pitney bowes

Spectrum Technology Platform

Version 12.0 SP1

Web Services Guide

Table of Contents

1 - Getting Started

REST 4
SOAP 26

2 - Web Services

REST 45
SOAP 447

Chapter : Appendix

Appendix A:
Buffering 851
Appendix B:
Country Codes 854
Appendix C:
Validate Address Confidence Algorithm 886

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 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 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 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.pb.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 wold result in this response since a JSON response was requested:

```
{
 "ns1.json.ValidateAddressResponse" :
```

```
 {
 "ns1.output_port" :
 {
 "ns1.Confidence" : 82,
 "ns1.RecordType" : "Normal",
 "ns1.CountryLevel" : "A",
 "ns1.ProcessedBy" : "USA",
 "ns1.MatchScore" : 0,
 "ns1.AddressLine1" : "1825 KRAMER LN",
 "ns1.City" : "AUSTIN",
 "ns1.StateProvince" : "TX",
 "ns1.PostalCode" : "78758-4260",
 "ns1.PostalCode.Base" : 78758,
 "ns1.PostalCode.AddOn" : 4260,
 "ns1.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 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 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 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`.

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 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 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 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 Enterprise Designer and select **Edit** > **Dataflow Options**.

OptionValue1

A legal value for the option. For a list of legal values, open the service in 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"
         }
        ],
        "user_fields": [
        {
          "name": "Note1",
          "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 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 name `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.pb.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.pb.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 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.pb.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.pb.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.pb.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.pb.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 Enterprise Designer and select **Edit** > **Dataflow Options**.

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

```
<ServiceNameRequest
xmlns:svc="http://www.pb.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 Enterprise Designer and select **Edit** > **Dataflow Options**.

OptionValue

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

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

```
<AddressValidationRequest
xmlns:svc="http://www.pb.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.pb.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>
```

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 Enterprise Designer.
- `Row` 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.
- `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 requesters to authenticate with valid user credentials. There are two methods for authenticating: Basic authentication and token-based authentication.

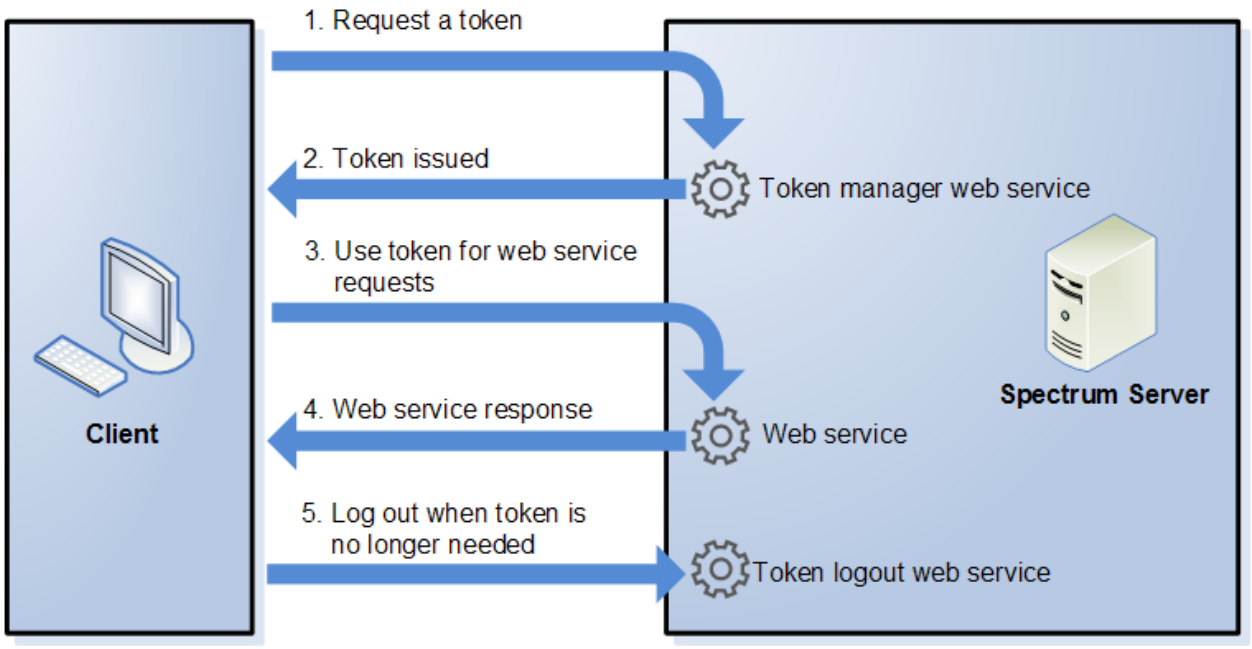
Basic Authentication

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-based authentication to access web services.

Token-Based Authentication

With token-based authentication, the requester obtains a token from the Spectrum™ Technology Platform server, then uses the token 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 following diagram 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":
"eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiZGlyIn0..ESnq4JNEBbVMKycd139z0w.NFXAskVY0seX",

   "session": "09aa1fbb-71j3-43c7-ab8c-d800214283d4",
   "username": "admin"
}
```

The response contains the following 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.
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": "eyJlbmMiOiJBMTI4Q0JDLUhTMjU2IiwiYWxnIjoiZGlyIn0..fI",

   "username": "admin"
}
```

The response contains the following 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 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
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
```

For example,

```
HTTP/1.1

POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbmMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..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 `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 18.This service does not take any parameters.

Exposing a Service as a Web Service

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

1. Open 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 the following 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 dataflow 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 `${Option.`*`Name`*`}` for dataflow options and `${Data.`*`Name`*`}` for dataflow 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 the following the 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 dataflow is exposed the light bulb button in the Enterprise Designer tool bar indicates that the dataflow 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`

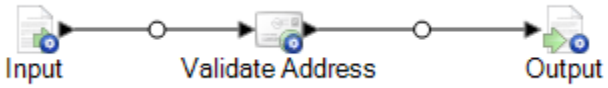
Where *server* is the name or IP address of your Spectrum™ Technology Platform server and *port* is the port used for HTTP communication.

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 Universal Addressing Module. These web services support GET only. You can add POST support to these standard web services by creating a user-defined service in 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 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 20.

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 as well for the Client SDK.

Micro-Batch Size

There is no limit to the number of records you can include in a request, but in general you will see the best performance when sending between 50 and 100 records in a micro-batch. We recommend that you test micro-batches of various sizes to determine the optimal micro-batch size for your environment. Keep in mind that in some cases you may get multiple records in the response for each input record. For example, if you are performing address validation and include 10 addresses in the micro-batch, and each address matches to two possible validated addresses, you would get 20 records in the response, not just 10.

Use caution when using both micro-batches and multiple threads for requests to Spectrum™ Technology Platform. Multiple threads can overwhelm the system if each thread's micro-batch size is too large.

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
http://spectrum.example.com:8080/rest/ValidateAddressPOST/results.xml
HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/xml
Authorization: Basic YWRtaW46YWRtaW4=
Content-Length: 533
Host: config813vm0:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

<ValidateAddressPOSTRequest
xmlns:svc="http://www.pb.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.pb.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.pb.com/"
 xmlns:ns3="http://www.pb.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.pb.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.pb.com/"
 xmlns:ns3="http://www.pb.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 requesters to authenticate with valid user credentials. There are two methods for authenticating: Basic authentication and token-based authentication.

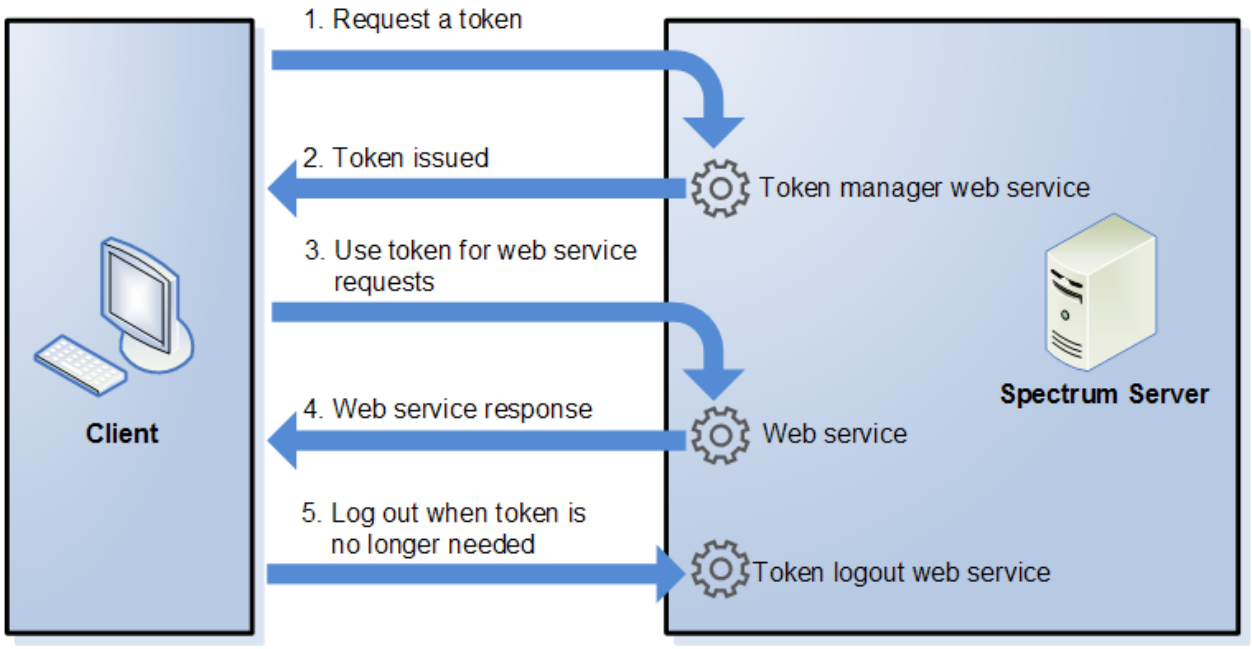
Basic Authentication

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-based authentication to access web services.

Token-Based Authentication

With token-based authentication, the requester obtains a token from the Spectrum™ Technology Platform server, then uses the token 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 following diagram 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`	

Operation	Description

	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.spectrumb.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.spectrumb.com/">`
	`<return>`
	`<session>ebcd7904b-07f6-15c9-82e4-71589131eb01</session>`
	`<token>eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJzaW1vbjA4OTciLCJleHAiOjE0MjU2MTI4MDAsImlhdCI6MTQyNTYwNTYwMH0.2f3qpd96fDR8</token>`
	`<username>simon0897</username>`
	`</return>`
	`</ns2:getAccessExpiringTokenResponse>`

Operation	Description
	</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.spectrumb.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.spectrumb.com/">`
	`<return>`
	`<session>6822c9b-362e-2e0e-a02a-a50a1a761323</session>`
	`<token>eyJhbGciOiJIUzI1NiJ9.eyJzZXNzaW9uIjoiZGJyf0.CE-C1VjKPa</token>`
	`<username>simon0897</username>`
	`</return>`
	`</ns2:getAccessSessionTokenResponse>`
	`</soap:Body>`
	`</soap:Envelope>`

The response contains the following 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.
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`	Use this operation to get an open 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.pb.com/">`
	`<soapenv:Header/>`
	`<soapenv:Body>`
	`<tok:getAccessToken/>`
	`</soapenv:Body>`
	`</soapenv:Envelope>`
	
	Here is a sample response:
	
	`<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">`
	`<soap:Body>`
	`<ns2:getAccessTokenResponse xmlns:ns2="http://token.security.common.server.platform.spectrum.pb.com/">`
	`<return>`
	`<token>eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJwYXVsMTIzNCIsImV4cCI6MTQ0MDU1MjI2MywiaWF0IjoxNDQwNDY2MjYzfQ.f4WB4pSk</token>`
	`<username>paul1234</username>`
	`</return>`
	`</ns2:getAccessTokenResponse>`
	`</soap:Body>`
	`</soap:Envelope>`

The response contains the following 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 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
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
```

For example,

```
HTTP/1.1

POST http://MySpectrumServer:8080/soap/ValidateAddress
Host: MySpectrumServer:8080
Authorization: Bearer
eyJlbmMiBMQI4Q0JDLUhTMjU2I5wiYWxnIjoiZGlyIn0..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 18.This service does not take any parameters.

