fields contain the input address without any changes. Note that the last line of the address is contained in the LastLine field. For example:</p> <p>AddressLine1: 4200 PARLIAMENT PL STE 600 LastLine: LANHAM MD 20706-1882</p> |
| AdministrativeDistrict | An area smaller than a state/province but larger than a city. |
| ApartmentLabel | The flat or unit type (such as STE or APT), for example: 123 E Main St Apt 3 |
| ApartmentNumber | The flat or unit number, for example: 123 E Main St Apt 3 |
| BlockName | An estate or block name. |
| BuildingName | The name of a building, for example Sears Tower. |
| City | The name of the town or city. For example, Vancouver , BC. |
| City.AddInfo | Additional information about the city. |
| City.SortingCode | A code used by the postal authority to speed up delivery in certain countries for large localities, for example Prague or Dublin. |
| Contact | The name of the addressee. For example, Mr. Jones . |
| Country | The country in the language or code specified in the option. |
| County | Dependent state or province information that further subdivides a state or province. An example would be a U.S. county. |
| FirmName | The name of a company. |
| Floor | Information that further subdivides a building, for example, the suite or apartment number. For example: 123 E Main St Apt 3, 4th Floor |

| Response Element | Description |
|--------------------|---|
| HouseNumber | The house number 1, for example: 298A-1B New South Head Rd |
| LastLine | Complete last address line (city, state/province, and postal code). |
| LeadingDirectional | Street directional that precedes the street name. For example, the N in 138 N Main Street. |
| Locality | Dependent place name that further subdivides a Locality. Examples are colonias in Mexico, Urbanisaciones in Spain. |
| POBox | Post Box descriptor (POBox, Postfach, Case Postale etc.) and number. |
| PostalCode | The postal code for the address. The format of the postcode varies by country. |
| PostalCode.AddOn | The second part of a postcode. For example, for Canadian addresses this will be the LDU. For U.S. addresses this is the ZIP + 4 add on. This field is not used by most countries. |
| PostalCode.Base | The base portion of the postcode. |
| Room | A room number in a building. |
| SecondaryStreet | The name of a secondary street or rural route. |
| StateProvince | The name of the state or province. |
| StreetName | The name of street where property is located, for example: 123 E Main St Apt 3 |
| StreetSuffix | The street suffix, for example: 123 E Main St Apt 3 |
| SubBuilding | A portion of a building, such as a suite. For example, Suite 102. |
| Suburb | Dependent place name that further subdivides a Locality. An example would be Mahalle in Turkey. |
| Territory | The name of a territory. Territories are larger than a state/province. |

| Response Element | Description |
|---------------------|---|
| TrailingDirectional | The trailing directional, for example: 123 Pennsylvania Ave NW |

Original Input Data

This option outputs the original input data in <FieldName>.Input fields.

Table 142: Original Input Data

| Response Element | Format | Description |
|---------------------|-------------|---------------------|
| AddressLine1.Input | String [79] | First address line |
| AddressLine2.Input | String [79] | Second address line |
| AddressLine3.Input | String [79] | Third address line |
| AddressLine4.Input | String [79] | Fourth address line |
| AddressLine5.Input | String [79] | Fifth address line |
| AddressLine6.Input | String [79] | Sixth address line |
| City.Input | String [79] | City name |
| StateProvince.Input | String [79] | State or province |

| Response Element | Format | Description |
|-----------------------|---------------|---|
| PostalCode.Input | String [79] | The postal code for the address. In the U.S. this is the ZIP Code. One of these formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999 |
| Contact.Input | String [79] | The name of the addressee. For example, "Mr. Jones". |
| Country.Input | String [79] | Specify the country using the format you chose for input country format (English name, ISO code, or UPU code). For a list of valid values, see ISO Country Codes and Module Support on page 1011. |
| FirmName.Input | String [79] | The company or firm name. |
| Street.Input | String [79] | Street |
| Number.Input | Building [79] | Number |
| Building.Input | String [79] | Building |
| SubBuilding.Input | String [79] | SubBuilding |
| DeliveryService.Input | String [79] | DeliveryService |

Result Codes

These output fields contain information about the result of the validation processing.

Table 143: Result Codes

| Response Element | Result Code |
|--------------------|---|
| AddressType | <p>For United States and Canada addresses only, the AddressType field indicates the type of address. One of the following:</p> <p>F The address was validated/corrected to the firm name.</p> <p>B The address was validated/corrected to the building name.</p> <p>G The address is a general delivery address.</p> <p>H The address was validated/corrected to the high-rise default.</p> <p>L The address is a large volume receiver.</p> <p>M The address is a military address.</p> <p>P The address was validated/corrected to PO box.</p> <p>R The address was validated/corrected to a rural route.</p> <p>S The address was validated/corrected to a street address.</p> <p>U The address could not be validated/corrected so the type is unknown.</p> |
| Confidence | <p>The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct.</p> |
| CountOverflow | <p>Indicates whether the number of candidate addresses exceeds the number returned. One of the following:</p> <p>Yes Yes, there are additional candidate addresses. To obtain the additional candidates, increase the value.</p> <p>No No, there are no additional candidates.</p> |
| ElementInputStatus | <p>ElementInputStatus provides per element information on the matching of input elements to reference data. The values in this field vary depending on whether you are using batch mode or parse mode. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 961.</p> |
| ElementRelevance | <p>Indicates which address elements are actually relevant from the local postal authority's point of view. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 961 .</p> |

| Response Element | Result Code | | | | | | | | | | | | |
|---------------------|---|----------|--|----------|------------------------------|----------|-----------------------|----------|-------------|----------|-------|----------|-----------|
| ElementResultStatus | ElementResultStatus categorizes the result in more detail than the ProcessStatus field by indicating if and how the output fields have been changed from the input fields. For information about the value in this field, see Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance on page 961. | | | | | | | | | | | | |
| MailabilityScore | <p>An estimate of how likely it is that mail sent to the address would be successful delivered. One of the following:</p> <table border="0"> <tr> <td>5</td> <td>Completely confident of deliverability</td> </tr> <tr> <td>4</td> <td>Almost certainly deliverable</td> </tr> <tr> <td>3</td> <td>Should be deliverable</td> </tr> <tr> <td>2</td> <td>Fair chance</td> </tr> <tr> <td>1</td> <td>Risky</td> </tr> <tr> <td>0</td> <td>No chance</td> </tr> </table> | 5 | Completely confident of deliverability | 4 | Almost certainly deliverable | 3 | Should be deliverable | 2 | Fair chance | 1 | Risky | 0 | No chance |
| 5 | Completely confident of deliverability | | | | | | | | | | | | |
| 4 | Almost certainly deliverable | | | | | | | | | | | | |
| 3 | Should be deliverable | | | | | | | | | | | | |
| 2 | Fair chance | | | | | | | | | | | | |
| 1 | Risky | | | | | | | | | | | | |
| 0 | No chance | | | | | | | | | | | | |
| ModeUsed | Indicates the processing mode used. The processing mode is specified in the option. For a description of the modes, see Process Options on page 539. | | | | | | | | | | | | |
| MultimatchCount | If the address was matched to multiple candidate addresses in the reference data, this field contains the number of candidate matches found. | | | | | | | | | | | | |

| Response Element | Result Code |
|------------------|-------------|
|------------------|-------------|

| | |
|---------------|--|
| ProcessStatus | |
|---------------|--|

Response Element Result Code

Provides a general description of the output quality. For a more detailed description of the output quality, see the ElementResultStatus field.

One of the following:

| | |
|-----------|--|
| V4 | Verified. The input data is correct. All elements were checked and input matched perfectly. |
| V3 | Verified. The input data is correct on input but some or all elements were standardized or the input contains outdated names or exonyms. |
| V2 | Verified. The input data is correct but some elements could not be verified because of incomplete reference data. |
| V1 | Verified. The input data is correct but the user standardization has deteriorated deliverability (wrong element user standardization - for example, postcode length chosen is too short). Not set by validation. |
| C4 | Corrected. All elements have been checked. |
| C3 | Corrected, but some elements could not be checked. |
| C2 | Corrected, but delivery status unclear (lack of reference data). |
| C1 | Corrected, but delivery status unclear because user standardization was wrong. Not set by validation. |
| I4 | Data could not be corrected completely, but is very likely to be deliverable. Single match (for example, HNO is wrong but only 1 HNO is found in reference data). |
| I3 | Data could not be corrected completely, but is very likely to be deliverable. Multiple matches (for example, HNO is wrong but more than 1 HNO is found in reference data). |
| I2 | Data could not be corrected, but there is a slim chance that the address is deliverable. |
| I1 | Data could not be corrected and is unlikely to be delivered. |
| RA | Country recognized from the Force country Setting |
| R9 | Country recognized from DefaultCountryISO3 Setting |
| R8 | Country recognized from name without errors |
| R7 | Country recognized from name with errors |
| R6 | Country recognized from territory |
| R5 | Country recognized from province |
| R4 | Country recognized from major town |
| R3 | Country recognized from format |
| R2 | Country recognized from script |
| R1 | Country not recognized - multiple matches |

| Response Element | Result Code |
|--------------------|--|
| | R0 Country not recognized |
| | S4 Parsed perfectly |
| | S3 Parsed with multiple results |
| | S2 Parsed with errors. Elements change position. |
| | S1 Parse Error. Input Format Mismatch. |
| | N1 Validation Error: No validation performed because country was not recognized. |
| | N2 Validation Error: No validation performed because required reference database is not available. |
| | N3 Validation Error: No validation performed because country could not be unlocked. |
| | N4 Validation Error: No validation performed because reference database is corrupt or in wrong format. |
| | N5 Validation Error: No validation performed because reference database is too old. |
| | N6 Validation Error: No validation performed because input data was insufficient. |
| | Q3 FastCompletion Status: Suggestions are available - complete address. |
| | Q2 FastCompletion Status: Suggested address is complete but combined with elements from the input (added or deleted). |
| | Q1 FastCompletion Status: Suggested address is not complete (enter more information). |
| | Q0 FastCompletion Status: Insufficient information provided to generate suggestions. |
| Status | Reports the success or failure of the processing attempt. null Success F Failure |
| Status.Code | The reason for the failure, if there was one. |
| Status.Description | A description of the reason for the failure, if there was one. |

Interpreting ElementInputStatus, ElementResultStatus, and ElementRelevance

The ElementInputStatus, ElementResultStatus, and ElementRelevance output fields contain a series of digits that describe the outcome of the validation operation in detail. ElementInputStatus contains some information for parsing operations.

This is what an ElementInputStatus value looks like:

44606040600000000060

This is what an ElementResultStatus value looks like:

88F0F870F00000000040

This is what an ElementRelevance value looks like:

11101010100000000000

To understand the values in these fields you need to know which element each position represents, and the meaning of the values in each position. For example, the first digit indicates the result from the PostalCode.Base output field. The position meanings are listed below.