Exposing a Service as a Web Service

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

1. Open 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 the following 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 dataflow 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 `${Option.Name}` for dataflow options and `${Data.Name}` for dataflow 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 the following the 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 dataflow is exposed the light bulb button in the Enterprise Designer tool bar indicates that the dataflow 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 as well for the Client SDK.

Micro-Batch Size

There is no limit to the number of records you can include in a request, but in general you will see the best performance when sending between 50 and 100 records in a micro-batch. We recommend that you test micro-batches of various sizes to determine the optimal micro-batch size for your

environment. Keep in mind that in some cases you may get multiple records in the response for each input record. For example, if you are performing address validation and include 10 addresses in the micro-batch, and each address matches to two possible validated addresses, you would get 20 records in the response, not just 10.

Use caution when using both micro-batches and multiple threads for requests to Spectrum™ Technology Platform. Multiple threads can overwhelm the system if each thread's micro-batch size is too large.

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.pb.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

REST 45
SOAP 447

REST

Enterprise Routing Module

What is the Routing Demo Page?

The Routing Demo page is an interactive user interface for demonstrating basic routing capabilities using the Enterprise Routing Module 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 84.

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 46.

Launching The Routing Demo Page

To launch 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 84 or **GetRoute** on page 46.
- **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 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=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 54.

Query Parameters

This operation takes the following query parameters.

Parameter	Type	Required	Description
startPoint	String	yes	The start location of the route in the format: x,y,coordSys. For example: -74.2,40.8,epsg:4326
endPoint	String	yes	The end location of the route in the format: x,y,coordSys. For example: -74.2,40.8,epsg:4326

Parameter	Type	Required	Description
intermediatePoints	String	no	A list of intermediate points to include along the route. To include in the HTTP GET request, use the format: Long,Lat,Long,Lat,…,coordsys. For example: -74.2,40.8,-73,42,epsg:4326. 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.
oip	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.
returnIntermediatePoints	Boolean	no	Whether to return the intermediate points with the route response. The default is false. For any value other that 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 *oip* is set to true, then this option will return intermediate points in optimized order.
destinationSrs	String	no	The coordinate system to return the route and resulting geometries. The default is the coordinate system of the data used.
optimizeBy	String	no	The type of optimizing to use for the route. Valid values are *time* or *distance*. The default is time.
returnDistance	Boolean	no	The route directions include the distance traveled. The default is true.
distanceUnit	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).
returnTime	Boolean	no	The route directions include the time it takes to follow a direction. The default is true.

Parameter	Type	Required	Description
timeUnit	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	The language the travel directions should be returned, only if route directions are returned (if `directionsStyle` is defined as Normal or Terse). The default being English (en). Directions can be returned in the following languages: **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.

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	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: **None** No directions returned. Default, if not specified. **Normal** Directions are returned in a full format, appropriate for web-based applications. **Terse** Directions are returned in a short format, appropriate for mobile applications.
segmentGeometryStyle	String	no	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: **None** No geometric representation of a segment will be returned. Default, if not specified. **End** Each segment of the route will be returned with just its endpoints in a LineString. **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.
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
historicTrafficTimeBucket	String	no	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: **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.

Examples

Simple Route with start and end points.

```
http://www.pb.com/rest/$patial/em/databases/usroutedatabase.json?q=route&startPoint=-73.97,40.79,epsg:4326&endPoint=-73.98,40.74,epsg:4326
```

Response

```
{
  "time": 7.67,
  "timeUnit": "min",
  "distance": 8865,
  "distanceUnit": "m"
}
```

Route with intermediate points.

```
http://www.pb.com/rest/Spatial/erm/databases/usroutedatabase.json?
q=route&startPoint=-73.970257,
40.794045,epsg:4326&endPoint=-73.972103,40.786605,
epsg:4326&intermediatePoints=-73.976266,
40.788717,-73.973562,40.792193,-73.971802,40.794630,
epsg:4326&oip=true&returnIntermediatePoints=true
```

Response

```
"intermediatePoints": {
 "type": "MultiPoint",
 "coordinates": [
  [-73.971802,
   40.79463
  ],
  [-73.973562,
   40.792193
  ],
  [-73.976266,
   40.788717
  ]
 ]
}
```

Route with directions enabled.

```
http://www.pb.com/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
],
...
]
}
}
]
}
```

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	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: **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. **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). **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). **exclude** This is astring 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).

Update Type	Description
segmentID	

Update Type	Description
	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:
percentage	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.
speed	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).
speedAdjustment	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).
exclude	This is astring 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).
roadType	This is a string value to change the value of the road type for the segment for the route calculation. The `roadType` can be one of the following: • 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 • mormal 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	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: **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. **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). **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).

Example transient update HTTP POST payload.

```
{
  "transientUpdates": [
    {
        "segmentID": "specify a segment id",
        "updates": [
            {"exclude": "Y"},
            {"roadType": "a road type"},
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}

        ]
    },
    {
        "point": "specify a geojson point",
        "updates": [
            {"exclude": "Y"},
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}
```

```
        ]
    },
    {
        "roadType": "specify a road type",
        "updates": [
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "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 it shorter/faster than a route with no barrier.

Option	Description
vehicleAttributes	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: **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`. **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`. **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`. **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`. **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`. **Note:** You need to specify either weight/height or length/width alongwith its corresponding unit.

Examples

Without Vehicle Restrictions

Request:

```
HTTP GET
/rest/$patial/em/databases/US_CAR.json?q=route&startPoint=-74.7221203,42.9737073,epsg:4326&endPoint=-74.6671887,42.8097083,epsg:4326
```

Response:

```
{
   "distance": 24.87,
   "distanceUnit": "mi",
```

```
   "time": 36.57,
   "timeUnit": "min"
}
```

With Vehicle Restrictions

Request:

```
HTTP POST
/rest/spatial/em/databases/US_CVR.json?q=route&startPoint=-74.7221203,42.9737073,epsg:4326&endPoint=-74.6671887,42.8097083,epsg:4326

POST BODY: Content-Type:application/json {CVR Data}
```

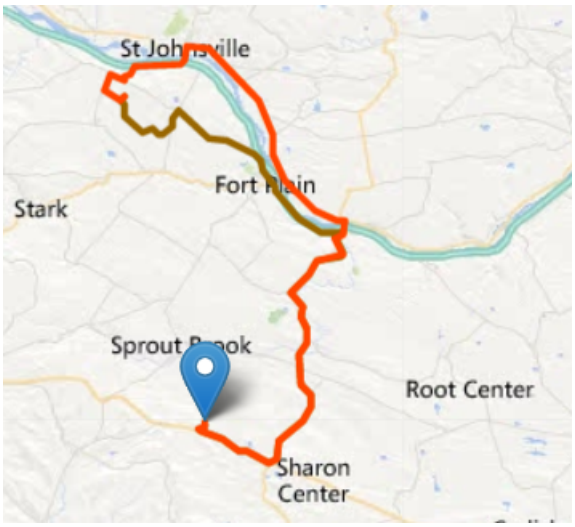
Commercial Vehicle Restriction HTTP POST payload.

```
{
     "cvr":{
     "looseningBarrierRestrictions":"n","vehicleAttributes":{

"vehicleType":"ALL","heightUnit":"meter","height":"4","weightUnit":"Kilogram","weight":"40000"

                                                                    }
            }
}
```

Response:

```
  {
  "distance": 27.92,
  "distanceUnit": "mi",
  "time": 37.48,
  "timeUnit": "min"
  }
```

The two routes in the map below show the CVR applied for the same start and end locations. The route displayed in the brown color is the one without CVR and the route displayed in the red color is with CVR. Notice the deviation in route in the beginning of the journey; this is due to the height and weight restrictions applied to them.



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 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=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 69.

Query Parameters

This operation takes the following query parameters.

Parameter	Type	Required	Description
startPoints	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
endPoints	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
destinationSrs	String	no	The coordinate system to return the route and resulting geometries. The default is the coordinate system of the data used.
optimizeBy	String	no	The type of optimizing to use for the route. Valid values are *time* or *distance*. The default is time.
returnDistance	Boolean	no	The route directions include the distance traveled. The default is true. Both returnDistance and returnTime parameters cannot be false in the same request.
distanceUnit	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).
returnTime	Boolean	no	The route directions include the time it takes to follow a direction. The default is true. Both returnDistance and returnTime parameters cannot be false in the same request.
timeUnit	String	no	The units to return time. The default is min (minute). Available values are: min(minute), msec (millisecond), s(second), h(hour).
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
returnOptimalRoutesOnly	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 *optimizeBy* parameter.
historicTrafficTimeBucket	String	no	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: **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.

Examples

Cost matrix route with two start and two end points.

```
http://localhost:8080/rest/routing/1/costMatrix?start=40.759011,-73.984472&start=40.748433,-73.985656&end=40.7
```

Response

```
{
    "matrix":
```

```
[
    {
        "startPoint":
        {
            "type": "Point",
            "coordinates":
            [
                -73.56567672202618,
                40.554384822358614
            ],
            "crs":
            {
                "type": "name",
                "properties":
                {
                    "name": "epsg:4322"
                }
            }
        },
        "endPoint":
        {
            "type": "Point",
            "coordinates":
            [
                -73.34345711862802,
                40.66688488742393
            ],
            "crs":
            {
                "type": "name",
                "properties":
                {
                    "name": "epsg:4322"
                }
            }
        },
        "distance": 35.268,
        "distanceUnit": "km
    },
    {
        "startPoint":
        {
            "type": "Point",
            "coordinates":
            [
                -73.46567684021008,
                40.454384834155185
            ],
            "crs":
            {
                "type": "name",
                "properties":
                {
```

```
                        "name": "epsg:4322"
                    }
                }
            },
            "endPoint":
            {
                "type": "Point",
                "coordinates":
                [
                    -73.34345711862802,
                    40.66688488742393
                ],
                "crs":
                {
                    "type": "name",
                    "properties":
                    {
                        "name": "epsg:4322"
                    }
                },
                "distance": 44.444,
                "distanceUnit": "km"
            }
        ]
    }
```

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 wrapper.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	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: **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. **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). **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). **exclude** This is astring 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).

Update Type	Description
segmentID	

Update Type	Description
	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:
percentage	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.
speed	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).
speedAdjustment	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).
exclude	This is astring 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).
roadType	This is a string value to change the value of the road type for the segment for the route calculation. The `roadType` can be one of the following: • 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 • mormal 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	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: **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. **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). **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).

Example transient update HTTP POST payload.

```
{
  "transientUpdates": [
    {
        "segmentID": "specify a segment id",
        "updates": [
            {"exclude": "Y"},
            {"roadType": "a road type"},
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}

        ]
    },
    {
        "point": "specify a geojson point",
        "updates": [
            {"exclude": "Y"},
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}
```

```
        ]
    },
    {
        "roadType": "specify a road type",
        "updates": [
            {"percentage": 26.0},
            {"speed": { "velocity": 25, "velocityUnit": "kph"}},
            {"percentage": -13.5},
          {"speedAdjustment" : { "velocity": 5, "velocityUnit": "kph"}}

        ]
    }
]
}
```

Example geojson point.

```
"point": {"type": "Point", "crs":
          {
              "type": "name",
              "properties":
                {"name": "epsg:4326"}
               },
               "coordinates":
               [-73.979102,40.785193]
      }
```

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 it shorter/faster than a route with no barrier.
vehicleAttributes	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: **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`. **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`. **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`. **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`. **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`. **Note:** You need to specify either weight/height or length/width alongwith its corresponding unit.

Examples

Without Vehicle Restrictions

Request:

```
HTTP GET
/rest/spatial/em/databases/US_CAR.json?q=route&startPoint=-74.7221203,42.9737073,epsg:4326&endPoint=-74.6671887,42.8097083,epsg:4326
```

Response:

```
{
   "distance": 24.87,
   "distanceUnit": "mi",
   "time": 36.57,
   "timeUnit": "min"
}
```

With Vehicle Restrictions

Request:

```
HTTP POST
/rest/spatial/erm/databases/US_CVR.json?q=route&startPoint=-74.722203,42.973073,epsg:4326&endPoint=-74.6671887,42.8097083,epsg:4326

POST BODY: Content-Type:application/json {CVR Data}
```

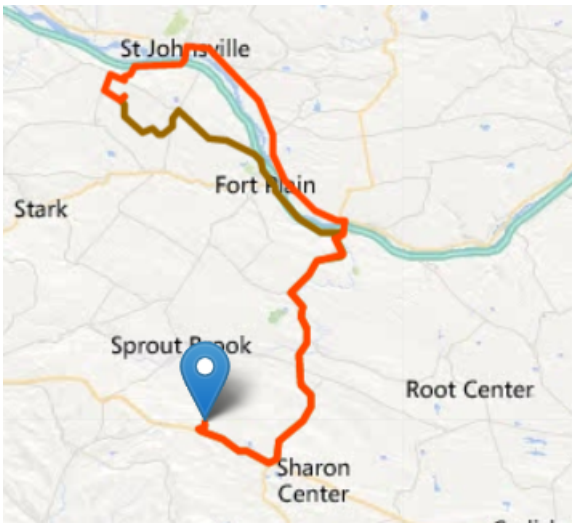
Commercial Vehicle Restriction HTTP POST payload.

```
{
     "cvr":{
     "looseningBarrierRestrictions":"n","vehicleAttributes":{

"vehicleType":"ALL","heightUnit":"meter","height":"4","weightUnit":"Kilogram","weight":"40000"

                                                                                  }
            }
}
```

Response:

```
  {
  "distance": 27.92,
  "distanceUnit": "mi",
  "time": 37.48,
  "timeUnit": "min"
  }
```

The two routes in the map below show the CVR applied for the same start and end locations. The route displayed in the brown color is the one without CVR and the route displayed in the red color is with CVR. Notice the deviation in route in the beginning of the journey; this is due to the height and weight restrictions applied to them.



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"
      }
}
```

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 Enterprise Routing Module 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
destinationSrs	String	no	The coordinate system to return the segment data and resulting geometry. The default is the coordinate system of the data used.
distanceUnit	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).
timeUnit	String	no	The units to return time. The default is min (minute). Available values are: min(minute), msec(millisecond), s(second), h(hour).
velocityUnit	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).
angularUnit	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).
segmentGeometryStyle	String	no	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: **None** No geometric representation of a segment will be returned. Default, if not specified. **End** Each segment of the route will be returned with just its endpoints in a LineString. **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.

Examples

Return segment data specifying a point.

```
http://www.pb.com/rest/Spatial/ern/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,
 "segmentLengtUnit": "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": [[…]]
 }
}]
```


Return segment data specifying a segmentID.

```
http://www.pb.com/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
}
```

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 89.

Query Parameters

This operation takes the following query parameters.

Parameter	Type	Required	Description
point	String	yes	The start location from where to calculate the travel boundary in the format: x,y,coordSys. For example: -74.2,40.8,epsg:4326
costs	Double	yes	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.
costUnit	String	yes	The type of metric used to calculate the travel boundary. Available distance values are: m(meter), km(kilometer), yd(yard), ft(foot), mi(mile). Available time values are: min(minute), msec(millisecond), s(second), h(hour).

Parameter	Type	Required	Description
maxOffroadDistance	Double	no	The maximum distance to allow travel off the road network using the *maxOffroadDistanceUnit*. 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.
maxOffroadDistanceUnit	String	no	The distance unit defining the *maxOffroadDistance*. You must also define maxOffroadDistance if you define this parameter. Available distance values are: m(meter), km(kilometer), yd(yard), ft(foot), mi(mile).
destinationSrs	String	no	The coordinate system to return the travel boundary geometries. The default is the coordinate system of the data used (for example, epsg:4326).
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 true.
returnHoles	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.
returnIslands	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.
simplificationFactor	Integer	no	What percentage of the original points should 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	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: **Donut** Each boundary is determined by subtracting out the next smallest boundary. This is the default method. **Encompassing** Each boundary is determined independent of all others.
historicTrafficTimeBucket	String	no	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: **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.

Examples

Travel boundary with a single cost.

```
http://www.pb.com/rest/$patial/em/databases/usroutedatabase.json?q=travelBoundary&point=-77.02609,38.871256,epsg:4326&costs=5&costUnit=m
```

Response

```
{
"travelBoundary": {
  "costs": [
 {
  "cost": 5,
  "costUnit": "m",
  "geometry":
{"type":"MultiPolygon","crs":{"type":"name","properties":{"name":"epsg:
 4326"}},"coordinates":[[[...]]]}
 }
  ]
}
}
```

Travel boundary with multiple costs.

```
http://www.pb.com/rest/spatial/em/databases/usroutedatabase.json?q=travelBoundary&point=-77.09269,38.871256,epsg:4326&costs=2,5&costUnit=m
```

Response

```
{
"travelBoundary": {
  "costs": [
     {
                "cost": 2,
                "costUnit": "m",
  "geometry":
"{"type":"MultiPolygon","crs":{"type":"name","properties":{"name":"epsg:
 4326"}},"coordinates":[[[...]]]}"
 },
 {
  "cost": 5,
  "costUnit": "m",
  "geometry":
"{"type":"MultiPolygon","crs":{"type":"name","properties":{"name":"epsg:
 4326"}},"coordinates":[[[...]]]}"
 }
  ]
}
}
```

GetTravelBoundary HTTP POST Options

HTTP POST URL Format

In addition to the regular HTTP GET parameters, you can add a 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}
```

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
DefaultAmbientSpeed	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.
AmbientSpeedUnit	The unit of measure to use to calculate the ambient speed. Available speed units are: MPH(miles per hour), KPH(kilometers per hour), MTPS(meters per second), MTPM(meters per minute)

Parameter	Description
AmbientSpeed.RoadType	

Parameter	Description
	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 *AmbientSpeedUnit*. The following road types can be used: • 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 • mormal 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
	• secondary highway dense urban • secondary highway rural • secondary highway suburban • secondary highway urban

Example ambient speed update HTTP POST payload.

```
{
  "DefaultAmbientSpeed": 45,
  "AmbientSpeedUnit" : "MPH"

  "AmbientSpeed.RoadType.PrimaryHighwayUrban":15,
  "AmbientSpeed.RoadType.SecondaryHighwayUrban":10
}
```

PersistentUpdate

Description

The PersistentUpdate service allows a user to override aspects of the network. The overrides can be done on a per-road type, at a specific point or at a specific segment. The persistent update is valid only for a specific data source and may not be valid after a data update.

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
- List persistent updates

> **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. The response from the REST 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.

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/erm/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 Enterprise Routing Module Routing 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/erm/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 Enterprise Routing Module Routing Database Resource tool.

Query Parameters

The HTTP POST operation takes the following query parameters.

Parameter	Type	Required	Description
exclude	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.
velocity	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 *velocityUnit* parameter.

Parameter	Type	Required	Description
velocityUnit	String	no	This is a unit of speed for the *velocity* or *velocityAdjustment* (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.
velocityAdjustment	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 *velocityUnit* parameter.
velocityPercentage	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
resetType	String	no	Reset (undo) a type of update for a point. **speed** Reset the speed update for a specific point. **exclude** Reset the exclude for a specific point.

Examples

Exclude a point (HTTP POST)

```
http://www.pb.com/rest/spatial/erm/databases/usroutedatabase/persistentUpdates.json?point=-73.6,43.5,epsg:4326&exclude=true
```

Remove a point exclude persistent update (HTTP DELETE)

```
http://www.pb.com/rest/Spatial/erm/databases/usroutedatabase/persistentUpdates.json?point=-73.6,43.5,epsg: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 Enterprise Routing Module 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?mo=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 Enterprise Routing Module Routing Database Resource tool.

Query Parameters

The HTTP POST operation takes the following query parameters.

Parameter	Type	Required	Description
exclude	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.
velocity	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 *velocityUnit* parameter.
velocityUnit	String	no	This is a unit of speed for the *velocity* or *velocityAdjustment*. 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.
velocityAdjustment	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 *velocityUnit* parameter.
velocityPercentage	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
roadType	String	no	

Parameter	Type	Required	Description
			This is an update where you define the new road type of the segment. The road type can be one of the following: • 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
			• secondary highway suburban • secondary highway urban

Reset Parameter

The HTTP DELETE operation takes the following query parameter.

Parameter	Type	Required	Description
resetType	String	no	Reset (undo) a type of update for a segment. **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/erm/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/erm/databases/US_NE/persistentUpdates/segments/efed6c1: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/erm/databases/US_NE/persistentUpdates/segments.json?segments=
```

Return a list of updates for the multiple segments (HTTP GET)

```
http://<server>:<port>/rest/Spatial/erm/databases/US_NE/persistentUpdates/segments.json?segments=27e2f762:4718d9,7e3396fc:14c9c2c
```

Remove a segment speed persistent update (HTTP DELETE)

```
http://<server>:<port>/rest/Spatial/erm/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 Enterprise Routing Module 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 Enterprise Routing Module 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
velocity	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 *velocityUnit* parameter.
velocityUnit	String	no	This is a unit of speed for the *velocity* or *velocityAdjustment*. 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.

Parameter	Type	Required	Description
velocityAdjustment	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 *velocityUnit* parameter.
velocityPercentage	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/erm/databases/usroutedatabase/persistentUpdates/roadTypes/ferry.json?velocity=5&velocityUnits=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 Enterprise Routing Module Routing Database Resource tool.

Example

Removes all persistent updates for the US_NE routing database resource.

```
http://www.pb.com/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 Enterprise Routing Module Routing Database Resource tool.

Query Parameters

This operation takes the following query parameter.

Parameter	Type	Required	Description
velocityUnit	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 get metadata about the routing engine deployed. This metadata allows users to explore a service and its capabilities, therefore optimizing their experience using the routing services.

This is available as REST service only.

Request URL

```
http://<server>:<port>/rest/Spatial/erm/v1/capabilities.json
```

Query Parameters

Parameter	Required	Description
acceptVersions	Optional	Placeholder (not functional)
sections	Optional When omitted, returns information about all sections.	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 ServiceIdentification, ServiceProvider, operationsMetadata and databases.

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 will look as follows:

```
"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 same as what is available in the `getCapabilities.json` configuration file.

This file is present in `<SpectrumFolder>\server\modules\routing`. The server needs to be restarted for any change made to the file to have an effect. The administrator determines which information the user should get and can modify or delete 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 will look like as follows:

```
"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 will look like as follows:

```
{"operationsMetadata": [{
  "name": "GetRoute",
  "DCP": {
   "HTTP": {
    "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=route",

    "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=route"

   }
  },
  "parameter": {
   "name": "OutputFormat",
   "value": "text/json"
  }
 },

 {
  "name": "GetRouteCostMatrix",
  "DCP": {
   "HTTP": {
    "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=routeCostMatrix",

    "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=routeCostMatrix"

   }
  },"parameter": {
   "name": "OutputFormat",
   "value": "text/json"}

 },

 {

  "name": "GetTravelBoundary",
  "DCP": {
```

```
    "HTTP": {
     "GET":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=travelBoundary",

     "POST":
"<schema>://<server>:<port>/rest/Spatial/erm/databases/<DB_NAME>.json?q=travelBoundary"

    }
   },
   "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.jason"

   }
  },
  "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.

Request URL (All Datasets)

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets.json
```

Request URL (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 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. Otherwise, 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`

Metadata information is available for `E:\\db\\ERM-US\\2014.09\\driving\\northeast` and `E:\\db\\ERM-US\\2014.09\\driving\\midwest`, but not for `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",
                    "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 Midwest",
            "name": "USA",
            "product": "Spatial",
            "vintage": "September 2015"
        },
        {
            "component": "routing",
            "description": "USA Test dataset",
            "ext":
            {
                "bbox":
                [
                    68.291015625,
                    7.9721977144,
                    97.55859375,
                    35.4606699515
                ],
                "crs": "epsg:4326",
                "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 NE",
```

```
                "name": "USA",
                "product": "Spatial",
                "vintage": "September 2015"
            }
        ]
    }
```

Sample Request (Single Dataset):

```
http://<server>:<port>/rest/Spatial/erm/v1/datasets/US%20Midwest.json
```

Response:

```
{
        "dataSets":
        [
            {
                "component": "routing",
                "description": "USA Test dataset",
                "ext":
                {
                    "bbox":
                    [
                        68.291015625,
                        7.9721977144,
                        97.55859375,
                        35.4606699515
                    ],
                    "crs": "epsg:4326",
                    "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 Midwest",
                "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"
  }
```

```
  ]
}
```

Enterprise Tax Module

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.
- **Property tax districts**—Areas designated for the collection of taxes imposed on property owners to support local government facilities and services based on the property's location. Examples

include taxes collected for districts to pay for schools, police, or other services. Property tax districts are created by local governments.