- Position 1—PostalCode.Base
- Position 2—PostalCode.AddOn
- Position 3—City
- Position 4—Locality and Suburb
- Position 5—StateProvince
- Position 6—County
- Position 7—StreetName
- Position 8—SecondaryStreet
- Position 9—HouseNumber
- Position 10—Number level 1
- Position 11—POBox
- Position 12—Delivery service level 1
- Position 13—Building level 0
- Position 14—BuildingName
- Position 15—Sub building level 0
- Position 16—Floor and Room
- Position 17—FirmName
- Position 18—Organization level 1
- Position 19—Country
- Position 20—Territory

For ElementInputStatus, the possible values for validation are:

- 0—Empty
- 1—Not found
- 2—Not checked (no reference data)

- 3—Wrong - Set by validation only: The reference database suggests that either Number or DeliveryService is out of valid number range. Input is copied, not corrected for batch mode, for interactive mode and FastCompletion suggestions are provided.
- 4—Matched with errors in this element
- 5—Matched with changes (inserts and deletes) For example:
 - Parsing: Splitting of house number for "MainSt 1"
 - Validation: Replacing input that is an exonym or dropping superfluous fielded input that is invalid according to the country reference database
- 6—Matched without errors

For ElementInputStatus, the possible values for parsing are:

- 0—Empty
- 1—Element had to be relocated
- 2—Matched but needed to be normalized
- 3—Matched

For ElementRelevance, the possible values for parsing are:

- 0—Empty
- 1—Element had to be relocated
- 2—Matched but needed to be normalized
- 3—Matched

For ElementResultStatus, the possible values are (for all address elements apart from country):

- 0—Empty
- 1—Not validated and not changed. Original is copied.
- 2—Not validated but standardized.
- 3—Validated but not changed due to invalid input, database suggests that number is out of valid ranges. Input is copied, not corrected - this status value is only set in batch mode.
- 4—Validated but not changed due to lack of reference data.
- 5—Validated but not changed due to multiple matches. Only set in batch mode, otherwise multiple suggestions that replace the input are marked as corrected (status value 7).
- 6—Validated and changed by eliminating the input value
- 7—Validated and changed due to correction based on reference data
- 8—Validated and changed by adding value based on reference data
- 9—Validated, not changed, but delivery status not clear (for example, DPV value wrong; given number ranges that only partially match reference data).
- C—Validated, verified but changed due to outdated name
- D—Validated, verified but changed from exonym to official name
- E—Validated, verified but changed due to standardization based on casing or language. Validation only sets this status if input fully matches a language alternative.
- F—Validated, verified and not changed due to perfect match

For Country (position 19 & 20), the following values are possible:

- 0—Empty
- 1—Country not recognized
- 4—Country recognized from DefaultCountryISO3 setting
- 5—Country not recognized - multiple matches
- 6—Country recognized from script
- 7—Country recognized from format
- 8—Country recognized from major town
- 9—Country recognized from province
- C—Country recognized from territory
- D—Country recognized from name with errors
- E—Country recognized from name without errors
- F—Country recognized from ForceCountryISO3 setting

ValidateAddressLoqate

ValidateAddressLoqate standardizes and validates addresses using postal authority address data. ValidateAddress Loqate can correct information and format the address using the format preferred by the applicable postal authority. It also adds missing postal information, such as postal codes, city names, state/province names, and so on.

ValidateAddressLoqate also returns result indicators about validation attempts, such as whether or not ValidateAddressLoqate validated the address, the level of confidence in the returned address, the reason for failure if the address could not be validated, and more.

During address matching and standardization, ValidateAddressLoqate separates address lines into components and compares them to the contents of the Spectrum Universal Address databases. If a match is found, the input address is *standardized* to the database information. If no database match is found, ValidateAddressLoqate optionally *formats* the input addresses. The formatting process attempts to structure the address lines according to the conventions of the appropriate postal authority.

ValidateAddressLoqate is part of Spectrum Universal Address.

Resource URL

```
http://server:port/soap/ValidateAddressLoqate
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:val="http://www.precisely.com/spectrum/services/ValidateAddressLoqate"
```

```

xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <val:ValidateAddressLoqateRequest>
      <val:input_port>
        <val:Address>
          <val:AddressLine1>1825 Kramer Ln</val:AddressLine1>
          <val:City>Austin</val:City>
          <val:StateProvince>TX</val:StateProvince>
        </val:Address>
      </val:input_port>
    </val:ValidateAddressLoqateRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

This would be the response:

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:ValidateAddressLoqateResponse
xmlns:ns2="http://spectrum.precisely.com/"

xmlns:ns3="http://www.precisely.com/spectrum/services/ValidateAddressLoqate">

      <ns3:output_port>
        <ns3:Address>
          <ns3:Confidence>95</ns3:Confidence>
          <ns3:CouldNotValidate/>
          <ns3:ProcessedBy>LOQATE</ns3:ProcessedBy>
          <ns3:MatchScore>100.0</ns3:MatchScore>
          <ns3:AddressLine1>1825 Kramer Ln</ns3:AddressLine1>
          <ns3:AddressLine2/>
          <ns3:City>Austin</ns3:City>
          <ns3:StateProvince>TX</ns3:StateProvince>
          <ns3:PostalCode>78758-4260</ns3:PostalCode>
          <ns3:PostalCode.Base>78758</ns3:PostalCode.Base>
          <ns3:PostalCode.AddOn>4260</ns3:PostalCode.AddOn>
          <ns3:Country>United States</ns3:Country>
          <ns3:FirmName/>
          <ns3:user_fields/>
        </ns3:Address>
      </ns3:output_port>
    </ns3:ValidateAddressLoqateResponse>
  </soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

Table 144: Input Format

| Parameter | Format | Description |
|---------------|--------|--|
| AddressLine1 | String | The first address line. |
| AddressLine2 | String | The second address line. |
| AddressLine3 | String | The third address line. |
| AddressLine4 | String | The fourth address line. |
| City | String | The city name. |
| Country | String | <p>The country code or name, in any of the following formats:</p> <ul style="list-style-type: none"> • Two-character ISO 3166-1 Alpha 2 country code • Three-character ISO 3166-1 Alpha 3 country code • English country name <p>See ISO Country Codes and Module Support on page 1011 for a list of ISO codes.</p> |
| FirmName | String | The company or firm name. |
| PostalCode | String | <p>The postal code for the address in one of these formats:</p> <p>99999 99999-9999 A9A9A9 A9A 9A9 9999 999</p> |
| StateProvince | String | The state or province. |

Address Line Processing for U.S. Addresses

The input fields AddressLine1 through AddressLine4 are handled differently for U.S. addresses depending on whether the firm name extraction or urbanization code extraction options are enabled. If either of these options is enabled, ValidateAddressLoqate will look at the data in all four fields to validate the address and extract the requested data (firm name and/or urbanization code). If neither of these options is enabled, ValidateAddressLoqate uses only the first two non-blank address line fields in its validation attempt. The data in the other address line fields is returned in the output field AdditionalInputData. For example,

AddressLine1: A1 Calle A

AddressLine2:

AddressLine3: URB Alamar

AddressLine4: Precisely

In this address, if either firm name extraction or urbanization code extraction were enabled, ValidateAddressLoqate would examine all four address lines. If neither firm name extraction nor urbanization code extraction were enabled, ValidateAddressLoqate would examine AddressLine1 and AddressLine3 (the first two non-blank address lines) and attempt to validate the address using that data; the data in AddressLine4 would be returned in the output field AdditionalInputData.

Options

The following table lists the options that control the type of information returned by ValidateAddressLoqate.

Table 145: Output Data Options

| Parameter | Description | | | | |
|-----------------------------|--|----------|---|----------|---------------------------------------|
| Database.Loqate | Specifies which database you want to use for validating international addresses. To specify a database for international address validation, select a database in the Database drop-down list. | | | | |
| OutputFieldLevelReturnCodes | Specifies whether to include field-level result indicators. Field-level result indicators describe how ValidateAddressLoqate handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in HouseNumber.Result . For a complete listing of result indicator output fields, see Result Indicators on page 573. <table border="0" data-bbox="682 1711 1347 1795"> <tr> <td>N</td> <td>No, do not output field-level return codes (default).</td> </tr> <tr> <td>Y</td> <td>Yes, output field-level return codes.</td> </tr> </table> | N | No, do not output field-level return codes (default). | Y | Yes, output field-level return codes. |
| N | No, do not output field-level return codes (default). | | | | |
| Y | Yes, output field-level return codes. | | | | |

| Parameter | Description |
|-----------------------|--|
| OutputFormattedOnFail | <p data-bbox="685 380 1425 520">Specifies whether to return a formatted address when an address cannot be validated. The address is formatted using the preferred address format for the address's country. If this option is not selected, the output address fields are blank when ValidateAddressLoqate cannot validate the address.</p> <p data-bbox="685 541 1284 617"> N No, do not format failed addresses (default). Y Yes, format failed addresses. </p> <p data-bbox="685 646 1425 940">Formatted addresses are returned using the format specified by the Include a standard address, Include address line elements, and Include postal information check boxes. Note that if you select Include address line elements, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, select Include standardized input address elements.</p> <p data-bbox="685 961 1425 1016">If you check this option, you must select Include a standard address and/or Include address line elements.</p> <p data-bbox="685 1037 1425 1297">Formatted addresses are returned using the format specified by the OutputRecordType option. Note that if you specify OutputRecordType=E, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, specify OutputRecordType=I.</p> <p data-bbox="685 1318 1425 1339">If you specify Y, you must specify "A" and/or "E" for OutputRecordType.</p> <p data-bbox="685 1360 1425 1619">Formatted addresses are returned using the format specified by the Option.OutputRecordType option. Note that if you specify Option.OutputRecordType=E, the parsed address elements will contain the parsed, validated address for addresses that could be validated. If the address could not be validated the parsed address elements will contain the input address in parsed form. If you always want the output to contain the input address in parsed form, regardless of whether or not ValidateAddressLoqate could validate the address, specify Option.OutputRecordType=I.</p> |

| Parameter | Description |
|---------------------|--|
| OutputAddressBlocks | <p data-bbox="685 380 1427 495">Specifies whether to return a formatted version of the address as it would be printed on a physical mailpiece. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9.</p> <p data-bbox="685 516 1024 543">For example, this input address:</p> <p data-bbox="685 564 1084 705">AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p data-bbox="685 726 1068 753">Results in this address block output:</p> <p data-bbox="685 774 1203 863">AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA</p> <p data-bbox="685 884 1427 1031">ValidateAddressLoqate formats the address into address blocks using postal authority standards. The country name is returned using the Universal Postal Union country name. Note that the option does not affect the country name in the address block, it only affects the name returned in the Country output field.</p> <p data-bbox="685 1052 902 1079">One of the following:</p> |

| Parameter | Description |
|----------------|--|
| AmasFormatting | <p>Specifies that output address data is to be formatted using Address Matching Approval System (AMAS) conventions.</p> <p>This option causes Validate Address Loqate to use AMAS rules when standardizing an address. AMAS is an Australia Post program for enforcing addressing standards. For more information on the AMAS formatting conventions, refer to the Address Matching Approval System (AMAS) Handbook.</p> <p>This option modifies the output data as follows.</p> <ul style="list-style-type: none"> • Numeric fields are padded with zeros. This affects the following output fields: HouseNumber, HouseNumber2, PostalDeliveryNumber, and DPID. For example, if the input address is 298 New South Head Rd Double Bay NSW 2028, then the format of the HouseNumber field is changed from 298 to 00298. • If a match is not made, then all digits in the DPID field will be zero. For example, 00000000. • If a match is not made, then all return fields (parsed address elements) will be blank, except numeric fields which will contain all zeros. • The CCD field is not output. <p>Valid values are:</p> <p>N No, do not format the output data using AMAS conventions (default).</p> <p>Y Yes, format the output data using AMAS conventions.</p> <p>Note: When this option is selected, results will be returned with AMAS formatting regardless of selections made in the Acceptance level and Minimum match score fields.</p> |
| OutputCasing | <p>Specifies the casing of the output data. One of the following:</p> <p>M Returns the output in mixed case (default). For example:</p> <p style="margin-left: 40px;">123 Main St Mytown FL 12345</p> <p>U Returns the output in upper case. For example:</p> <p style="margin-left: 40px;">123 MAIN ST MYTOWN FL 12345</p> |

| Parameter | Description |
|-------------|--|
| HomeCountry | <p>Specifies the default country. You should specify the country where most of your addresses reside. For example, if most of the addresses you process are in Germany, specify Germany. ValidateAddressLoqate uses the country you specify to attempt validation when it cannot determine the country from the StateProvince, PostalCode, and Country address fields. The valid country names are:</p> <p>Afghanistan, Albania, Algeria, American Somoa, Andorra, Angola, Anguilla, Antigua And Barbuda, Argentina, Armenia, Aruba, Australia, Austria, Azerbaijan, Bahamas, Bahrain, Bangladesh, Barbados, Belarus, Belgium, Belize, Benin, Bermuda, Bhutan, Bolivia, Bosnia And Herzegovina, Botswana, Brazil, British Virgin Islands, Brunei Darussalam, Bulgaria, Burkina Faso, Burundi, Cambodia, Cameroon, Canada, Cape Verde, Cayman Islands, Central African Republic, Chad, Chile, China, Colombia, Comoros Islands, Congo, Cook Islands, Costa Rica, Cote D'Ivoire, Croatia, Cuba, Cyprus, Czech Republic, Democratic Republic Of Congo, Denmark, Djibouti, Dominica, Dominican Republic, East Timor, Ecuador, Egypt, El Salvador, Equitorial Guinea, Eritrea, Estonia, Ethiopia, Falkland Islands, Faroe Islands, Federated States Of Micronesia, Fiji, Finland, France, French Guiana, Gabon, Gambia, Germany, Ghana, Gibraltar, Greece, Greenland, Grenada, Guadeloupe, Guam, Guatemala, Guinea, Guinea Bissau, Guyana, Haiti, Holy See, Honduras, Hong Kong, Hungary, Iceland, India, Indonesia, Iran, Iraq, Ireland, Israel, Italy, Jamaica, Japan, Jordan, Kazakhstan, Kenya, Kiribati, Korea, Kuwait, Kyrgyzstan, Laos, Latvia, Lebanon, Lesotho, Liberia, Libya, Liechtenstein, Lithuania, Luxembourg, Macau, Macedonia, Madagascar, Malawi, Malaysia, Maldives, Mali, Malta, Marshall Islands, Martinique, Mauritania, Mauritius, Mayotte, Mexico, Moldova, Monaco, Mongolia, Monserrat, Morocco, Mozambique, Myanmar, Namibia, Nauru, Nepal, Netherlands Antilles, New Caledonia, New Zealand, Nicaragua, Niger, Nigeria, Niue, Norway, Oman, Pakistan, Palau, Panama, Papua New Guinea, Paraguay, Peru, Philippines, Pitcairn Islands, Poland, Portugal, Puerto Rico, Qatar, Republic Of Georgia, Republic Of Korea, Republic Of Singapore, Reunion, Romania, Russia, Rwanda, Saint Helena, Saint Kitts And Nevis, Saint Lucia, Saint Pierre And Miquelon, Saint Vincent And The Grenadines, Samoa, San Marino, Sao Tome and Principe, Saudi Arabia, Senegal, Seychelles, Sierra Leone, Slovakia, Slovenia, Solomon Islands, Somalia, South Africa, Spain, Sri Lanka, Sudan, Surivalue, Swaziland, Sweden, Switzerland, Syria, Tahiti, Taiwan, Tajikistan, Tanzania, Thailand, The Netherlands, Togo, Tonga, Trinidad And Tobago, Tristan Da Cunha, Tunisia, Turkey, Turkmenistan, Turks And Caicos Islands, Tuvalu, Uganda, Ukraine, United Arab Emirates, United Kingdom, United States, Uruguay, Uzbekistan, Vanuatu, Venezuela, Vietnam, Virgin Islands (US), Wallis And Futuna, Yemen, Yugoslavia, Zambia, Zimbabwe</p> |

| Parameter | Description |
|---------------------|--|
| OutputCountryFormat | <p>Specifies the format to use for the country name returned in the Country output field. For example, if you select English, the country name "Deutschland" would be returned as "Germany".</p> <p>E Use English country names (default).</p> <p>I Use two-letter ISO abbreviation for the countries instead of country names.</p> <p>U Use Universal Postal Union abbreviation for the countries instead of country names.</p> |
| OutputScript | <p>Specifies the alphabet or script in which the output should be returned. This option is bi-directional and generally takes place from Native to Latin and Latin to Native.</p> <p>Input Do not perform transliteration and provide output in the same script as the input (default).</p> <p>Native Output in the native script for the selected country wherever possible.</p> <p>Latn Use English values.</p> |

| Parameter | Description |
|-------------------------------------|--|
| Acceptance level AcceptanceLevel | <p data-bbox="685 380 1425 491">Specifies the minimum verification level a record must reach to be considered successfully processed. The value in this field corresponds to the second character of the Address Verification Code, which is called "Post-Processed Verification Match Level":</p> <ul data-bbox="685 520 1425 1633" style="list-style-type: none"> <li data-bbox="685 520 1425 779">• 5—Delivery point (building or post box). The record will be passed or will have high confidence if ApartmentNumber, HouseNumber, Street, City, and StateProvince supplied in the input record match to the Loqate reference dataset. Will have moderate confidence if ApartmentNumber is correct but other remaining fields are incorrect, but in this case the Loqate engine should be able to identify the ApartmentNumber as ApartmentNumber is at a more granular level. It will have zero confidence if ApartmentNumber and other fields are unable to be parsed by the Loqate engine. <li data-bbox="685 789 1425 1024">• 4—Premise or building. The record will be passed or will have high confidence if House Number, Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if HouseNumber is correct but the other fields are not; however, in this case the Loqate engine should be able to identify the HouseNumber because HouseNumber is at a more granular level. It will have zero confidence if the HouseNumber and other fields are unable to be parsed by the Loqate engine. <li data-bbox="685 1035 1425 1270">• 3—Thoroughfare, road, or street. The record will be passed or will have high confidence if Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and State Province) are unable to be parsed by the Loqate engine. <li data-bbox="685 1281 1425 1516">• 2—Locality (city or town). The record will be passed or will have high confidence if both City and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate Engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and StateProvince) are unable to be parsed by the Loqate engine. <li data-bbox="685 1526 1425 1602">• 1—Administrative area (state or region). The record will be passed or will have high confidence if the StateProvince supplied in the input record matches the Loqate reference dataset. <li data-bbox="685 1612 1425 1633">• 0—None. This is equivalent to loosest match option. |

| Parameter | Description |
|-------------------------------|---|
| IsDuplicateHandlingMaskEnable | <p>Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options:</p> <p>S Selected by default. Preprocess the input and remove duplicates that occur in a single field.</p> <p>C Selected by default. Pre-process the input and remove duplicates across all fields.</p> <p>T Pre-process the input and remove duplicates in fields that are not standard address fields.</p> <p>F Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.</p> |
| MinimumMatchScore | <p>Specifies a numeric value between 0 and 100 that indicates the degree to which Validate Address Loqate will change an address in order to obtain a match in the Loqate reference database. The lower the number, the greater amount of change is allowed. A value of 100 means that after parsing the input address is nearly identical to the validated address. A value of 0 means that the parsed input address may be completely changed in order to obtain a validated address.</p> |
| KeepMultimatch | <p>Specifies whether or not to return multiple address for those input addresses that have more than one possible match.</p> <p>Y Yes, return multiple matches (default).</p> <p>N No, do not return multiple matches.</p> <p>For more information, see Returning Multiple Matches on page 564.</p> |
| FailMultipleMatches | <p>Fails multiple addresses for those input addresses that have more than one possible match.</p> |

Returning Multiple Matches

If ValidateAddressLoqate finds multiple address in the postal database that are possible matches for the input address, you can have ValidateAddressLoqate return the possible matches. For example, the following address matches multiple addresses in the U.S. postal database:

PO BOX 1 New York, NY

Options

To return multiple matches, use the options described in the following table.

Table 146: Multiple Match Option

| Description/Valid Values |
|---|
| Indicates whether or not to return multiple address for those input addresses that have more than one possible match. |
| number between 1 and 10 that indicates the maximum number of addresses to return. The default value is 1. Note: The difference between <code>multipleMatch</code> and <code>multipleMatchLimit</code> is that a <code>multipleMatch</code> will return a failure if, whereas a <code>multipleMatchLimit</code> will return one record if. |
| To identify which output addresses are candidate addresses, you must. When you do this, records that are candidate addresses will have one or more "M" values in the field-level result indicators. |

Output

When you choose to return multiple matches, the addresses are returned in the address format you specify. For information on specifying address format, see [Options](#) on page 556. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information, see [Result Indicators](#) on page 573.

Match Score Threshold Options

There are two options for setting match score thresholds.

Note: These options are not available in the Validate Address Loqate user interface; they are located in the following file:

```
SpectrumDirectory/server/modules/loqate/env.properties
```

The **MatchScoreAbsoluteThreshold** option is used to specify the minimum match score a record must reach to be considered a candidate for matching. The default value is 60, and the maximum value is 100.

The **MatchScoreThresholdFactor** is a value that represents a factor of the highest matching result. This value is used as a cutoff for considering result candidates. The higher the value of the factor, the higher the chance of getting a good verification result. The default value is 95 and the maximum value is 100.

Response

The output from `ValidateAddressLoqate` contains various information depending on the output categories you select.

Standard Address Output

Standard address output consists of four lines of the address which correspond to how the address would appear on an address label. City, state/province, postal code, and other data is also included in standard address output. `ValidateAddressLoqate` returns standard address output for validated addresses if you. Standard address fields are always returned for addresses that could not be validated regardless of whether or not you. For non-validated addresses, the standard address output fields contain the address as it appeared in the input ("pass through" data). If you want `ValidateAddressLoqate` to standardize address according to postal authority standards when validation fails,.

Table 147: Standard Address Output

| Response Element | Description |
|----------------------------------|--|
| <code>AdditionalInputData</code> | Input data that could not be matched to a particular address component. For more information, see About Additional Input Data . |
| <code>AddressLine1-4</code> | If the address was validated, the first line of the validated and standardized address. If the address could not be validated, the first line of the input address without any changes. There can be up to four address block output fields: <code>AddressLine1</code> through <code>AddressLine4</code> . |
| <code>City</code> | The validated city name. |
| <code>Country</code> | The country in the format determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| <code>FirmName</code> | The validated firm or company name. |
| <code>PostalCode</code> | The validated ZIP Code™ or postal code. |

| Response Element | Description |
|------------------|---|
| PostalCode.AddOn | The 4-digit add-on part of the ZIP Code™. For example, in the ZIP Code™ 60655-1844, 1844 is the 4-digit add-on. |
| PostalCode.Base | The 5-digit ZIP Code™; for example 20706. |
| StateProvince | The validated state/province or its abbreviated value. |

Parsed Address Elements Output

Output addresses are formatted in the parsed address format if you. If you want `ValidateAddressLoqate` to return formatted data in the Parsed Address format when validation fails (that is, a normalized address),.

Note: If you want `ValidateAddressLoqate` to always return parsed input data regardless of whether or not validation is successful,. For more information, see [Parsed Input](#) on page 569.

Table 148: Parsed Address Output

| Response Element | Description |
|------------------|---|
| AddressBlock1-9 | <p>The AddressBlock output fields contain a formatted version of the standardized or normalized address as it would be printed on a physical mailpiece. Validate Address Global formats the address into address blocks using postal authority standards. Each line of the address is returned in a separate address block field. There can be up to nine address block output fields: AddressBlock1 through AddressBlock9. For example, this input address:</p> <p>AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706</p> <p>Results in this address block output:</p> <p>AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882</p> |

| Response Element | Description |
|--------------------|--|
| ApartmentLabel | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| ApartmentNumber | Apartment number, for example: 123 E Main St APT 3 |
| ApartmentNumber2 | Secondary apartment number, for example: 123 E Main St APT 3, 4th Floor Note: In this release, this field will always be blank. |
| Building | Descriptive name identifying an individual location. |
| City | Validated city name |
| Country | Country. Format is determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| County* | The smallest geographic data element within a country, for instance, USA County |
| FirmName | The validated firm or company name |
| HouseNumber | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional | Leading directional, for example: 123 E Main St Apt 3 |
| POBox | Post office box number. If the address is a rural route address, the rural route box number will appear here. |

| Response Element | Description |
|---------------------|---|
| PostalCode | Validated postal code. For U.S. addresses, this is the ZIP Code. |
| Principality * | The largest geographic data element within a country |
| StateProvince | Validated state or province name |
| StreetAlias | Alternate street name; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. for example: 123 E Main St Apt 3 |
| StreetName | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix | Street suffix, for example: 123 E Main St Apt 3 |
| Subcity* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet* | The dependent street or block data element within a country. For instance, UK Dependent Street . |
| TrailingDirectional | Trailing directional, for example: 123 Pennsylvania Ave NW |

*This is a subfield and may not contain data.

Parsed Input

The output can include the input address in parsed form. This type of output is referred to as "parsed input." Parsed input fields contain the address data that was used as input regardless of whether or not ValidateAddress validated the address. Parsed input is different from the "parsed address elements" output in that parsed address elements contain the validated address if the address could

be validated, and, optionally, the input address if the address could not be validated. Parsed input always contains the input address regardless of whether or not `ValidateAddress` validated the address.

To include parsed input fields in the output,.

Table 149: Parsed Input

| Response Element | Description |
|--------------------------|---|
| ApartmentLabel.Input | Apartment designator (such as STE or APT), for example: 123 E Main St APT 3 |
| ApartmentNumber.Input | Apartment number, for example: 123 E Main St APT 3 |
| City.Input | Validated city name |
| Country.Input | Country. Format is determined by what you selected in : <ul style="list-style-type: none"> • ISO Code • UPU Code • English |
| County.Input* | The smallest geographic data element within a country, for instance, USA County |
| FirmName.Input | The validated firm or company name |
| HouseNumber.Input | House number, for example: 123 E Main St Apt 3 |
| LeadingDirectional.Input | Leading directional, for example: 123 E Main St Apt 3 |
| POBox.Input | Post office box number. If the address is a rural route address, the rural route box number will appear here. |
| PostalCode.Input | Validated postal code. For U.S. addresses, this is the ZIP Code. |

| Response Element | Description |
|---------------------------|---|
| Principality.Input * | The largest geographic data element within a country |
| StateProvince.Input | Validated state or province name |
| StreetAlias.Input | Alternate street name; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. The base name is the name that applies to the entire street. For example: If StreetName is "N MAIN ST" the StreetAlias field would contain "MAIN" and the thoroughfare type, "ST", would be returned in the StreetSuffix field. |
| StreetName.Input | Street name, for example: 123 E Main St Apt 3 |
| StreetSuffix.Input | Street suffix, for example: 123 E Main St Apt 3 |
| Subcity.Input* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet.Input* | The dependent street or block data element within a country. For instance, UK Dependent Street . |
| TrailingDirectional.Input | Trailing directional, for example: 123 Pennsylvania Ave NW |

*This is a subfield and may not contain data.

Geocode Output

ValidateAddressLoqate returns the latitude/longitude, geocoding match code, dependent and double dependent localities, dependent thoroughfare, subadministrative and superadministrative areas, and the search distance as output. Match codes describe how well the geocoder matched the input address to a known address; they also describe the overall status of a match attempt. Search distance codes represent how close the geocode is to the actual physical location of an address.

Table 150: Geocode Address Output

| Response Element | Description |
|-------------------|--|
| Geocode.MatchCode | <p>This two-byte code reflects the status and level of geocode matching for an address. The first byte represents the geocoding status and is one of the following:</p> <p>A Multiple candidate geocodes were found to match the input address, and an average of these was returned</p> <p>I A geocode was able to be interpolated from the input addresses location in a range</p> <p>P A single geocode was found matching the input address</p> <p>U A geocode was not able to be generated for the input address</p> <p>The second byte represents the level of geocoding matching and is one of the following:</p> <p>5 Delivery point (post box or subbuilding)</p> <p>4 Premise or building</p> <p>3 Thoroughfare</p> <p>2 Locality</p> <p>1 Administrative area</p> <p>0 None</p> |
| Latitude | Eight-digit number in degrees and calculated to five decimal places (in the format specified). |
| Longitude | Eight-digit number in degrees and calculated to five decimal places (in the format specified). |
| SearchDistance | The radius of accuracy in meters, providing an indication of the probable maximum distance between the given geocode and the actual physical location. This field is derived from and dependent upon the accuracy and coverage of the underlying reference data. |

Table 151: City/Street/Postal Code Centroid Match Codes

| Element | Match Code |
|----------------------------|------------|
| Address Point | P4 |
| Address Point Interpolated | I4 |
| Street Centroid | A4/P3 |
| Postal Code/City Centroid | A3/P2/A2 |

Note: Geocode.Match.Code does not return two coordinates for a street segment (such as the beginning and ending of a portion of a street). Instead, with input resulting in return codes of I3 (interpolated to thoroughfare or street level, where no input premise number was provided), the complete street is used in the computation.

Result Indicators

Result indicators provide information about the kinds of processing performed on an address. There are two types of result indicators:

Record-Level Result Indicators

Record-level result indicators provide data about the results of ValidateAddressLoqate processing for each record, such as the success or failure of the match attempt, which coder processed the address, and other details. The following table lists the record-level result indicators returned by ValidateAddressLoqate.

Table 152: Record Level Indicators

| Response Element | Description |
|------------------|--|
| Confidence | The level of confidence assigned to the address being returned. Range is from zero (0) to 100; zero indicates failure, 100 indicates a very high level of confidence that the match results are correct. For multiple matches, the confidence level is 0. For details about how this number is calculated, see Introduction to the Validate Address Confidence Algorithm on page 1036. |

| Response Element | Description |
|------------------|--|
| CouldNotValidate | <p>If no match was found, which address component could not be validated:</p> <ul style="list-style-type: none"> • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber <p>Note: More than one component may be returned, in a comma-separated list.</p> |
| MatchScore | <p>MatchScore provides an indication of the similarity between the input data and the closest reference data match. It is significantly different from Confidence in that Confidence indicates how much the input address changed to obtain a match, whereas the meaning of Match Score varies between U.S. and non-U.S. addresses.</p> <p>The int getFieldMatchscore (unit record, const char*) field is a decimal value between 0 and 100 that reflects the similarity between the identified input data and the closest reference data match. A result of 100 indicates that no changes other than alias, casing, or diacritic changes have been made to the input data. A result of 0 indicates that there is no similarity between the input data and closest reference data match.</p> <p>Note: The Validate Address Loqate and Advanced Matching Module components both use the MatchScore field. The MatchScore field value in the output of a dataflow is determined by the last stage to modify the value before it is sent to an output stage. If you have a dataflow that contains Validate Address Loqate and Advanced Matching Module components and you want to see the MatchScore field output for each stage, use a Transformer stage to copy the MatchScore value to another field. For example, Validate Address Loqate produces an output field called MatchScore and then a Transformer stage copies the MatchScore field from Validate Address Loqate to a field called AddressMatchScore. When the matcher stage runs it populates the MatchScore field with the value from the matcher and passes through the AddressMatchScore value from Validate Address Loqate.</p> |
| ProcessedBy | <p>Which address coder processed the address:</p> <p>LOQATE The Loqate coder processed the address.</p> |

| Response Element | Description | | | | |
|--------------------------|---|--------------------------|---|----------|---------|
| Status | <p>Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches.</p> <table> <tr> <td>null</td> <td>Success</td> </tr> <tr> <td>F</td> <td>Failure</td> </tr> </table> | null | Success | F | Failure |
| null | Success | | | | |
| F | Failure | | | | |
| Status.Code | <p>Reason for failure, if there is one.</p> <ul style="list-style-type: none"> UnableToValidate | | | | |
| Status.Description | <p>Description of the problem, if there is one.</p> <table> <tr> <td>Address Not Found</td> <td>This value will appear if Status.Code=UnableToValidate.</td> </tr> </table> | Address Not Found | This value will appear if Status.Code=UnableToValidate. | | |
| Address Not Found | This value will appear if Status.Code=UnableToValidate. | | | | |

Field-Level Result Indicators

Field-level result indicators describe how ValidateAddressLoqate handled each address element. Field-level result indicators are returned in the qualifier "Result". For example, the field-level result indicator for HouseNumber is contained in **HouseNumber.Result**.

To enable field-level result indicators, .

The following table lists the field-level result indicators. If a particular field does not apply to an address, the result indicator may be blank.