- **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 Pitney Bowes sales representative for more information.
- **Sales and Use Tax Rates**—Using the optional Pitney Bowes 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 the Enterprise Tax Module.

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.DataTypeName": "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.pb.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.AddressLine2	String [100]	Third address line
Data.AddressLine4	String [100]	Fourth address line

Parameter	Format	Description
Data.BufferWidth	String [10]	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 852. This field overrides the value specified in the `Option.DefaultBufferWidth` parameter. Specify the border width in the units specified by the `Option.DistanceUnits` parameter. If you do not specify a buffer width in this input field, the default is used.
Data.CandidateID	String [1]	For a multi-match outcome, indicates which match result should be returned; valid values are 1-9.
Data.City	String [50]	City name
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.

Parameter	Format	Description
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 852. This field overrides the value specified in the `Option.DefaultBufferWidth` parameter. Specify the border width in the units specified by the `Option.DistanceUnits` 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.	
Option.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. **Y** Use the GeoTAX Auxiliary file for matching. (default) **N** Do not use the GeoTAX Auxiliary file for matching.
Option.UseAuxiliaryFile	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. **Y** Use the User Auxiliary file for matching. **N** Do not use the User Auxiliary file for matching. (default)

Parameter	Description
Option.UseStateProvidedFile	Specifies whether to attempt a match to the state-supplied file. Use this option in combination with `FileSearchOrder` to specify a state-supplied file to use. 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. 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. 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. **Y** Use the State-supplied file for matching. **N** Do not use the State-supplied file for matching. (default) **Note:** You must install the appropriate State-supplied file to use these options. For instructions, see the *Spectrum™ Technology Platform Installation Guide*.
Option.FileSearchOrder	Specifies which state-supplied file to use. This option only takes effect if you specify `Option.UseStateProvidedFile=Y`. One of the following: **FLOnly** Use only the Florida-native formatted file. (default) **TSOnly** Use only the TS-158 formatted file.
Option.UseRelaxedSecondaryMatching	Specifies whether input addresses with secondary information are matched to records without secondary information. This option applies only to Florida-native files. **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.	

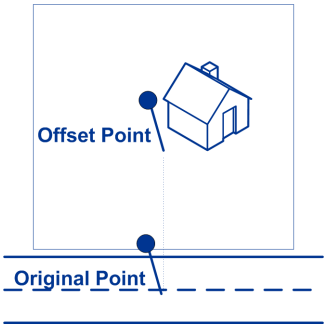
Parameter	Description
Option.GsMatchMode	Match modes determine the leniency used to make a match between your input and the reference database. Select a match mode based on the quality of your input and your desired output. For example, if you have an input database that is prone to errors, you may want to select the relaxed match mode. **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. **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. **1-99 miles** Search radius. (default = 0 miles)
Option.GsEnableFirstLetterExpanded	Looks for the correct first letter of a street address if the first letter is missing or incorrect. The Enterprise Tax Module searches through the alphabet looking for possible correct first letters to complete the street address. **Note:** This feature is disabled by default and cannot be enabled in Exact mode. **Y** Enable first letter change matches. **N** Do not allow first letter change matches. (default)

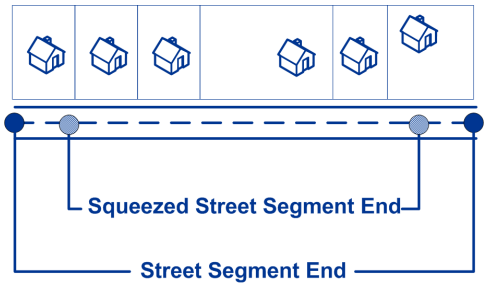
Parameter	Description
Option.GsEnableRangedAddress	Matches to a house range input. Some business locations are identified by address ranges. For example, a shopping plaza could be addressed as 10-12 Front St. - this is how business mail is typically addressed to such a business location. When this feature is enabled, the address range is geocoded to the interpolated mid-point of the range. **Note:** This feature is disabled by default and cannot be enabled in Exact mode. **Y** Allow address range matches. **N** Do not allow address range matches. (default)
Option.GsAlternateLookup	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. **1** Searches for street name, but if there isn't a match, will use the firm name. **2** Looks up the firm name, but if there isn't a match, will use the street name. **3** Searches only street records. (default)
Option.GsMultiMatchResolution	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: **N** No matches are returned. (default) **R** Return the first match candidate in the list. **A** The information for all the match candidates is returned.

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
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. **60** Sixty feet offset from street centerline.

Parameter	Description
Option.Squeeze	Specifies if the street end points should be "squeezed" when determining the geocode of an address in street-level matching. When `Option.Squeeze` 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.  **Y** Apply squeeze. **N** Do not apply squeeze. (default)
Option.LatLongFormat	Indicates the desired format for the returned latitude/longitude. One of the following: **PreZero** Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W. **PreZeroDecimal** Decimal degrees using directional indicator. For example, 090.000000N180.000000W. (default) **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. **DegMinSec** Degrees, minutes, seconds. For example, 90 00 00N180 00 00W.

Expanded Geocoding options: These options enable additional geocoding functionality.

Parameter	Description
Option.GsEnableAddressPointInterpolation	Address point interpolation uses a patented process that improves upon regular street segment interpolation by inserting point data into the interpolation process. **Note:** This feature is only for use with point-level geocoding. 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, the Enterprise Tax Module attempts to find high and low boundary address points to use for address point interpolation. **Y** Enable address point interpolation. **N** Disable address point interpolation. (default)
Minimum geocode quality	
Option.GsEnableGeographicFallback	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. **Y** If a definitive match cannot be made, then return the next higher level geographic centroid. **N** Disable geographic fallback feature. (default)

Parameter	Description
Option.GsEnableStreetCentroid	If an input street address cannot be found using the street number and name, the Enterprise Tax Module then searches the input ZIP Code or city/state for the closest match. If the Enterprise Tax Module 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. 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 Enterprise Tax Module does not change the street name on the output address. **Y** If a street or point match cannot be made, then return a street level centroid. **N** Do not return a street level centroid if a match cannot be made. (default) **Note:** This feature should only be used for exception processing or research. It should not be used in a production process.
Boundary matching: These 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: **Feet** Distances are measured in feet. (default) **Meters** Distances are measured in meters.
Option.DefaultBufferWidth	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, the Personal Property Tax District (PTD) file and Place and MCD files. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. For more information about buffers, see **Buffering** on page 852.

Parameter	Description
Option.DefaultUserBufferWidth	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 852. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. **Note:** To use buffers, the user-defined boundary file must support buffers.

Output Data Options

Data options control the data returned by AssignGeoTAXInfo.

Parameter	Description
Option.GeoTAXOutputRecordType	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 `Option.OutputFields` option. • **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 You can also specify one, and only one, of the following: **I** Insurance Premium Tax District (IPD) **P** Property Tax District (PTD) **R** Payroll Tax District (PAY) **S** Special Purpose Tax District (SPD) For a description of the fields in each output group, see **Response** on page 134. **Note:** If you specify `W`, to obtain the best payroll system tax code match possible.

Parameter	Description
Option.TaxKey	If you use third-party tax compliance software from Vertex or Taxware, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following: **N** Do not return either the Taxware or Vertex jurisdiction codes (default). **T** Return the Taxware jurisdiction code for the address. **Y** Return the Vertex jurisdiction code for the address. Select this option if you obtained a MatchMaster file from Vertex. **V** Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Pitney Bowes. **Note:** The Special Purpose District data is needed to achieve the best results from this option.
Option.TaxRate	Indicates the sales and use tax rate type to return or none: **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:** The Special Purpose District data is needed to achieve the best results from this option.
Option.OutputFields	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. For a list of the fields included in each data type, see **Response** on page 134.

Output Format

Output format options control how AssignGeoTAXInfo formats output data.

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

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	Indicates the file from which the match was obtained. One of the following: • 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	General Return Code indicating the type of match. **5** ZIP Code match **9** ZIP+4 Code 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
AddressMatch.Lastline*	61	The complete matched last address line (city, state, and postal code).
AddressMatch.LocationCode*	5	The Location Code indicates the methodology used to complete the geocode and may also provide some information about the quality of the geocode. For the list of location codes, see **Location Codes.**
AddressMatch.MatchCode*	5	The Match Code indicates the portions of the address that matched or did not match to the reference file. For the list of match codes, see **Match Codes.**
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	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. The AddressMatch.PBKey field has "P" as the leading character, for example: P00001XSF1IF.
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	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

Response Element	Max. Field Length with null terminator	Description

Confidence*	4	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:
		
		Census.MatchCode + LatLong.StreetMatchCode + LatLong.MatchCode
		
		The maximum confidence score is 100, so if this calculation results in a value greater than 100, the Confidence score is returned as 100.
		
		The following values are used:
		
		• Census.MatchCode
		  • 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
		
		• LatLong.StreetMatchCode
		  • H = 5
		  • L = 15
		  • S = -10
		  • Z = -5
		  • null = 0

Response Element	Max. Field Length with null terminator	Description

		• LatLong.MatchCode
		  • 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 151.		
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	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.txb file. For more information on buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
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.txb file.
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.

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 852. **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	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.txb file. For more information on buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
MCD.DistanceToBorder*	10	Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb 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 `AddressMatch.GenRC` 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 `LatLong.MatchCode` 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
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

Response Element	Max. Field Length with null terminator	Description
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.

Response Element	Max. Field Length with null terminator	Description
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 119. 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.

Response Element	Max. Field Length with null terminator	Description
PTC*n*.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*n*.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*n*.PayrollDescription	41 per PTC	A description of the purpose of this payroll code.
PTC*n*.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 jurisdiciton data in the output. Contents returned when available; otherwise, left blank.

Response Element	Max. Field Length with null terminator	Description
Confidence.SurfaceType	3	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. **0** Undefined **1** The search failed - the address was not found. **2** Intersection confidence surface generated. **3** Interpolated street segment. **4** Point-level match. **5** State confidence surface generated. **6** County confidence surface generated. **7** City confidence surface generated. **8** Reserved **9** A ZIP Code confidence surface generated. **10** A ZIP+2 confidence surface generated. **11** A ZIP+4 confidence surface generated. **12** Reserved **13** A street centroid confidence surface generated.
GeoTAXKey	10	The value in this field varies depending on the option you specified in the Option.TaxKey option: If you specified `T`, GeoTAXKey contains the proprietary codes used in Taxware tax compliance software. You can use this code in your Taxware application to find out the tax rate for the jurisdiction. If you specified `Y` or `V`, GeoTAXKey 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.

Response Element	Max. Field Length with null terminator	Description
GeoTAXKey.MatchCode	2	Return code denoting the level of match obtained against the Vertex or Taxware cross reference files. **E** Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name. **P** Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code. **A** Alternate match using two fields: ZIP Code, FIPS place name. **N** Record is default coded based on valid state code. **null** No matching record found.
GeoTAXKey.MatchLevel	12	A description of the value returned in the `GeoTAXKey.MatchCode` field. **Exact** Exact match. See description in GeoTAXKey.MatchCode. **Partial** Partial match. See description in GeoTAXKey.MatchCode. **Alternate** Alternate match. See description in GeoTAXKey.MatchCode. **DefaultCode** Record is default coded. See description in GeoTAXKey.MatchCode. **NoMatch** No matching record found.
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	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 852. **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.
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*n*.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	Indicates where in the polygon the address resides in relation to the edge of the area. Buffer width is specified by the option Option.DefaultUserBufferWidth or by the input field Data.BufferWidth. One of the following: **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. For more information, see **Buffering** on page 852.
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 Option.DistanceUnits.
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 `Option.GeoTAXOutputRecordType` = `I`.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, PTDs, 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*n*.BoundaryBuffer.BufferRelation	2 per IPD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
IPD*n*.BoundaryBuffer.DistanceToBorder	10 per IPD	Indicates the distance from the address to the border of the district.
IPD*n*.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*n*.DistrictID	11 per IPD	IPD ID.
IPD*n*.DistrictName	61 per IPD	IPD name.
IPD*n*.DistrictType	7 per IPD	IPD district type.
IPD*n*.UpdateDate	7 per IPD	IPD update date (MMYYYY).

Response Element	Max. Field Length with null terminator	Description
IPD*n*.VersionDate	7 per IPD	IPD compiled date (MMYYYY).
IPD*n*.Notes	21 per IPD	Tax code descriptions. For example: 01, 33, A, B
IPD*n*.ChangeDate	7 per IPD	IPD change date.
IPD*n*.EffectiveDate	7 per IPD	MMDDYY - Identifies when district becomes active - State supplied For example: 010108
IPD*n*.ExpirationDate	7 per IPD	MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009
IPD*n*.FireRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.CasualtyRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.VehicleRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MarineRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.HealthRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.LifeRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.OtherRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MinimumRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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 address 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 119. To include this data in the output, set `Option.GeoTAXOutputRecordType` = `R`.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, PTDs, 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*n*.BoundaryBuffer.BufferRelation	2 per PAY	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option Option.DefaultBufferWidth or by the input field Data.BufferWidth. One of the following: **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. For more information, see **Buffering** on page 852.
PAY*n*.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 Option.DefaultBufferWidth.
PAY*n*.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*n*.DistrictID	11 per PAY	PAY district ID.
PAY*n*.DistrictName	61 per PAY	PAY district name.
PAY*n*.DistrictType	7 per PAY	PAY district type.
PAY*n*.ID	11 per PAY	PAY ID.

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

Property Tax Districts

The table below lists the output fields that contain Property Tax Districts (PTD) data. For more information about property tax districts, see **AssignGeoTAXInfo** on page 119. To include this data in the output, set `Option.GeoTAXOutputRecordType` = `P`.

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

Response Element	Max. Field Length with null terminator	Description
NumberPTDsFound	3	Number of PTDs returned.
PTD*n*.BoundaryBuffer.BufferRelation	2 per PTD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
PTD*n*.BoundaryBuffer.DistanceToBorder	10 per PTD	Indicates the distance from the address to the border of the district. The distance is in the units specified by the option `DistanceUnits` `Option.DistanceUnits`.
PTD*n*.BoundaryConfidence	4 per PTD	Provides the percentage overlap of the geocode to the PTD boundary polygon layer. The returned percentage value describes the probability that the point falls in the PTD boundary area.
PTD*n*.DistrictID	11 per PTD	PTD district ID.

Response Element	Max. Field Length with null terminator	Description
PTD*n*.DistrictName	61 per PTD	PTD district name.
PTD*n*.DistrictType	2 per PTD	PTD district type. Only returned for Indiana. **R** Reporting district **B** Billing district
PTD*n*.JurisdictionID	11 per PTD	PTD jurisdiction ID.
PTD*n*.UpdateDate	7 per PTD	PTD update date.

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 119. To include this data in the output, set `Option.GeoTAXOutputRecordType` = `S`.

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

Response Element	Max. Field Length with null terminator	Description
NumberSPDsFound	3	Number of SPDs returned.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.BoundaryBuffer.BufferRelation	2 per SPD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
SPD*n*.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 `Option.DefaultBufferWidth`.
SPD*n*.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*n*.CompiledDate	7 per SPD	SPD compiled date.
SPD*n*.DistrictCode	4 per SPD	3-digit district type code.
SPD*n*.DistrictName	61 per SPD	SPD name.
SPD*n*.DistrictNumber	6 per SPD	SPD district number.
SPD*n*.EffectiveDate	7 per SPD	SPD effective date.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.UpdateDate	7 per SPD	SPD update date.
SPD*n*.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 Pitney Bowes 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 Pitney Bowes 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.

Response Element	Max. Field Length with null terminator	Description
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).
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	This field contains a return code if the GeoTAX engine experiences an abnormal termination. **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. The first character indicates the file (or set of files affected). **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 The second position is one of the following: **E** Fatal issue, program terminating **F** Expired database **I** Informational

Response Element	Max. Field Length with null terminator	Description
GTX.ErrorDescription*	81	

Response Element	Max. Field Length with null terminator	Description
		If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally. **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. 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" 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" 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 = "

Response Element	Max. Field Length with null terminator	Description
		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 `WN` 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.
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.
Status	2	Reports the success or failure of the match attempt. **null** Success **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.
Status.Code	12	If AssignGeoTAXInfo could not process the address, this field will show the reason. Currently there is one possible value for this field: Invalid Address.

Response Element	Max. Field Length with null terminator	Description
Status.Description	64	If AssignGeoTAXInfo could not process the address, this field will show a description of the failure. One of the following: 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 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 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 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 the Enterprise Tax Module.

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.pb.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. **N** North **S** South
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.

Parameter	Description	
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	Indicates the format of the input latitude/longitude. The options are:	
	DegMinSec	For example 90 00 00N180 00 00W.
	PreZero	(090000000N180000000W). (default)
	PreZeroDecimal	(090.000000N180.000000W)
	Decimal	(90.000000-180.000000)
	DecimalAssumed	(90000000-180000000)
Option.ReturnUnits	Indicates the measurement units returned for distance calculation: • 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 Pitney Bowes 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 ReverseGeoTAXInfo Lookup 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.
- **Property tax districts**—Areas designated for the collection of taxes imposed on property owners to support local government facilities and services, based on the property's location. Examples include taxes collected for districts to pay for schools, police, or other services. Property tax districts are created by local governments.
- **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 Pitney Bowes sales representative for more information.

Using the optional Pitney Bowes 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
Data.InputLongitude	000.000000 or without the decimal point 000000000, or 00.000000 or without the decimal point 00000000

ReverseGeoTAXInfoLookup is part of the Enterprise Tax Module.

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.pb.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: **Feet** Distances are measured in feet. (default) **Meters** Distances are measured in meters.
Default buffer widths	
Option.DefaultBufferWidth	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, the Property Tax Districts (PTD) file, Place and MCD files. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. For more information about buffers, see **Buffering** on page 852.
Option.DefaultUserBufferWidth	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 852. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. **Note:** To use buffers, the user-defined boundary file must support buffers.

Output Data Options

Data options control the data returned by ReverseGeoTAXInfoLookup.

Parameter	Description
Option.GeoTAXOutputRecordType	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 checkbox, and list only those fields you want returned in `Option.OutputFields`. • **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 You can also specify one, and only one, of the following: **I** Insurance Premium Tax District (IPD) **P** Property Tax District (PTD) **R** Payroll Tax District (PAY) **S** Special Purpose Tax District (SPD) For a description of the fields in each output group, see **Response** on page 185. **Note:** If you specify `W`, also specify `R` to obtain the best payroll system tax code match possible.
Option.TaxKey	If you integrate ReverseGeoTAXInfo Lookup with third-party tax compliance software from Vertex or Taxware, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following: **N** Do not return either the Taxware or Vertex jurisdiction codes (default). **T** Return the Taxware jurisdiction code for the address. **Y** Return the Vertex jurisdiction code for the address. Select this option if you obtained a MatchMaster file from Vertex. **V** Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Pitney Bowes.
Option.TaxRate	Select the desired Sales tax rate type or none: **N** Do not return sales tax rates. (default) **G** Return the General sales tax rates. **A** Return the Automotive sales tax rates. **C** Return the Construction sales tax rates. Return the Medical sales tax rates.

Parameter	Description
Option.OutputFields	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. For a list of the fields included in each data type, see **Response** on page 185.

Output Format

Output format options control how ReverseGeoTAXInfo Lookup formats output data.

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

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	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.txb file. For more information about buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
MCD.DistanceToBorder	10	Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb file.
StateCode	3	Two-digit Federal Information Processing Standards (FIPS) state 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.

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.
PTC*n*.MatchCode	2 per PTC	Indicates the level of match obtained for the location. 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*n*.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.
PTC*n*.PayrollDescription	41 per PTC	A description of the purpose of this payroll code.

Response Element	Max. Field Length with null terminator	Description
PTC*n*.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	The value in this field varies depending on the option you specified in the `Option.TaxKey` option: If you specified `T`, GeoTAXKey contains the proprietary codes used in Taxware tax compliance software. You can use this code in your Taxware application to find out the tax rate for the jurisdiction. If you specified `Y` or `V`, GeoTAXKey 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.

Response Element	Max. Field Length with null terminator	Description
GeoTAXKey.MatchCode	2	Return code denoting the level of match obtained against the Vertex or Taxware cross reference files. **E** Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name. **P** Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code. **A** Alternate match using two fields: ZIP Code, FIPS place name. **N** Record is default coded based on valid state code. **null** No matching record found.
GeoTAXKey.MatchLevel	12	A description of the value returned in the GeoTAXKey.MatchCode field. **Exact** Exact match. See description in GeoTAXKey.MatchCode. **Partial** Partial match. See description in GeoTAXKey.MatchCode. **Alternate** Alternate match. See description in GeoTAXKey.MatchCode. **DefaultCode** Record is default coded. See description in GeoTAXKey.MatchCode. **NoMatch** No matching record found.
GNISCode	10	Unique nine-digit Geographic Names Information System (GNIS) code. **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
Place.ClassCode	3	Place class code. Place class codes are used to determine the proper taxing jurisdictions **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output. **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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.

Response Element	Max. Field Length with null terminator	Description
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.LastVerifiedDate	8	Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.Name	41	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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.PointStatus	2	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 852. **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.
Place.DistanceToBorder	10	Returns the distance in feet between the matched address point to the polygon defined by the Place.txb file.

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*n*.BoundaryDescription	51 per User Boundary	A description of the polygon.
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	Indicates where in the polygon the location resides in relation to the edge of the area. Buffer width is specified by the option `Option.DefaultUserBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
UserBoundary*n*.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 179. To include IPD data in the output, set `Option.GeoTAXOutputRecordType` = `I`.

Note: ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, PTDs, 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	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
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
IPD*n*.DistrictID	11 per IPD	IPD ID.
IPD*n*.DistrictName	61 per IPD	IPD name.
IPD*n*.DistrictType	7 per IPD	IPD district type.
IPD*n*.UpdateDate	7 per IPD	IPD update date (MMYYYY).
IPD*n*.VersionDate	7 per IPD	IPD compiled date (MMYYYY).
IPD*n*.Notes	21 per IPD	Tax code descriptions. For example: 01, 33, A, B
IPD*n*.ChangeDate	7 per IPD	IPD change date.
IPD*n*.EffectiveDate	7 per IPD	MMDDYY - Identifies when district becomes active - State supplied For example: 010108
IPD*n*.ExpirationDate	7 per IPD	MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009
IPD*n*.FireRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.CasualtyRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.VehicleRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MarineRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.HealthRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.LifeRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.OtherRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MinimumRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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, PTDs, and PAYs.

Response Element	Max. Field Length with null terminator	Description
NumberPAYsFound	3	Number of payroll tax districts found for the location.
PAY*n*.BoundaryBuffer.BufferRelation	2 per PAY	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
PAY*n*.BoundaryBuffer.DistanceToBorder	10 per PAY	Indicates the distance in feet from the location to the border of the district.
PAY*n*.DistrictID	11 per PAY	PAY district ID.
PAY*n*.DistrictName	61 per PAY	PAY district name.
PAY*n*.DistrictType	7 per PAY	PAY district type.
PAY*n*.ID	11 per PAY	PAY ID.

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

Property Tax District

The following table lists the output fields that contain Property Tax District (PTD) data. For more information on property tax districts, see **ReverseGeoTAXInfoLookup** on page 179. To include this data in the output, set `Option.GeoTAXOutputRecordType` = `P`.

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

Response Element	Max. Field Length with null terminator	Description
NumberPTDsFound	3	The number of Property Tax Districts found for the location.
PTD*n*.BoundaryBuffer.BufferRelation	2 per PTD	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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 address is in the buffer area. For more information, see **Buffering** on page 852.
PTD*n*.BoundaryBuffer.DistanceToBorder	10 per PTD	Indicates the distance in feet from the location to the border of the district.
PTD*n*.DistrictID	11 per PTD	PTD district ID.
PTD*n*.DistrictName	61 per PTD	PTD district name.

Response Element	Max. Field Length with null terminator	Description
PTD*n*.DistrictType	2 per PTD	PTD district type. Only returned for Indiana. **R** Reporting district **B** Billing district
PTD*n*.JurisdictionID	11 per PTD	PTD jurisdiction ID.
PTD*n*.UpdateDate	7 per PTD	PTD update date.

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, PTDs, and PAYs.