Table 153: Field-Level Result Indicators

| Response Element | Description |
|------------------------|--|
| ApartmentLabel.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. |
| | R The apartment label is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| ApartmentNumber.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. addresses that are an EWS match will have a value of P. U.S. and Canadian addresses only. |
| | R The apartment number is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|------------------|---|
| City.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Hyphens missing or punctuation errors. Canadian addresses only. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. |
| | R The city is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| Country.Result | These result codes do not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| County.Result* | The smallest geographic data element within a country, for instance, USA County |
| FirmName.Result | C Corrected. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only. |
| | U Unmatched. U.S. and Canadian addresses only. |
| | V Validated. The data was confirmed correct and remained unchanged from input. U.S. addresses only. |

| Response Element | Description |
|---------------------------|--|
| HouseNumber.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | O Out of range. Does not apply to U.S. or Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | R The house number is required but is missing from the input address. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| LeadingDirectional.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. Non-blank input was corrected to a non-blank value. U.S. addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. Does not apply to Canadian addresses. |

| Response Element | Description |
|-------------------|---|
| POBox.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple matches. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | R The P.O. Box number is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| PostalCode.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to Canadian addresses. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. |
| | R The postal code is required but is missing from the input address. U.S. addresses only. |
| | S Standardized. This option includes any standard abbreviations. Does not apply to U.S. or Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

| Response Element | Description |
|-----------------------|--|
| PostalCode.Type | <p>P The ZIP Code™ contains only PO Box addresses. U.S. addresses only.</p> <p>U The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.</p> <p>M The ZIP Code™ is for military addresses. U.S. addresses only.</p> <p>null The ZIP Code™ is a standard ZIP Code™.</p> |
| Principality.Result * | The largest geographic data element within a country |
| StateProvince.Result | <p>A Appended. The field was added to a blank input field. U.S. and Canadian addresses only.</p> <p>C Corrected. U.S. addresses only.</p> <p>M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. Does not apply to U.S. or Canadian addresses.</p> <p>P Pass-through. The data was not used in the validation process, but it was preserved in the output. U.S. and Canadian addresses only.</p> <p>R The state is required but is missing from the input address. U.S. addresses only.</p> <p>S Standardized. This option includes any standard abbreviations. Does not apply to U.S. addresses.</p> <p>U Unmatched. Does not apply to Canadian addresses.</p> <p>V Validated. The data was confirmed correct and remained unchanged from input.</p> |
| StreetAlias.Result | An alternate name for a street; typically applies only to a specific range of addresses on the street. If you do not allow street aliases in the output then the street's "base" name will appear in the output regardless of whether or not there is an alias for the street. The base name is the name that applies to the entire street. For example: If StreetName is "N MAIN ST" the StreetAlias field would contain "MAIN" and the thoroughfare type,"ST", would be returned in the StreetSuffix field. |

| Response Element | Description |
|---------------------|--|
| StreetName.Result | A Appended. The field was added to a blank input field. Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Does not apply to U.S. addresses. |
| | S Standardized. This option includes any standard abbreviations. U.S. and Canadian addresses only. |
| | U Unmatched. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| StreetSuffix.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to U.S. addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |
| Subcity.Result* | A smaller population center data element, dependent on the contents of the Locality field. For instance, Turkish Neighbourhood . |
| Substreet.Result* | The dependent street or block data element within a country. For instance, UK Dependent Street . |

| Response Element | Description |
|----------------------------|--|
| TrailingDirectional.Result | A Appended. The field was added to a blank input field. U.S. and Canadian addresses only. |
| | C Corrected. U.S. and Canadian addresses only. |
| | F Formatted. The spacing and/or punctuation was changed to conform to postal standards. Does not apply to U.S. or Canadian addresses. |
| | M Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. U.S. addresses only. |
| | P Pass-through. The data was not used in the validation process, but it was preserved in the output. Canadian addresses only. |
| | S Standardized. This option includes any standard abbreviations. |
| | U Unmatched. Does not apply to Canadian addresses. |
| | V Validated. The data was confirmed correct and remained unchanged from input. |

*This is a subfield and may not contain data.

The AVC Code

The *Address Verification Code (AVC)* is an 11-byte code that is made up of accuracy indicators for addresses; the codes tell you the quality of the processing results and provide guidelines on how to correct the input data if necessary. Each individual address receives its own code. This code is automatically returned within your dataflow output. An example of an AVC is:

V44-I44-P6-100

An AVC has eight parts:

- Verification Status
- Post-Process Verification Match Level
- Pre-Process Verification Match Level
- Parsing Status
- Lexicon Identification Match Level
- Context Identification Match Level
- Postcode Status
- Matchscore

Verification Status

The level to which an address was verified.

- **V**—Verified. A complete match was made between the input data and a single record from the available reference data. For simple address validation, this is considered the best code to return.
- **P**—Partially verified. A partial match was made between the input data and a single record from the available reference data. This could mean that there is granular data for the address information that was provided, but additional information is required to return a full validation.
- **A**—Ambiguous. There are multiple addresses that could match the input.
- **U**—Unable to verify. This gets returned when there is not enough information to verify an address or when the input query is unreadable. The output fields will contain the input data.
- **R**—Reverted. The record could not be verified to the specified minimum acceptable level. This occurs when advanced options such as minimum reversion levels are set on a process. The output fields will contain the input data.
- **C**—Conflict. There is more than one close reference data match with conflicting values.

Post-Process Verification Match Level

The level to which the input data matches the available reference data after processing.

- **5**—Delivery point (building or post box). The record will be passed or will have high confidence if ApartmentNumber, HouseNumber, Street, City, and StateProvince supplied in the input record match to the Loqate reference dataset. Will have moderate confidence if ApartmentNumber is correct but other remaining fields are incorrect, but in this case the Loqate engine should be able to identify the ApartmentNumber as ApartmentNumber is at a more granular level. It will have zero confidence if ApartmentNumber and other fields are unable to be parsed by the Loqate engine.
- **4**—Premise or building. The record will be passed or will have high confidence if House Number, Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if HouseNumber is correct but the other fields are not; however, in this case the Loqate engine should be able to identify the HouseNumber because HouseNumber is at a more granular level. It will have zero confidence if the HouseNumber and other fields are unable to be parsed by the Loqate engine.
- **3**—Thoroughfare, road, or street. The record will be passed or will have high confidence if Street, City, and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and State Province) are unable to be parsed by the Loqate engine.
- **2**—Locality (city or town). The record will be passed or will have high confidence if both City and StateProvince supplied in the input record match the Loqate reference dataset. Will have moderate confidence if City is correct but StateProvince is not; however, in this case the Loqate Engine should be able to identify the StateProvince as City itself is the part of StateProvince. It will have zero confidence if City or both fields (City and StateProvince) are unable to be parsed by the Loqate engine.
- **1**—Administrative area (state or region). The record will be passed or will have high confidence if the StateProvince supplied in the input record matches the Loqate reference dataset.
- **0**—None. This is equivalent to loosest match option.

Pre-Process Verification Match Level

The level to which the input data matches the available reference data before processing.

- **5**—Delivery point (building or post box)
- **4**—Premise or building.
- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Parsing Status

The level to which an address was parsed.

- **I**—Identified and parsed. The input data has been identified and placed into components. For example, with "123 Kingston Av" Validate Address Loqate would be able to determine that "123" was a Premise Number, "Kingston" was the Thoroughfare Name, and "Av" or "Avenue" would be the Thoroughfare Type.
- **U**—Unable to parse. Validate Address Loqate was unable to identify and parse the input data. As with the "Unverified" verification status, the input data was incomplete or vague.

Lexicon Identification Match Level

The level to which the input data has some recognized form through the use of pattern matching (for instance, a numeric value could be a premise number) and lexicon matching (for example, "rd" could be Thoroughfare Type "road"; "London" could be a locality, and so on).

- **5**—Delivery point (building or post box)
- **4**—Premise or building.
- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Context Identification Match Level

The level to which the input data can be recognized based on the context in which it appears. This is the least accurate form of matching and is based on identifying a word as a particular address element. For example, input could be determined to be a thoroughfare because it was preceded by something that could be a premise and followed by something that could be a locality, the latter items being identified through a match against the reference data or the lexicon.

- **5**—Delivery point (building or post box)
- **4**—Premise or building.

- **3**—Thoroughfare, road, or street.
- **2**—Locality (city or town).
- **1**—Administrative area (state or region).
- **0**—None.

Postcode Status

The level to which a postal code was verified.

- **P8**—PostalCodePrimary and PostalCodeSecondary verified.
- **P7**—PostalCodePrimary verified, PostalCodeSecondary added or changed.
- **P6**—PostalCodePrimary verified.
- **P5**—PostalCodePrimary verified with small change.
- **P4**—PostalCodePrimary verified with large change.
- **P3**—PostalCodePrimary added.
- **P2**—PostalCodePrimary identified by lexicon.
- **P1**—PostalCodePrimary identified by context.
- **P0**—PostalCodePrimary empty.

Match Score

A numeric value between 0 and 100 representing the similarity between the identified input data and the output data for the record. A result of 100 means that no changes other than additions, alias, casing, or diacritic changes have been made to the input data. A result of 0 means there is no similarity between the input data item and the output data provided.

AMAS Output

The following table lists the standard fields that are output by ValidateAddressLoqate.

Table 154: Output Fields

| Response Element | Description |
|------------------|-------------------------------------|
| Barcode | Standard barcode based on the DPID. |
| | F Failure (no barcode found) |
| | 20-digit number Success |

| Response Element | Description |
|------------------|---|
| DPID | The Delivery Point Identifier. An eight-digit number from the Australia Post Postal Address File that uniquely identifies a mail delivery point, such as a street address. Note: This field will contain "00000000" for Australian addresses that are not AMAS-verified and will be empty for non-Australian addresses. |
| FloorNumber | The floor/level number, for example: 123 E Main St Apt 3, 4th Floor |
| FloorType | The floor/level type, for example: 123 E Main St Apt 3, 4th Floor |
| PostalBoxNum | The postal delivery number, for example: PO Box 42 |

Spectrum Universal Name

OpenNameParser

OpenNameParser breaks down personal and business names and other terms in the name data field into their component parts. These parsed name elements are then subsequently available to other automated operations such as name matching, name standardization, or multiple-record name consolidation.

OpenNameParser does the following:

- Determines the type of a name in order to describe the function that the name performs. Name entity types are divided into two major groups: personal names and business names. Within each of these major groups are subgroups.
- Determines the form of a name in order to understand which syntax the parser should follow for parsing. Personal names usually take on a natural (signature) order or a reverse order. Business names are usually ordered hierarchically.
- Determines and labels the component parts of a name so that the syntactical relationship of each name part to the entire name is identified. The personal name syntax includes prefixes, first, middle, and last name parts, suffixes, and account description terms, among other personal name parts. The business name syntax includes the firm name and suffix terms.
- Parses conjoined personal and business names and either retains them as one record or splits them into multiple records. Examples of conjoined names include "Mr. and Mrs. John Smith" and "Baltimore Gas & Electric dba Constellation Energy".

- Parses output as records or as a list.
- Assigns a parsing score that reflects the degree of confidence that the parsing is correct.

Resource URL

```
http://server:port/soap/OpenNameParser
```

Example

The following shows a SOAP request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:open="http://www.precisely.com/spectrum/services/OpenNameParser"
xmlns:spec="http://spectrum.precisely.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <open:OpenNameParserRequest>
      <open:input_port>
        <open:Input>
          <open:Name>John Williams Smith</open:Name>
        </open:Input>
      </open:input_port>
    </open:OpenNameParserRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns3:OpenNameParserResponse xmlns:ns2="http://spectrum.precisely.com/"
xmlns:ns3="http://www.precisely.com/spectrum/services/OpenNameParser">
      <ns3:output_port>
        <ns3:Result>
          <ns3:Name>John Williams Smith</ns3:Name>
          <ns3:CultureCodeUsedToParse/>
          <ns3:FirstName>John</ns3:FirstName>
          <ns3:LastName>Smith</ns3:LastName>
          <ns3:MiddleName>Williams</ns3:MiddleName>
          <ns3:Names/>
          <ns3:IsParsed>true</ns3:IsParsed>
          <ns3:IsPersonal>true</ns3:IsPersonal>
          <ns3:IsConjoined>>false</ns3:IsConjoined>
          <ns3:IsReverseOrder>>false</ns3:IsReverseOrder>
          <ns3:IsFirm>>false</ns3:IsFirm>
          <ns3:NameScore>100</ns3:NameScore>
          <ns3:user_fields/>
        </ns3:Result>
      </ns3:OpenNameParserResponse>
    </soap:Body>
  </soap:Envelope>
```

```

</ns3:output_port>
</ns3:OpenNameParserResponse>
</soap:Body>
</soap:Envelope>