Response Element	Max. Field Length with null terminator	Description
NumberSPDsFound	3	Number of Special Purpose Tax Districts found for the location.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.BoundaryBuffer.BufferRelation	2 per SPD	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `Option.DefaultBufferWidth` or by the input field `Data.BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
SPD*n*.BoundaryBuffer.DistanceToBorder	10 per SPD	Indicates the distance in feet from the address to the border of the district.
SPD*n*.CompiledDate	7 per SPD	SPD compiled date.
SPD*n*.DistrictCode	4 per SPD	3-digit district type code.
SPD*n*.DistrictName	61 per SPD	SPD name.
SPD*n*.DistrictNumber	6 per SPD	SPD district number.
SPD*n*.EffectiveDate	7 per SPD	SPD effective date.
SPD*n*.UpdateDate	7 per SPD	SPD update date.
SPD*n*.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 Pitney Bowes 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 Pitney Bowes 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.

Response Element	Max. Field Length with null terminator	Description
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).
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	This field contains a return code if the GeoTAX engine experiences an abnormal termination. **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. The first character indicates the file (or set of files affected). **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 The second position is one of the following: **E** Fatal issue, program terminating **F** Expired database **I** Informational

Response Element	Max. Field Length with null terminator	Description
GTX.ErrorDescription	81	

Response Element	Max. Field Length with null terminator	Description
		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. **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. 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" 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" 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 = "

Response Element	Max. Field Length with null terminator	Description
		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 `WN` 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.
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 215). 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.
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 211.

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

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

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

Extended 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 208).

> **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 210
> - 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 `AddressLine2`.
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.

Input Addressline	Output Addressline	Extended Code	Description
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.
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 AddressLine2.
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 AddressLine2.

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.

Code	In 3rd hex position means:
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 `AddressLine2`.
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 `AddressLine2`.
4	Unit Number changed or ignored.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.

Code	In 3rd hex position means:
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 `AddressLine2`.

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.

Code	"nnn" values	Description
	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.
	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 the Enterprise Tax Module 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 the Enterprise Tax Module did not standardize the address; therefore, the Enterprise Tax Module 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.

1st character	Always an "A" indicating an address location.		
2nd character	May be one of the following:		
		C	Interpolated address point location
		G	Auxiliary file data location
		I	Application infers the correct segment from the candidate records
		P	Point-level data location
		R	Location represents a ranged address

	S	Location on a street range
	X	Location on an intersection of two streets
3rd and 4th character	Digit indicating other qualities about the location.	

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:

Code	Description
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.
nn = 05	Structure centroid Indicates the center of a building footprint polygon, where the building receives mail or has telephone service. Usually a residential address consists of a single building. For houses with outbuildings (detached garages, shed, barns, etc.), only the residences have a structure point. Condominiums and duplexes have multiple points for each building. Larger buildings, such as apartment complexes, typically receive mail at one address for each building and therefore individual apartments are not represented as discrete structure points. Shopping malls, industrial complexes, and academic or medical center campuses where one building accepts mail for the entire complex are represented as one point. When addresses are assigned to multiple buildings within one complex, each addressed structure is represented by a point. If the center of a structure falls outside of its polygon, the center is manually repositioned to fall inside the polygon.
nn = 07	Manually placed Address points are manually placed to coincide with the midpoint of a parcel's street frontage at a distance from the center line.

Code	Description
nn = 08	Front door point Represents the designated primary entrance to a building. If a building has multiple entrances and there is no designated primary entrance or the primary entrance cannot readily be determined, the primary entrance is chosen based on proximity to the main access street and availability of parking.
nn = 09	Driveway offset point Represents a point located on the primary access road (most commonly a driveway) at a perpendicular distance of between 33-98 feet (10-30 meters) from the main roadway.
nn = 10	Street access point Represents the primary point of access from the street network. This address point type is located where the driveway or other access road intersects the main roadway.
nn = 21	Base parcel point When unable to match to an input unit number, or when the unit number is missing from an address location with multiple units, the "base" parcel information is returned, the address is not standardized to a unit number, and additional information, such as an Assessor's Parcel Number, is not returned.
nn = 22	Backfill address point The precise parcel centroid is unknown. The address location assigned is based on two known parcel centroids.
nn = 23	Virtual address point The precise parcel centroid is unknown. The address location assigned is relative to a known parcel centroid and a street segment end point.

Code	Description
nn = 24	Interpolated address point The precise parcel centroid is unknown. The address location assigned is based on street segment end points.
AIn	The correct segment is inferred from the candidate records at match time.
ASn	House range address geocode. This is the most accurate street interpolated geocode available.
AIn, ASn and ACn share the same values for the 3rd 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	Indicates one or both of the following: • The address is interpolated onto a TIGER segment that did not initially contain address ranges. • The original segment name changed to match the USPS spelling. This specifically refers to street type, predirectional, and postdirectional. **Note:** Only the second case is valid for non-TIGER data because segment range interpolation is only completed for TIGER data.
n = 3	Both 1 and 2.
n = 7	Placeholder. Used when starting and ending points of segments contain the same value and shape data is not available.

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

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

The table below contains the values and descriptions for the 2nd - 3rd characters in the street centroid location codes.

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

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

ZIP + 4 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.

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

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

Character Position	Code	Description
2nd Character		
	B	Block Group accuracy (most accurate).
	T	Census Tract accuracy.

Character Position	Code	Description
	C	Unclassified Census accuracy. Normally accurate to at least the County level.
3rd Character		
	5	Location of the Post Office that delivers mail to the address, a 5-digit ZIP Code centroid, or a location based upon locale (city). See the 4th character for a precise indication of locational accuracy.
	7	Location based upon a ZIP + 2 centroid. These locations can represent a multiple block area in urban locations, or a slightly larger area in rural settings.
	9	Location based upon a ZIP + 4 centroid. These are the most accurate centroids and normally place the location on the correct block face. For a small number of records, the location may be the middle of the entire street on which the ZIP + 4 falls. See the 4th character for a precise indication of locational accuracy.
4th Character		
	A	Address matched to a single segment. Location assigned in the middle of the matched street segment, offset to the proper side of the street.

Character Position	Code	Description
	a	Address matched to a single segment, but the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	B	Address matched to multiple segments, all segments have the same Block Group. Location assigned to the middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	b	Same as methodology B except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	C	Address matched to multiple segments, with all segments having the same Census Tract. Returns the Block Group representing the most households in this ZIP + 4. Location assigned to t he middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	c	Same as methodology C except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.

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

Character Position	Code	Description
	I	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, and b. All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.
	J	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, b, C, and c. All centroids in this ZIP + 2 cluster have the same Census Tract. Location assigned to the ZIP + 2 centroid.
	K	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, b, C, c, D, and d. Location assigned to the ZIP + 2 centroid.
	L	Created ZIP + 2 cluster centroid as defined by methodology E. All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.
	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.

Character Position	Code	Description
	V	Over 95% of addresses in this ZIP Code are in a single Census Tract. Location assigned to the ZIP Code centroid.
	W	Over 80% of addresses in this ZIP Code are in a single Census Tract. Reasonable Census Tract accuracy. Location assigned to the ZIP Code centroid.
	X	Less than 80% of addresses in this ZIP Code are in a single Census Tract. Census ID is uncertain. Location assigned to the ZIP Code centroid.
	Y	Rural or sparsely populated area. Census code is uncertain. Location based upon the USGS places file.
	Z	P.O. Box or General Delivery addresses. Census code is uncertain. Location based upon the Post Office location that delivers the mail to that address.

Geographic Centroid Location Codes

Location codes that begin with "G" are geographic centroid location codes. Geographic centroids may be returned if the 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.

1st character	Always "G" indicating a location derived from a geographic centroid.

2nd character	Geographic area type. One of the following: **M** Municipality (for example, a city) **C** County **S** State

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)**
- **Property Tax Districts (PTD)**

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

Type	Descriptions
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
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

Type	Descriptions
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
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 Distrit
DE	FIRE	Fire District
FL	FIRE	Fire District
FL	POLICE	Police District
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

State	Type	Descriptions
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
UNI	School District

Property Tax Districts (PTD)

Type	Descriptions
B	Billing
R	Reporting
X	Reporting and Billing

Class Codes

This appendix lists definitions for the FIPS Class Codes.

Class C—Incorporated Places

Class Code	Description
C1	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. 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.
C2	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. 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.
C3	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. 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.

Class Code	Description
C4	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. For example, the entity in California whose legal name is San Buenaventura (subclass C1) is commonly known as Ventura, which is coded C4.
C5	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.
C6	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.
C7	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.
C8	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.
C9	Identifies an inactive or nonfunctioning incorporated place.

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

Type	Descriptions
U1	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.
U2	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.
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.

Type	Descriptions
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.

GeoConfidence Module

GeoConfidenceSurface

GeoConfidenceSurface returns geoconfidence polygons (also called surfaces) based on the quality of the geocode information generated by the Enterprise Geocoding Module. 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 GeoConfidence Module's FloodZoneAnalysis dataflow template.

> **Note:** GeoConfidence uses services provided by the Enterprise Geocoding and Location Intelligence 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	The value returned in this field indicates which geoconfidence surface type has been returned. The possible values are: **INTERSECTION** A geocode point for the intersection of two streets. **ADDRESS** An array of street segment points representing the street segment where the address is located. **POINT** If the geocoder was able to match the address using point data, the point geometry where the address is located. **POSTAL1** A geocode point for the ZIP centroid. **POSTAL2** An array of points for all street segments in the ZIP + 2 in which the address is located. **POSTAL3** An array of points for street segments in the ZIP + 4 in which the address is located. **ERROR** An error has occurred.
StreetSegmentPoints	1024	An array of latitude/longitude values that represent the street segment points. **Note:** This field contains values only if the `GeoConfidenceCode` field returns a value of `ADDRESS`, `POSTAL2`, or `POSTAL3`.
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.

Global Sentry Module

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.pb.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 1: 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.
Data.PlaceOfBirth	Any place of birth data. Recommended if provided.

Parameter	Description
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 2: 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.
InputFilteredName	Input Name with titles, suffixes and special characters removed from the original data source.

Response Element	Description
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.
InputAddressLine2	Input Address line from the original data source.

Response Element	Description
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.
IDNumberProvided	Identifies if the ID Number is provided in the input data. Values are `Yes` or `No`.

Response Element	Description
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	Identifiesif 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	
InputCitizenship	Input Citizenship from the original data source.

Response Element	Description
Citizenship	Citizenship returned from database.
CitizenshipScore	Citizenship match score. 0 to 100.
CitizenshipMatchIdentified	Identifies if Citizenship is a match. Values are `Yes` or `No`.
CitizenshipProvided	Identifies if the Citizenship is provided in the input data. Values are `Yes` or `No`.
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 `Yes` or `No`.
NationalityProvided	Identifies if the Nationality is provided in the input data. Values are `Yes` or `No`.
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	

Response Element	Description
OverAllRiskLevel	Risk score per match. 0-16. For more information, see **Understanding the Risk Analysis Score** on page 250.
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 3: 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 the Spectrum™ Technology Platform, Universal Name, Data Normalization and Advanced Matching modules.

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

Information Extraction Module

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.pb.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 4: 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.).

Parameter	Description
Option.EntityList	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. **Address** **CreditCard** **Date** **Email** **HashTag** **ISBN** **Location** **Mention** **Organization** **Person** **Phone** **ProperNouns** **SSN** **WebAddress** **ZipCode**
Option.OutputEntityCount	Specifies whether to return a count of how many times a particular entity occurred in the output. **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.

Location Intelligence Module

Where to Find Documentation?

The Location Intelligence Module 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

To learn about the Location Intelligence Module services, see the *Spectrum Spatial Guide* on **support.pb.com**.

Universal Addressing Module

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.pb.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 5: 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.
Data.Country	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
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 6: 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	Specifies the casing of the output data. One of the following: **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

Parameter	Description
Option.HomeCountry	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: 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
Option.OutputCountryFormat	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". **E** Use English country names (default). **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
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.
Option.MaximumResults	The maximum number of addresses that AutoCompleteLoqate should return. The default is 10.
Option.isPowersearchEnable	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. **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. 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. Powersearch cannot be used for the following countries: Botswana, Ethiopia, India, Kazakhstan, Malaysia, Mongolia, Saint Kitts and Nevis, and San Marino. **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.

Parameter	Description
Option.IsDuplicateHandlingMaskEnable	Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options: **S** Selected by default. Pre-process the input and remove duplicates that occur in a single field. **C** Selected by default. Pre-process the input and remove duplicates across all fields. **T** Pre-process the input and remove duplicates in fields that are not standard address fields. **F** Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.
Option.FailJobOnDataLicenseError	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

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 7: 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 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
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 state or province abbreviation.
Status	Reports the success or failure of the match attempt. **null** Success **F** Failure
Status.Code	The reason for failure, if there is one. • DisabledCoder • RequestFailed • NoLookupAddressFound

Response Element	Description	
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. AutoCompleteLoqate returns data only if multiple possible matches were found.
	Not able to look up the address pattern	AutoCompleteLoqate is not able to process the partial address.