```

Request

Parameters for Input Data

Table 155: Open Name Parser Input

| Parameter | Description | | | | | | | | |
|---------------------|--|---------------------|---------------------------|-----------|---------|-----------|----------|-----------|-----------|
| CultureCode | <p>The culture of the input name data. The options are listed below.</p> <table border="0"> <tr> <td>Null (empty)</td> <td>Global culture (default).</td> </tr> <tr> <td>de</td> <td>German.</td> </tr> <tr> <td>es</td> <td>Spanish.</td> </tr> <tr> <td>ja</td> <td>Japanese.</td> </tr> </table> <p>Note: If you added your own domain using the Open Parser Domain Editor, the cultures and culture codes for that domain are also valid.</p> | Null (empty) | Global culture (default). | de | German. | es | Spanish. | ja | Japanese. |
| Null (empty) | Global culture (default). | | | | | | | | |
| de | German. | | | | | | | | |
| es | Spanish. | | | | | | | | |
| ja | Japanese. | | | | | | | | |
| Name | The name you want to parse. This field is required. | | | | | | | | |

Options

Parameters for Parsing Options

The following table lists the options that control the parsing of names.

Table 156: Open Name Parser Parsing Options

| Parameter | Description | | | | |
|--------------------------------|---|-------------|---|--------------|---|
| ParseNaturalOrderPersonalNames | <p>Specifies whether to parse names where the is in the order Title, First Name, Middle Name, Last Name, and Suffix.</p> <table border="0"> <tr> <td>true</td> <td>Parse personal names that are in natural order.</td> </tr> <tr> <td>false</td> <td>Do not parse names that are in natural order.</td> </tr> </table> | true | Parse personal names that are in natural order. | false | Do not parse names that are in natural order. |
| true | Parse personal names that are in natural order. | | | | |
| false | Do not parse names that are in natural order. | | | | |

| Parameter | Description |
|--------------------------------|--|
| ParseReverseOrderPersonalNames | <p>Specifies whether to parse names where the last name is specified first.</p> <p>true Parse personal names that are in reverse order.</p> <p>false Do not parse names that are in reverse order.</p> |
| ParseConjoinedNames | <p>Specifies whether to parse conjoined names.</p> <p>true Parse conjoined names.</p> <p>false Do not parse conjoined names.</p> |
| SplitConjoinedNames | <p>Specifies whether to separate names containing more than one individual into multiple records, for example, Bill & Sally Smith.</p> <p>true Split conjoined names.</p> <p>false Do not split conjoined names.</p> |
| ParseBusinessNames | <p>Specifies whether to parse business names.</p> <p>true Parse business names.</p> <p>false Do not parse business names.</p> |
| OutputAsList | <p>Specifies whether to return the parsed name elements in a list form.</p> <p>true Return the parsed elements in a list form.</p> <p>false Do not return the parsed elements in a list form.</p> |
| ShortcutThreshold | <p>Specifies how to balance performance versus quality. A faster performance will result in lower quality output; likewise, higher quality will result in slower performance. When this threshold is met, no other processing will be performed on the record.</p> <p>Specify a value from 0 to 100. The default is 100.</p> |

Parameters for Culture Options

The following table lists the options that control name cultures.

Table 157: Open Name Parser Cultures Options

| Parameter | Description |
|----------------|---|
| DefaultCulture | <p>Specifies which culture(s) you want to include in the parsing grammar. Global Culture is the default selection.</p> <p>Specify cultures by specifying the two-character culture code in a comma-separated list in priority order. For example, to attempt to parse the name using the Spanish culture first then Japanese, you would specify:</p> <pre>es,ja,,</pre> |

Parameters for Advanced Options

The following table lists the advanced options for name parsing.

Table 158: Open Name Parser Advanced Options

| Option | Description |
|-----------------------------------|---|
| NaturalOrderPersonalNamesDomain | <p>Specifies the domain to use when parsing natural order personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer.</p> |
| NaturalOrderPersonalNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the natural order personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |

| Option | Description |
|--|--|
| ReverseOrderPersonalNamesDomain | Specifies the domain to use when parsing reverse order personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |
| ReverseOrderPersonalNamesPriority | Specify a number between 1 and 5 that indicates the priority of the reverse order personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run. Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned. |
| NaturalOrderConjoinedPersonalNamesDomain | Specifies the domain to use when parsing natural order conjoined personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |
| NaturalOrderConjoinedPersonalNamesPriority | Specify a number between 1 and 5 that indicates the priority of the natural order conjoined personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run. Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned. |
| ReverseOrderConjoinedPersonalNamesDomain | Specifies the domain to use when parsing reverse order conjoined personal names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer. |

| Option | Description |
|--|---|
| ReverseOrderConjoinedPersonalNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the reverse order conjoined personal names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |
| BusinessNamesDomain | <p>Specifies the domain to use when parsing business names. The valid values are the domain names defined in the Open Parser Domain Editor too in Enterprise Designer.</p> |
| BusinessNamesPriority | <p>Specify a number between 1 and 5 that indicates the priority of the business names domain relative to the other domains that you are using. This determines the order in which you want the parsers to run.</p> <p>Results will be returned for the first domain that scores higher than the number set in the shortcut threshold option. If no domain reaches that threshold, results for the domain with the highest score are returned. If multiple domains reach the threshold at the same time, priority goes to the domain that was run first (determined by the order set here) and its results will be returned.</p> |

Response

Table 159: Open Name Parser Output

| Response Element | Format | Description |
|--------------------|--------|--|
| AccountDescription | String | An account description that is part of the name. For example, in "Mary Jones Account # 12345", the account description is "Account#12345". |

| Response Element | Format | Description |
|------------------|--------|--|
| Names | String | A hierarchical field that contains a list of parsed elements. This field is returned when you check the Output results as list box under Parsing Options. |

Fields Related to Names of Companies

| | | |
|-----------------|--------|---|
| FirmConjunction | String | Indicates that the name of a firm contains a conjunction such as "d/b/a" (doing business as), "o/a" (operating as), and "t/a" (trading as). |
| FirmName | String | The name of a company. For example, <code>Precisely</code> . |
| FirmSuffix | String | The corporate suffix. For example, "Co." and "Inc." |
| IsFirm | String | Indicates that the name is a firm rather than an individual. |

Fields Related to Names of Individual People

| | | | | | | | | | | |
|------------------------|---------------------------|--|---------------------|---------------------------|-----------|---------|-----------|----------|-----------|-----------|
| Conjunction | String | Indicates that the name contains a conjunction such as "and", "or", or "&". | | | | | | | | |
| CultureCode | String | The culture codes contained in the input data. | | | | | | | | |
| CultureCodeUsedToParse | String | Identifies the culture-specific grammar that was used to parse the data. <table border="0" style="margin-left: 20px;"> <tr> <td>Null (empty)</td> <td>Global culture (default).</td> </tr> <tr> <td>de</td> <td>German.</td> </tr> <tr> <td>es</td> <td>Spanish.</td> </tr> <tr> <td>ja</td> <td>Japanese.</td> </tr> </table> <p>Note: If you added your own domain using the Open Parser Domain Editor, the cultures and culture codes for that domain will appear in this field as well.</p> | Null (empty) | Global culture (default). | de | German. | es | Spanish. | ja | Japanese. |
| Null (empty) | Global culture (default). | | | | | | | | | |
| de | German. | | | | | | | | | |
| es | Spanish. | | | | | | | | | |
| ja | Japanese. | | | | | | | | | |

| Response Element | Format | Description |
|-------------------|--------|---|
| FirstName | String | The first name of a person. |
| GeneralSuffix | String | A person's general/professional suffix. For example, MD or PhD. |
| IsParsed | String | Indicates whether an output record was parsed. Values are true or false. |
| IsPersonal | String | Indicates whether the name is an individual rather than a firm. Values are true or false. |
| IsReverseOrder | String | Indicates whether the input name is in reverse order. Values are true or false. |
| LastName | String | The last name of a person. Includes the paternal last name. |
| LeadingData | String | Non-name information that appears before a name. |
| MaturitySuffix | String | A person's maturity/generational suffix. For example, Jr. or Sr. |
| MiddleName | String | The middle name of a person. |
| Name. | String | The personal or firm name that was provided in the input. |
| NameScore | String | Indicates the average score of known and unknown tokens for each name. The value of NameScore will be between 0 and 100, as defined in the parsing grammar. 0 is returned when no matches are returned. |
| SecondaryLastName | String | In Spanish parsing grammar, the surname of a person's mother. |
| TitleOfRespect | String | Information that appears before a name, such as "Mr.", "Mrs.", or "Dr." |

| Response Element | Format | Description |
|--|--------|--|
| TrailingData | String | Non-name information that appears after a name. |
| Fields Related to Conjoined Names | | |
| Conjunction2 | String | Indicates that a second, conjoined name contains a conjunction such as "and", "or", or "&". |
| Conjunction3 | String | Indicates that a third, conjoined name contains a conjunction such as "and", "or", or "&". |
| FirmName2 | String | The name of a second, conjoined company. For example, Baltimore Gas & Electric dba Constellation Energy. |
| FirmSuffix2 | String | The suffix of a second, conjoined company. |
| FirstName2 | String | The first name of a second, conjoined name. |
| FirstName3 | String | The first name of a third, conjoined name. |
| GeneralSuffix2 | String | The general/professional suffix for a second, conjoined name. For example, MD or PhD. |
| GeneralSuffix3 | String | The general/professional suffix for a third, conjoined name. For example, MD or PhD. |
| IsConjoined | String | Indicates that the input name is conjoined. An example of a conjoined name is "John and Jane Smith." |
| LastName2 | String | The last name of a second, conjoined name. |

| Response Element | Format | Description |
|------------------|--------|---|
| LastName3 | String | The last name of a third, conjoined name. |
| MaturitySuffix2 | String | The maturity/generational suffix for a second, conjoined name. For example, Jr. or Sr. |
| MaturitySuffix3 | String | The maturity/generational suffix for a third, conjoined name. For example, Jr. or Sr. |
| MiddleName2 | String | The middle name of a second, conjoined name. |
| MiddleName3 | String | The middle name of a third, conjoined name. |
| TitleOfRespect2 | String | Information that appears before a second, conjoined name, such as "Mr.", "Mrs.", or "Dr." |
| TitleOfRespect3 | String | Information that appears before a third, conjoined name, such as "Mr.", "Mrs.", or "Dr." |

Appendix

In this section

| | |
|--|------|
| Buffering..... | 1007 |
| Country Codes..... | 1010 |
| Validate Address Confidence Algorithm..... | 1035 |

A - Buffering

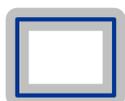
In this section

Buffering.....1008



Buffering

Use buffering to define areas that are close to the edges of a polygon, line, or point.

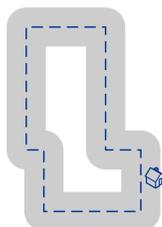


Buffered Polygon (zone)

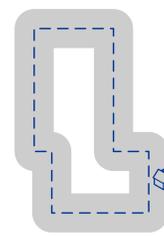
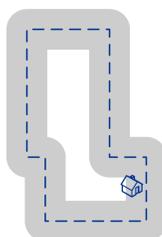
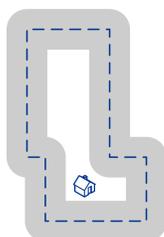
Buffered Line (corridor)

Buffered Point (circle)

For example, if you work for an insurance company you may want to know if a potential customer's house is within 500 feet of a flood plain so that you can suggest that they buy flood insurance even though they are not actually within the flood plain. The following illustration shows this scenario using a buffered polygon. The dotted line indicates the boundary of the flood plain and the shaded area shows a 500-foot buffer zone around the boundary.



The buffer area extends on both sides of the boundary (inside and outside). When you use buffering, the output field BufferRelation indicates whether or not the point is in the buffered zone, and whether the point is inside or outside of the polygon, as shown in the following illustrations.



The point is inside the polygon and not in the buffer area.

The output field BufferRelation will contain "P".

The point is inside the polygon and in the buffer area.

The output field BufferRelation will contain "I".

The point is outside the polygon but in the buffer area.

The output field BufferRelation will contain "B".

Specify the size of polygon buffers using the BufferWidth input field to set it on a record-by-record basis and the DefaultBufferWidth Default Buffer Width option to set a default polygon buffer width for the job.

B - Country Codes

In this section

| | |
|---|------|
| ISO Country Codes and Module Support..... | 1011 |
|---|------|



ISO Country Codes and Module Support

This table lists the ISO codes for each country as well as the modules that support addressing, geocoding, and routing for each country.

Note that the Spectrum Enterprise Geocoding includes databases for Africa (30 countries), Middle East (8 countries) and Latin America (20 countries). These databases cover the smaller countries in those regions that do not have their own country-specific geocoding databases. The Supported Modules column indicates which countries are covered by these Africa, Middle East, and Latin America databases.

Also, the Geocode Address World database provides geographic and limited postal geocoding (but not street-level geocoding) for all countries.

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------|--------------------|--------------------|---|
| Afghanistan | AF | AFG | Spectrum Universal Address |
| Aland Islands | AX | ALA | Spectrum Universal Address |
| Albania | AL or SQ (Routing) | ALB | Spectrum Universal Address Spectrum Enterprise Geocoding Spectrum Spatial Routing |
| Algeria | DZ | DZA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| American Samoa | AS | ASM | Spectrum Universal Address |
| Andorra | AD | AND | Spectrum Enterprise Geocoding. (Andorra is covered by the Spain geocoder) Spectrum Universal Address |
| Angola | AO | AGO | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------|--------------------|--------------------|---|
| Anguilla | AI | AIA | Spectrum Universal Address |
| Antarctica | AQ | ATA | Spectrum Universal Address |
| Antigua And Barbuda | AG | ATG | Spectrum Universal Address |
| Argentina | AR | ARG | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| Armenia | AM | ARM | Spectrum Universal Address |
| Aruba | AW | ABW | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Australia | AU | AUS | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Austria | AT | AUT | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Azerbaijan | AZ | AZE | Spectrum Universal Address |
| Bahamas | BS | BHS | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------|--------------------|--------------------|---|
| Bahrain | BH | BHR | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Bangladesh | BD | BGD | Spectrum Universal Address |
| Barbados | BB | BRB | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Belarus | BY | BLR | Spectrum Universal Address Spectrum Spatial Routing |
| Belgium | BE | BEL | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Belize | BZ | BLZ | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Benin | BJ | BEN | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Bermuda | BM | BMU | Spectrum Universal Address Spectrum Spatial Routing |
| Bhutan | BT | BTN | Spectrum Universal Address |
| Bolivia | BO | BOL | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|-----------------------------------|--------------------|--------------------|--|
| Bonaire, Saint Eustatius And Saba | BQ | BES | Spectrum Universal Address |
| Bosnia And Herzegovina | BA | BIH | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing Spectrum Enterprise Geocoding |
| Botswana | BW | BWA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Bouvet Island | BV | BVT | Spectrum Universal Address |
| Brazil | BR | BRA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| British Indian Ocean Territory | IO | IOT | Spectrum Universal Address |
| Brunei Darussalam | BN | BRN | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Bulgaria | BG | BGR | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Burkina Faso | BF | BFA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Burundi | BI | BDI | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--------------------------|-----------------------|--------------------|---|
| Cambodia | KH | KHM | Spectrum Universal Address |
| Cameroon | CM | CMR | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Canada | CA | CAN | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Cape Verde | CV | CPV | Spectrum Universal Address |
| Cayman Islands | KY | CYM | Spectrum Universal Address |
| Central African Republic | CF | CAF | Spectrum Universal Address |
| Chad | TD | TCD | Spectrum Universal Address |
| Chile | CL | CHL | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| China | CN or zh_CN (Routing) | CHN | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| Christmas Island | CX | CXR | Spectrum Universal Address |
| Cocos (Keeling) Islands | CC | CCK | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------------------------|--------------------|--------------------|---|
| Colombia | CO | COL | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Comoros | KM | COM | Spectrum Universal Address |
| Congo, Republic Of The | CG | COG | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Congo, The Democratic Republic Of The | CD | COD | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Cook Islands | CK | COK | Spectrum Universal Address |
| Costa Rica | CR | CRI | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Côte d'Ivoire | CI | CIV | Spectrum Universal Address |
| Croatia | HR | HRV | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Cuba | CU | CUB | Spectrum Enterprise Geocoding (Latin America) Spectrum Spatial Routing Spectrum Universal Address |
| Curacao | CW | CUW | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--------------------|-----------------------|--------------------|---|
| Cyprus | CY | CYP | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Czech Republic | CZ or CS (Routing) | CZE | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| Denmark | DK | DNK | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Djibouti | DJ | DJI | Spectrum Universal Address |
| Dominica | DM | DMA | Spectrum Universal Address |
| Dominican Republic | DO | DOM | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Ecuador | EC | ECU | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Egypt | EG | EGY | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| El Salvador | SV | SLV | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Equatorial Guinea | GQ | GNQ | Spectrum Universal Address |
| Eritrea | ER | ERI | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|-----------------------------|--------------------|--------------------|---|
| Estonia | EE | EST | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Addressing |
| Ethiopia | ET | ETH | Spectrum Universal Address |
| Falkland Islands (Malvinas) | FK | FLK | Spectrum Universal Address |
| Faroe Islands | FO | FRO | Spectrum Universal Address |
| Fiji | FJ | FJI | Spectrum Universal Address |
| Finland | FI | FIN | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| France | FR | FRA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| French Guiana | GF | GUF | Spectrum Enterprise Geocoding (<i>French Guiana is covered by the France geocoder.</i>) Spectrum Universal Address |
| French Polynesia | PF | PYF | Spectrum Universal Address |
| French Southern Territories | TF | ATF | Spectrum Universal Address |
| Gabon | GA | GAB | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------|--------------------|--------------------|--|
| Gambia | GM | GMB | Spectrum Universal Address |
| Georgia | GE | GEO | Spectrum Universal Address |
| Germany | DE | DEU | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Ghana | GH | GHA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Gibraltar | GI | GIB | Spectrum Enterprise Geocoding (<i>Gibraltar is covered by the Spain geocoder.</i>) Spectrum Universal Address |
| Greece | GR | GRC | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Greenland | GL | GRL | Spectrum Universal Address |
| Grenada | GD | GRD | Spectrum Universal Address |
| Guadeloupe | GP | GLP | Spectrum Enterprise Geocoding (<i>Guadeloupe is covered by the France geocoder.</i>) Spectrum Universal Address |
| Guam | GU | GUM | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|-----------------------------------|--------------------|--------------------|--|
| Guatemala | GT | GTM | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Guernsey | GG | GGY | Spectrum Universal Address |
| Guinea | GN | GIN | Spectrum Universal Address |
| Guinea-Bissau | GW | GNB | Spectrum Universal Address |
| Guyana | GY | GUY | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Haiti | HT | HTI | Spectrum Universal Address |
| Heard Island and McDonald Islands | HM | HMD | Spectrum Universal Address |
| Holy See (Vatican City State) | VA | VAT | Spectrum Enterprise Geocoding (<i>The Vatican is covered by the Italy geocoder.</i>) Spectrum Universal Address |
| Honduras | HN | HND | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Hong Kong | HK | HKG | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Hungary | HU | HUN | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------------|--------------------|--------------------|---|
| Iceland | IS | ISL | Spectrum Enterprise Geocoding Spectrum Universal Address |
| India | IN | IND | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Indonesia | ID | IDN | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Iran, Islamic Republic Of | IR | IRN | Spectrum Universal Address |
| Iraq | IQ | IRQ | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Ireland | IE | IRL | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Isle Of Man | IM | IMN | Spectrum Universal Address |
| Israel | IL | ISR | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| Italy | IT | ITA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Jamaica | JM | JAM | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--|--------------------|--------------------|---|
| Japan | JP | JPN | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Jersey | JE | JEY | Spectrum Universal Address |
| Jordan | JO | JOR | Spectrum Universal Address Spectrum Enterprise Geocoding (Middle East) Spectrum Spatial Routing |
| Kazakhstan | KZ | KAZ | Spectrum Universal Address |
| Kenya | KE | KEN | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Kiribati | KI | KIR | Spectrum Universal Address |
| Korea, Democratic People's Republic Of | KP | PRK | Spectrum Universal Address |
| Korea, Republic Of | KR | KOR | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Kosovo | Xk | XXK | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Kuwait | KW | KWT | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|----------------------------------|--------------------|--------------------|--|
| Kyrgyzstan | KG | KGZ | Spectrum Universal Address |
| Lao People's Democratic Republic | LA | LAO | Spectrum Universal Address |
| Latvia | LV | LVA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Lebanon | LB | LBN | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Lesotho | LS | LSO | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Liberia | LR | LBR | Spectrum Universal Address |
| Libyan Arab Jamahiriya | LY | LBY | Spectrum Universal Address |
| Liechtenstein | LI | LIE | Spectrum Enterprise Geocoding (<i>Liechtenstein is covered by the Switzerland geocoder.</i>) Spectrum Spatial Routing Spectrum Universal Address |
| Lithuania | LT | LTU | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Luxembourg | LU | LUX | Spectrum Enterprise Geocoding (<i>Luxembourg is covered by the Belgium geocoder.</i>) Spectrum Spatial Routing Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--|--------------------|--------------------|--|
| Macao | MO | MAC | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Macedonia, Former Yugoslav Republic Of | MK | MKD | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Madagascar | MG | MDG | Spectrum Universal Address |
| Malawi | MW | MWI | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Malaysia | MY | MYS | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Maldives | MV | MDV | Spectrum Universal Address |
| Mali | ML | MLI | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Malta | ML | MLT | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Marshall Islands | MH | MHL | Spectrum Universal Address |
| Martinique | MQ | MTQ | Spectrum Enterprise Geocoding (<i>Martinique is covered by the France geocoder.</i>) Spectrum Universal Address |
| Mauritania | MR | MRT | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------------------|--------------------|--------------------|---|
| Mauritius | MU | MUS | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Mayotte | YT | MYT | Spectrum Enterprise Geocoding (<i>Mayotte is covered by the France geocoder.</i>) Spectrum Universal Address |
| Mexico | MX | MEX | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Micronesia, Federated States Of | FM | FSM | Spectrum Universal Address |
| Moldova, Republic Of | MD | MDA | Spectrum Universal Address Spectrum Spatial Routing |
| Monaco | MC | MCO | Spectrum Enterprise Geocoding (<i>Monaco is covered by the France geocoder.</i>) Spectrum Universal Address |
| Mongolia | MN | MNG | Spectrum Universal Address |
| Montenegro | ME | MNE | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Montserrat | MS | MSR | Spectrum Universal Address |
| Morocco | MA | MAR | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------|--------------------|--------------------|--|
| Mozambique | MZ | MOZ | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Myanmar | MM | MMR | Spectrum Universal Address |
| Namibia | NA | NAM | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Nauru | NR | NRU | Spectrum Universal Address |
| Nepal | NP | NPL | Spectrum Universal Address |
| Netherlands | NL | NLD | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| New Caledonia | NC | NCL | Spectrum Universal Address |
| New Zealand | NZ | NZL | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Nicaragua | NI | NIC | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Niger | NE | NER | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------------------|--------------------|--------------------|---|
| Nigeria | NG | NGA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Niue | NU | NIU | Spectrum Universal Address |
| Norfolk Island | NF | NFK | Spectrum Universal Address |
| Northern Mariana Islands | MP | MNP | Spectrum Universal Address |
| Norway | NO | NOR | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Oman | OM | OMN | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Pakistan | PK | PAK | Spectrum Universal Address |
| Palau | PW | PLW | Spectrum Universal Address |
| Palestinian Territory, Occupied | PS | PSE | Spectrum Universal Address |
| Panama | PA | PAN | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Papua New Guinea | PG | PNG | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------|--------------------|--------------------|---|
| Paraguay | PY | PRY | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Peru | PE | PER | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Philippines | PH | PHL | Spectrum Enterprise Geocoding Spectrum Universal Address Spectrum Spatial Routing |
| Pitcairn | PN | PCN | Spectrum Universal Address |
| Poland | PL | POL | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Portugal | PT | PRT | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Puerto Rico | PR | PRI | Spectrum Universal Address |
| Qatar | QA | QAT | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Reunion | RE | REU | Spectrum Enterprise Geocoding (<i>Reunion is covered by the France geocoder.</i>) Spectrum Universal Address |
| Romania | RO | ROU | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--|--------------------|--------------------|---|
| Russian Federation | RU | RUS | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Rwanda | RW | RWA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Saint Barthelemy | BL | BLM | Spectrum Universal Address |
| Saint Helena, Ascension and Tristan Da Cunha | SH | SHE | Spectrum Universal Address |
| Saint Kitts and Nevis | KN | KNA | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Saint Lucia | LC | LCA | Spectrum Universal Address |
| Saint Martin (French Part) | MF | MAF | Spectrum Universal Address |
| Saint Pierre and Miquelon | PM | SPM | Spectrum Universal Address |
| Saint Vincent and the Grenadines | VC | VCT | Spectrum Universal Address |
| Samoa | WS | WSM | Spectrum Universal Address |
| San Marino | SM | SMR | Spectrum Enterprise Geocoding (<i>San Marino is covered by the Italy geocoder.</i>) Spectrum Universal Address |
| Sao Tome and Principe | ST | STP | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|---------------------------|--------------------|--------------------|---|
| Saudi Arabia | SA | SAU | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Senegal | SN | SEN | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Serbia | RS | SRB | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Seychelles | SC | SYC | Spectrum Universal Address |
| Sierra Leone | SL | SLE | Spectrum Universal Address |
| Singapore | SG | SGP | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Sint Maarten (Dutch Part) | SX | SXM | Spectrum Universal Address |
| Slovakia | SK | SVK | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Slovenia | SI | SVN | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Solomon Islands | SB | SLB | Spectrum Universal Address |
| Somalia | SO | SOM | Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--|--------------------|--------------------|---|
| South Africa | ZA | ZAF | Spectrum Enterprise Geocoding Spectrum Universal Address |
| South Georgia And The South Sandwich Islands | GS | SGS | Spectrum Enterprise Geocoding Spectrum Universal Address |
| South Sudan | SS | SSD | Spectrum Universal Address |
| Spain | ES | ESP | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Sri Lanka | LK | LKA | Spectrum Universal Address |
| Sudan | SD | SDN | Spectrum Universal Address |
| Suriname | SR | SUR | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Svalbard And Jan Mayen | SJ | SJM | Spectrum Universal Address |
| Swaziland | SZ | SWZ | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Sweden | SE | SWE | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Switzerland | CH | CHE | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|------------------------------|-----------------------|--------------------|--|
| Syrian Arab Republic | SY | SYR | Spectrum Universal Address |
| Taiwan, Province of China | TW or zh_TW (Routing) | TWN | Spectrum Universal Address Spectrum Spatial Routing |
| Tajikistan | TJ | TJK | Spectrum Universal Address |
| Tanzania, United Republic Of | TZ | TZA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address Spectrum Spatial Routing |
| Thailand | TH | THA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| Timor-Leste | TL | TLS | Spectrum Universal Address |
| Togo | TG | TGO | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Tokelau | TK | TKL | Spectrum Universal Address |
| Tonga | TO | TON | Spectrum Universal Address |
| Trinidad and Tobago | TT | TTO | Spectrum Enterprise Geocoding (Latin America) Spectrum Universal Address |
| Tunisia | TN | TUN | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|--------------------------------------|--------------------|--------------------|---|
| Turkey | TR | TUR | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Turkmenistan | TM | TKM | Spectrum Universal Address |
| Turks And Caicos Islands | TC | TCA | Spectrum Universal Address |
| Tuvalu | TV | TUV | Spectrum Universal Address |
| Uganda | UG | UGA | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Ukraine | UA | UKR | Spectrum Enterprise Geocoding Spectrum Universal Address |
| United Arab Emirates | AE | ARE | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| United Kingdom | GB | GBR | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| United States | US | USA | Spectrum Enterprise Geocoding Spectrum Spatial Routing Spectrum Universal Address |
| United States Minor Outlying Islands | UM | UMI | Spectrum Universal Address |
| Uruguay | UY | URY | Spectrum Enterprise Geocoding Spectrum Universal Address |