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 the Universal Addressing Module.

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.pb.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>
```

Request

Parameters for Input Data

The following table lists the input for GetCandidateAddresses.

Table 8: 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	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.
Data.Country	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name • French country name • German country name • Spanish country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
Data.FirmName	The company or firm name.
Data.USUrbanName	U.S. address urbanization name. Used primarily for Puerto Rico addresses.

Parameters for Options

Table 9: GetCandidateAddresses Options

Parameter	Description
Option.PerformUSProcessing	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. **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. **Y** Yes, process U.S. addresses (default). **N** No, do not process U.S. addresses.
Option.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.
Option.PerformCanadianProcessing	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. **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. **Y** Yes, process Canadian addresses (default). **N** No, do not process Canadian addresses.

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	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. **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. **Y** Yes, process international addresses (default). **N** No, do not process international addresses.
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	Specifies the casing of the output data. One of the following: **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
Option.MaximumResults	The maximum number of candidate addresses that GetCandidateAddresses should return. The default is 10. The maximum is 10.

Parameter	Description
Option.OutputShortCityName	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. **Y** Yes, return the short city name. **N** No, do not return the short city name.
Option.DualAddressLogic	(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 314. **N** (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority: **1.** PO Box **2.** Firm **3.** Highrise **4.** Street **5.** Rural Route **6.** General Delivery **S** Return a street match, regardless of the address line. **P** Return a PO Box match, regardless of the address line.
Option.StreetMatchingStrictness	The strictness of the street name match (U.S. addresses only). **E** The input street name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
Option.FirmMatchingStrictness	The strictness of the firm name match (U.S. addresses only). **E** The input firm name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."

Parameter	Description
Option.DirectionalMatchingStrictness	The strictness of the directional match. **E** The input directional must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
Option.PerformESM	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. **Y** Yes, perform ESM processing. **N** No, do not perform ESM processing (default).
Option.AddressLineSearchOnFail	Specifies whether ValidateAddress will search address lines for the city, state/province, and postal code. 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. Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields. 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). **Y** Yes, search the address line fields (default). **N** No, do not search the AddressLine fields.

Response

GetCandidateAddresses returns the following output.

Table 10: 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.
City	The city name.
Country	The three-character ISO 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
FirmName	The firm name.
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	Indicates the numbering scheme for the house numbers between HouseNumberLow and HouseNumberHigh, as follows: **E** Only even values **O** Only odd values **B** Both

Response Element	Description
MatchLevel	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: **A** The candidate matches the input address at the street level. **B** The candidate matches the input address at the state/province level.
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	The type of address record, as defined by U.S. and Canadian postal authorities (U.S. and Canadian addresses only): • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal
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 state or province abbreviation.
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: • DisabledCoder • RequestFailed

Response Element	Description
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.
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: **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 the Universal Addressing Module.

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.pb.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 11: 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.
Data.Country	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855. **Note:** This field is required. If this field is blank, no output will be returned.
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	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 12: 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.
Option.OutputCasing	Specifies the casing of the output data. One of the following: **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
Option.CandidateProcessOption	Specifies the method of searching for candidates. One of the following: **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
Option.HomeCountry	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: 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
Option.OutputCountryFormat	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". **E** Use English country names (default). **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
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.
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 13: 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 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.

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 state or province abbreviation.
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 the Universal Addressing Module.

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.pb.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: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 14: 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 15: GetCityStateProvince Options

Parameter Name	Description
Option.PerformUSProcessing	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. **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. **Y** Yes, process U.S. addresses (default). **N** No, do not process U.S. addresses.

Parameter Name	Description
Option.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.
Option.PerformCanadianProcessing	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. **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. **Y** Yes, process Canadian addresses (default). **N** No, do not process Canadian addresses.
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 16: 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 state or province abbreviation.
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: • DisabledCoder • UnrecognizedPostalCode

Response Element	Description	
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 Universal Addressing Module.

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.pb.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>
```

Request

Parameters for Input Data

The following table shows the input fields.

Table 17: GetCityStateProvinceLoqate Input

Parameter	Description
Data.Country	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
Data.PostalCode	The postal code for the address.

Options

Table 18: 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.
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 19: GetCityStateProvinceLoqate Output

Response Element	Description
City	The matched city name.
Country	The country in the format determined by what you selected in: • ISO Code • UPU Code • English
PostalCode	The input postal code.
ProcessedBy	Indicates which address coder processed the address. **LOQATE** The Loqate coder processed the address.
StateProvince	The state or province abbreviation.
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: • UnrecognizedPostalCode
Status.Description	The description of the failure. The only valid value is: **Postal code not found** This value will appear if Status.Code=UnrecognizedPostalCode.

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 Universal Addressing Module.

Resource URL

JSON endpoint:

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

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.pb.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 20: GetPostalCodes Input

Parameter	Description
Data.City	The city whose postal codes you want to look up. You may put the city and state in the City field. If you do this, you must leave the StateProvince field blank. The total length of the City and StateProvince fields cannot exceed 100 characters.
Data.StateProvince	The state or province of the city whose postal codes you want to look up. You may also put the state in the City field instead of the StateProvince field. The total length of the City and StateProvince fields cannot exceed 100 characters.
Data.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 21: GetPostalCodes Options

Parameter	Description
Option.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.
Option.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).
Option.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 22: GetPostalCodes Output

Response Element	Description
City.Type	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. This field column is only populated if `Option.OutputCityType=Y`. The possible values are: **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	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	Reports the success or failure of the match attempt. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. One of the following: • CountryNotSupported • UnableToLookup

Response Element	Description
Status.Description	Description of failure. • 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. ValidateAddress 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 ValidateAddress 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, ValidateAddress 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.pb.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 301 for important information about how address line data is processed for U.S. addresses.

Table 23: 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.AddressLine5	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.
Data.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.
Data.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.

Parameter	Format	Description
Data.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).
Data.Country	String [50]	The country code or name, in any of the following formats: • Two-character ISO 3116-1 Alpha-2 country code • Three-character ISO 3116-1 Alpha-3 country code • English country name • French country name • German country name • Spanish country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
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.
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 `Option.CanFrenchFormat=T` 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: Pitney Bowes

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 329.

Table 24: Output Data Options

Parameter	Description
Option.OutputRecordType	Type of output record. For more than one, provide a list. **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 **Response** on page 340. 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 `OutputRecordType=A`. **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 342. Note that if you specify "E" and specify `OutputFormattedOnFail=Y`, the parsed address elements will contain the input address for addresses that could not be validated. **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 `OutputRecordType=E` and `OutputFormattedOnFail=Y` 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 345. **P** Postal data. Output addresses contain additional data for each validated address. For more information, see **Postal Data Output** on page 347. **Blank** Do not return any address data or postal data.

Parameter	Description
Option.OutputFieldLevelReturnCodes	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 354. **N** No, do not output field-level return codes (default). **Y** Yes, output field-level return codes.

Parameter	Description
Option.OutputFormattedOnFail	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. **Note:** This option applies only to U.S. and Canadian addresses. Formatted data will not be returned for any other address. **N** No, do not format failed addresses (default). **Y** Yes, format failed addresses. 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 ValidateAddress could validate the address, specify `OutputRecordType=I`. 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 ValidateAddress could validate the address, specify `Option.OutputRecordType=I`. 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 ValidateAddress could validate the address, select **Include standardized input address elements**. If you specify Y, you must specify "A" and/or "E" for OutputRecordType. If you specify Y, you must specify "A" and/or "E" for `Option.OutputRecordType`. If you check this option, you must select **Include a standard address** and/or **Include address line elements**.

Parameter	Description
Option.OutputStreetNameAlias	For U.S. addresses only, specifies whether or not to use a street's alias in the output. A street alias is an alternate name for a street and 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. **N** No, do not return street name aliases in the output. **Y** Yes, return street name aliases in the output if there is an alias for the street (default).
Option.OutputStreetNameAlias	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 typically applies only to a specific range of addresses on the street. 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: • If a preferred alias is used in input the preferred alias will always be used in output. • Changed aliases used in input are always converted to the base street name in output. This is one of three options that control how ValidateAddress handles street name aliases. The other two are `Option.OutputPreferredAlias` and `Option.OutputAbbreviatedAlias`. **Note:** If `Option.OutputAbbreviatedAlias` is enabled, the abbreviated alias will always appear in the output even if you have `Option.OutputStreetNameAlias` disabled. **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).

Parameter	Description
Option.OutputAddressBlocks	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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 `Option.OutputCountryFormat` does not affect the country name in the address block, it only affects the name returned in the **Country** output field. 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. **N** No, do not return address blocks. Default. **Y** Yes, return address blocks.

Parameter	Description
Option.OutputAMAS	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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 `Option.OutputCountryFormat` does not affect the country name in the address block, it only affects the name returned in the **Country** output field. 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. **N** No, do not return address blocks. Default. **Y** Yes, return address blocks.

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 301.

Table 25: 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 301.

Table 26: 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 301.

Table 27: 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 301.

Table 28: 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 301.

Table 29: 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 the apply to all addresses. Some of these options can be overridden for Canadian addresses. For more information, see **Canadian Address Options** on page 329.

Table 30: Default Options

Parameter	Description
Option.OutputCasing	Specifies the casing of the output address. One of the following: **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
Option.OutputPostalCodeSeparator	Specifies whether to use separators (spaces or hyphens) in ZIP™ Codes or Canadian postal codes. 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. **Y** Yes, use separator (default). **N** No, do not use separator. **Note:** Spaces are used in Canadian postal codes and hyphens in U.S. ZIP + 4® Codes.
Option.OutputMultinationalCharacters	Specifies whether or not to return multinational characters, including diacritical marks such as umlauts or accents. (Not supported for U.S. addresses). **N** No, do not use multinational characters in the output (default). Only standard ASCII characters is returned. **Y** Yes, use multinational characters in the output.

Parameter	Description
Option.KeepMultimatch	Indicates whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches. For more information, see **Returning Multiple Matches** on page 315.
Option.StandardAddressFormat	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". Apt 10E 424 Washington Blvd Springfield MI 49423 **C** Place both primary and secondary address information in AddressLine1 (default). **S** Place the primary address information in AddressLine1 and the secondary address information in AddressLine2. **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 314.

Parameter	Description
Option.OutputShortCityName	Specifies how to format city names that have short city name or non-mailing city name alternatives. Applies to U.S. and Canadian addresses. **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.
Option.OutputCountryFormat	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". **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
Option.HomeCountry	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: 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

Parameter	Description
Option.DualAddressLogic	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.) **N** (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority: **1.** PO Box **2.** Firm **3.** Highrise **4.** Street **5.** Rural Route **6.** General Delivery **S** Return a street match, regardless of the address line. **P** Return a PO Box match, regardless of the address line. For more information, see **About Dual Address Logic** on page 314.

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 310. 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 310. 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 310 .

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 31: Multiple Match Option

Parameter	Description
Option.KeepMultimatch	Indicates whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches.
Option.MaximumResults	A number between 1 and 10 that indicates the maximum number of addresses to return. The default value is 1. **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.
Option.OutputFieldLevelReturnCodes	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.

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 301. 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 354.

U.S. Address Options

Parameter	Description
Option.PerformUSProcessing	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. **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. **N** No, do not process U.S. addresses. **Y** Yes, process U.S. addresses. Default.
Option.Database.US	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.
Option.PerformLOT	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. To perform eLOT processing you must have the eLOT database installed. **N** No, do not perform Line of Travel Processing. Default. **Y** Yes, perform Line of Travel processing. For a listing of the output fields returned by this option, see **Enhanced Line of Travel Output** on page 368.
Option.PerformRDI	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. 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. **N** No, do not perform Residential Delivery Indicator processing. Default. **Y** Yes, perform Residential Delivery Indicator processing.