| ISO Country Name | ISO 3166-1 Alpha-2 | ISO 3166-1 Alpha-3 | Supported Modules |
|-----------------------------------|--------------------|--------------------|---|
| Uzbekistan | UZ | UZB | Spectrum Universal Address |
| Vanuatu | VU | VUT | Spectrum Universal Address |
| Venezuela, Bolivarian Republic Of | VE | VEN | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Viet Nam | VN | VNM | Spectrum Enterprise Geocoding Spectrum Universal Address |
| Virgin Islands, British | VG | VGB | Spectrum Universal Address |
| Virgin Islands, U.S. | VI | VIR | Spectrum Universal Address |
| Wallis and Futuna | WF | WLF | Spectrum Universal Address |
| Western Sahara | EH | ESH | Spectrum Universal Address |
| Yemen | YE | YEM | Spectrum Enterprise Geocoding (Middle East) Spectrum Universal Address |
| Zambia | ZM | ZMB | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |
| Zimbabwe | ZW | ZWE | Spectrum Enterprise Geocoding (Africa) Spectrum Universal Address |

C - Validate Address Confidence Algorithm

In this section

| | |
|--|------|
| Introduction to the Validate Address Confidence Algorithm..... | 1036 |
| Confidence Algorithm for U.S. and Canadian Addresses..... | 1036 |
| Confidence Algorithm for International Addresses..... | 1037 |



Introduction to the Validate Address Confidence Algorithm

ValidateAddress computes a confidence score for each validated address. This score describes how likely it is that the validated address is correct. Confidence code values range from 0 to 100, with a zero confidence level indicating no confidence and 100 indicating a very high level of confidence that the match results are correct. Confidence codes are calculated based on an algorithm that takes into account the match results for individual output fields. The output fields involved in this calculation include:

- Country
- City
- StateProvince
- PostalCode
- StreetName
- HouseNumber
- LeadingDirectional
- TrailingDirectional
- StreetSuffix
- ApartmentNumber

Each field has its own Weight in the algorithm. Additionally, for each field the match result could be labeled as Success, Failure, or Changed. ("Changed" refers to cases where the contents of the field have been corrected in order to get a match.) The match result—Success, Failure, or Changed—determines what the Factor is for that field. Thus, the calculation for the confidence code is a product of Weight by Factor as follows:

```
Confidence = (Weight * Factor) for City
+ (Weight * Factor) for Country
+ (Weight * Factor) for State
+ (Weight * Factor) for PostalCode
+ (Weight * Factor) for StreetName
+ (Weight * Factor) for HouseNumber
+ (Weight * Factor) for Directionals
+ (Weight * Factor) for Street Suffix
+ (Weight * Factor) for ApartmentNumber
```

Confidence Algorithm for U.S. and Canadian Addresses

The following table details the scoring and logic behind the ValidateAddress confidence algorithm for U.S. and Canadian addresses.

Table 160: Confidence Algorithm for U.S. and Canadian Addresses

| Field | Weight/Match Score | Factor if Changed ¹ | Factor If Filled ² |
|-----------------|--------------------|--------------------------------|-------------------------------|
| Country | 10 | 100% | 0% |
| City | 10 | 50% | 75% |
| StateProvince | 15 | 50% | 75% |
| PostalCode | 15 | 25% | 25% |
| StreetName | 15 | 50% | 75% |
| HouseNumber | 15 | 50% | 75% |
| Directionals | 10 | 50% | 75% |
| StreetSuffix | 5 | 50% | 75% |
| ApartmentNumber | 5 | 50% | 75% |

Confidence Algorithm for International Addresses

There are two confidence algorithms for addresses outside the U.S. and Canada, one for addresses in countries that use postal codes and one for addresses in countries that do not use postal codes.

The following table details the confidence algorithm for non-U.S. and non-Canadian addresses from countries that use postal codes.

² Refers to instances when the input data in this field is not present but is filled in order to achieve a match.

¹ Refers to instances when the input data in this field is changed in order to achieve a match.

Table 161: Confidence Algorithm for Countries With Postal Codes

| Field | Weight/Match Score | Factor if Changed ³ | Factor If Filled ⁴ | Factor if Postal Data Unavailable |
|-----------------|--------------------|--------------------------------|-------------------------------|-----------------------------------|
| Country | 11.11111111111111 | 100% | 0% | 0% |
| City | 11.11111111111111 | 50% | 75% ⁵ | 0% |
| StateProvince | 16.66666666666667 | 100% | 100 | 80% |
| PostalCode | 16.66666666666667 | 100% | 100% | 80% |
| StreetName | 16.66666666666667 | 50% | 75% | 50% |
| HouseNumber | 16.66666666666667 | 50% | 75% | 50% |
| Directionals | 0 | 50% | 75% | 0% |
| StreetSuffix | 5.55555555555556 | 50% | 75% | 50% |
| ApartmentNumber | 5.55555555555556 | 50% | 75% | 50% |

⁴ Refers to instances when the input data in this field is not present but is filled in order to achieve a match.

³ Refers to instances when the input data in this field is changed in order to achieve a match.

⁵ If the country is a Category C country, this value is 50%. Countries fall into one of these categories:

- **Category A**—Enables the validation and correction of an address's postal code, city name, state/county name, street address elements, and country name.
- **Category B**—Enables the validation and correction of an address's postal code, city name, state/county name, and country name. It does not support the validation or correction of street address elements.
- **Category C**—Enables the validation and correction of the country name, and the validation of the format of the postal code.

The following table details confidence algorithm for countries that do not use postal codes.

Table 162: Confidence Algorithm for Countries Without Postal Codes

| Field | Weight/Match Score | Factor if Changed ⁶ | Factor If Filled ⁷ | Factor if Postal Data Unavailable |
|-----------------|--------------------|--------------------------------|-------------------------------|-----------------------------------|
| Country | 13.33333333333333 | 100% | 0% | 0% |
| City | 13.33333333333333 | 50% | 75% ⁸ | 0% |
| StateProvince | 20 | 100% | 100 | 80% |
| StreetName | 20 | 50% | 75% | 50% |
| HouseNumber | 20 | 50% | 75% | 50% |
| Directionals | 0 | 50% | 75% | 0% |
| StreetSuffix | 6.66666666666667 | 50% | 75% | 50% |
| ApartmentNumber | 6.66666666666667 | 50% | 75% | 50% |

⁷ Refers to instances when the input data in this field is not present but is filled in order to achieve a match.

⁶ Refers to instances when the input data in this field is changed in order to achieve a match.

⁸ If the country is a Category C country, this value is 50%. Countries fall into one of these categories:

- **Category A**—Enables the validation and correction of an address's postal code, city name, state/county name, street address elements, and country name.
- **Category B**—Enables the validation and correction of an address's postal code, city name, state/county name, and country name. It does not support the validation or correction of street address elements.
- **Category C**—Enables the validation and correction of the country name, and the validation of the format of the postal code.

The following table lists countries without postal codes.

Table 163: Countries Without Postal Codes

| | | |
|---------------|-------------------|----------------------|
| Afghanistan | Albania | Angola |
| Anguilla | Bahamas | Barbados |
| Belize | Benin | Bhutan |
| Botswana | Burkina Faso | Burundi |
| Cameroon | Cayman Islands | Central African Rep. |
| Chad | Cocos Islands | Colombia |
| Comoros | Congo (Dem. Rep.) | Congo (Rep.) |
| Cote d'Ivoire | Korea (North) | Djibouti |
| Dominica | Equatorial Guinea | Eritrea |
| Fiji | Gabon | Gambia |
| Ghana | Grenada | Guyana |
| Ireland | Jamaica | Kiribati |
| Libya | Malawi | Mali |
| Mauritania | Namibia | Nauru |

| | | |
|-------|--------|------|
| Palau | Panama | Peru |
|-------|--------|------|

| | | |
|-------|--------|-------------|
| Qatar | Rwanda | Saint Lucia |
|-------|--------|-------------|

| | | |
|----------------------------------|-------|-----------------------|
| Saint Vincent and the Grenadines | Samoa | Sao Tome and Principe |
|----------------------------------|-------|-----------------------|

| | | |
|------------|--------------|----------|
| Seychelles | Sierra Leone | Suriname |
|------------|--------------|----------|

| | | |
|----------|-------|------|
| Tanzania | Timor | Togo |
|----------|-------|------|

| | | |
|-------|-------------------|--------|
| Tonga | Trinidad & Tobago | Tuvalu |
|-------|-------------------|--------|

| | | |
|--------|----------------------|---------|
| Uganda | United Arab Emirates | Vanuatu |
|--------|----------------------|---------|

| | | |
|-------|----------|--|
| Yemen | Zimbabwe | |
|-------|----------|--|



2 Blue Hill Plaza, #1563
Pearl River, NY 10965
USA

www.precisely.com

© 2007, 2021 Precisely. All rights reserved.