Parameter	Description
Option.PerformESM	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. **N** No, do not perform enhanced street matching. Default. **Y** Yes, perform enhanced street matching.
Option.PerformASM	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. **N** No, do not perform all street matching. **Y** Yes, perform all street matching. Default.
Option.PerformDPV	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). To perform DPV and CMRA processing, you must have the DPV database installed. The DPV database contains both DPV and CMRA data. **N** No, do not perform Delivery Point Validation or CMRA processing. Default. **Y** Yes, perform Delivery Point Validation and CMRA processing. For a listing of the output fields returned by this option, see **DPV and CMRA Output** on page 371.
Option.PerformLACSLink	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. To perform LACS^{Link} processing, you must have the LACS^{Link} database installed. **N** No, do not attempt LACS^{Link} conversion. Default. **Y** Yes, attempt LACS^{Link} conversion. For a listing of the output fields returned by this option, see **LACSLink Output** on page 369

Parameter	Description
Option.PerformEWS	The Early Warning System (EWS) uses the USPS® EWS File to validate addresses that are not in the ZIP + 4® database. To perform EWS processing, you must have the EWS database installed. If an input address matches an address in the EWS file, the following record-level result indicators are returned: • Status="F" • Status.Code="EWSFailure" • Status.Description="Address found in EWS table" **N** No, do not perform EWS processing. Default. **Y** Yes, perform EWS processing.

Parameter	Description
Option.ExtractFirm	

Parameter	Description

Specifies whether to extract the firm name from AddressLine1 through AddressLine4 and place it in the FirmName output field. This option works in cases where the input record's FirmName field is blank and there is more than one address line.

Y	Yes, extract the firm name.
N	No, do not extract the firm name. Default.

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:

- 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,

  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes
  AddressLine4: <blank>

- 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.

Here are some examples of firm name extraction:

- In this example, AddressLine2 would get extracted into the FirmName output field

  FirmName: <blank>
  AddressLine1: 4200 Parliament Place Suite 600
  AddressLine2: International Goose Feathers inc.

- In this example, AddressLine3 would get extracted into the FirmName output field.

  FirmName: <blank>
  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes

- 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.

  FirmName: International Goose Feathers Inc.
  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes

- In this example, no firm name would be extracted because there is only one

Parameter	Description
	non-blank address line, which is always treated as the primary address element. **FirmName:** <blank> **AddressLine1:** 4200 Parliament Place Suite 600 • 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. **FirmName:** <blank> **AddressLine1:** 4200 Parliament Place Suite 600 **AddressLine2:** 1 Stop Software
Option.ExtractUrb	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. **Y** Yes, extract the urbanization name. **N** No, do not extract the urbanization name. Default. 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, **AddressLine1:** A1 Calle A **AddressLine2:** **AddressLine3:** URB Alamar **AddressLine4:** Pitney Bowes

Parameter	Description
Option.PerformSuiteLink	Specifies whether to perform SuiteLink™ processing. SuiteLink corrects secondary address information for U.S. business addresses whose secondary address information could not be validated. If SuiteLink processing is enabled, the firm name is matched to a database of known firm names and their secondary address information. For example, Firm Name: Pitney Bowes Address Line 1: 4200 Parliament Place Address Line 2: STE 1 Postal Code: 20706 In this case, SuiteLink processing would change the suite number to the correct suite number: Firm Name: Pitney Bowes Address Line 1: 4200 Parliament Place Address Line 2: **STE 600** Postal Code: 20706-1844 To perform SuiteLink™ processing, you must have the SuiteLink™ database installed. This option takes one of the following values: **N** No, do not use SuiteLink™. Default. **Y** Yes, use SuiteLink™ processing. For a listing of fields returned by this option, see **SuiteLink Output** on page 373.

Parameter	Description
Option.OutputPreferredAlias	Specifies whether to use a street's preferred alias in the output. Street name aliases in the United States are alternative names given to sections of a street. There are four types of street name aliases: • **Preferred**—A preferred alias is the street name preferred locally. It typically applies only to a specific range of addresses on the street. • **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. • **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. The non-alias version of the street name is called the base street name. 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. This is one of three options that control how ValidateAddress handles street name aliases. The other two are Option.OutputStreetNameAlias and Option.OutputAbbreviatedAlias. 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. **Y** Yes, perform preferred alias processing. **N** No, do not perform preferred alias processing. Default. **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.

Parameter	Description
Option.OutputAbbreviatedAlias	Specifies whether to use a street's abbreviated alias in the output if the output address line is longer than 31 characters. This is one of three options that control how ValidateAddress handles street name aliases. The other two are Option.OutputStreetNameAlias and Option.OutputPreferredAlias. **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. **Y** Yes, perform abbreviated alias processing. **N** No, do not perform abbreviated alias processing. Default. **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.
Option.DPVDetermineNoStat	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. **N** No, do not determine "no stat" status. Default. **Y** Yes, determine "no stat" status. **Note:** You must enable DPV processing to use this option. The result is returned in the DPVNoStat field. For more information see **LACSLink Output** on page 369
Option.DPVDetermineVacancy	Determines if the location has been unoccupied for at least 90 days. **N** No, do not determine vacancy. Default. **Y** Yes, determine vacancy. **Note:** You must enable DPV processing to use this option. The result is returned in the DPVVacant field. For more information see **LACSLink Output** on page 369

Parameter	Description
Option.ReturnVerimove	Returns VeriMove detail data in output. **N** No, do not return VeriMove detail data. Default. **Y** Yes, return VeriMove detail data.
Option.SuppressZplusPhantomCarrierR777	Specifies whether to supress 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: • No ZIP + 4 code is assigned • Address is not counted on the USPS Form 3553 (CASS Summary Report) • DPV Footnote of R7 is returned **N** No, do not suppress addresses with Carrier Route R777. **Y** Yes, suppress addresses with Carrier Route R777.
Option.StreetMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input street name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
Option.FirmMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input firm name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."

Parameter	Description
Option.DirectionalMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input directionals, such as the "N" in 123 N Main St., must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium". Default. **L** The matching algorithm is "loose."
Option.DPVSuccessfulStatusCondition	Select the match condition where a DPV result does NOT cause a record to fail. **F** Full match **P** Partial match **A** Always. Default. **Note:** You must enable DPV processing to use this option.
Option.FailOnCMRAMatch	Treat Commercial Mail Receiving Agency (CMRA) matches as failures? **N** No, do not treat CMRA matches as failures. Default. **Y** Yes, treat CMRA matches as failures. **Note:** You must enable DPV processing to use this option.
Option.StandardAddressPMBLine	Specifies where Private Mailbox (PMB) information is placed. **N** Do not include the PMB information in Standard Address output (default). **1** Place the PMB information in AddressLine1. If you specify 1, you must set Option.StandardAddressFormat to either C or D. **2** Place the PMB information in AddressLine2.

Parameter	Description
Option.PreferredCity	Specifies whether the preferred last line city name should be stored. **Z** Store the Preferred Last Line City Name from the USPS ZIP+4 File (Override City Name). **Note:** If you select this option, Validate Address generates a CASS-certified configuration and the USPS 3553 Report. **C** Store the USPS-preferred City Name from USPS City/State File. **Note:** If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report. **P** Store the Primary City Name from the USPS City/State File. **Note:** If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.

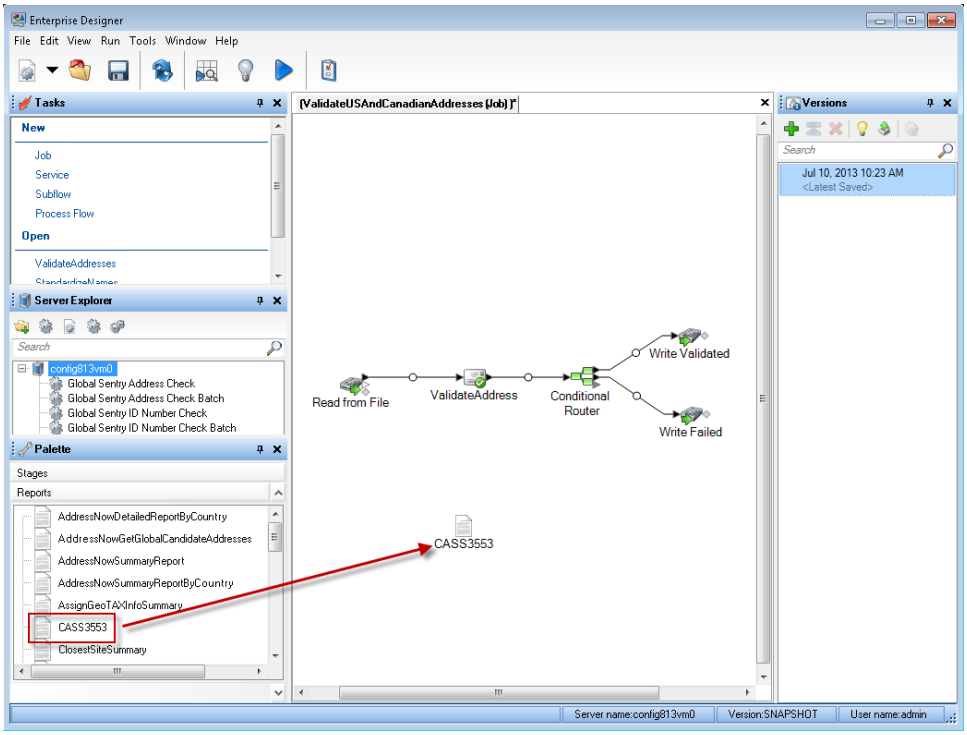
CASS Certified Processing

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 the **CASS3553** report 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** checkbox. 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	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. **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. **N** No, do not process Canadian addresses. **Y** Yes, process Canadian addresses (default).
Option.Database.Canada	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.

Parameter	Description
Option.CanFrenchFormat	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: English: 123 Main St W French: 123 Rue Main O The parameter controls the formatting of the address. It also affects the spelling of the directional but not spelling of the suffix. **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.CanFrenchFormat	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 `Option.CanStandardAddressFormat=F`. **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	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 `Option.CanStandardAddressFormat=F`. **App** Use "App" as the label. Default. **Appartement** Use "Appartement" as the label. **Bureau** Use "Bureau" as the label. **Suite** Use "Suite" as the label. **Unite** Use "Unite" as the label.
Option.ForceCorrectionLVR	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). **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.. **Y** Change the civic and/or suite information to match the LVR or single-single record. **Note:** If you check this box, the Statement of Address Accuracy will not be printed because this is **not** a SERP-recognized setting.
Option.CanPreferHouseNum	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 `Option.CanPreferHouseNum=Y`. If you do not select this option the house number is changed to match the postal code. **N** Change the house number to match the postal code. Default. **Y** Change the postal code to match the house number.

Parameter	Description
Option.CanOutputCityAlias	Specifies whether or not to return the city alias when the alias is in the input address. This option is disabled when you specify `Option.CanOutputCityFormat=D`. **Y** Output the city alias when the city alias is in the input. Default. **N** Never output the city alias even if it is in the input.
Option.CanNonCivicFormat	Specifies whether or not non-civic keywords are abbreviated in the output. For example, Post Office Box vs. PO Box. **A** Abbreviate non-civic keywords. Default. **F** Do not abbreviate non-civic keywords. The full keyword is used.
Option.EnableSERP	Specifies whether or not to use SERP options. **Y** Enable SERP options. **N** Do not enable SERP options. Default.
Option.CanStandardAddressFormat	Specifies where to place secondary address information in the output address. Secondary address information refers to apartment numbers, suite numbers, and similar designators. **D** Place apartment information in the location specified in the. Default. **B** Place apartment information at the at the end of the AddressLine1 field. **F** Place the apartment number only (no label) at the beginning of the AddressLine1 field. For example, 400-123 Rue Main **E** Place the apartment number and label at the beginning of the AddressLine1 field. For example, Apt 400 123 Rue Main **S** Place apartment information on a separate line. **S** Place apartment information in the same location as the input address.

Parameter	Description
Option.CanOutputCityFormat	Specifies whether to use the long, medium, or short version of the city if the city has a long name. For example, Long: BUFFALO HEAD PRAIRIE Medium: BUFFALO-HEAD-PR Short: BUFFALO-HD-PR **D** Use the default option specified by the `Option.OutputShortCityName` parameter. Default. If you specify `Option.OutputShortCityName=V`, the city is formatted as if you select for this option (see below) and`Y` for **CanOutputCityAlias**. **S** Output short city name. **L** Output the long city name. **M** Output the medium city name. **I** Use the same city format as used in the input address. Output is L, M, or S.
Option.CanRuralRouteFormat	Specifies where to place rural route delivery information. An example of an address with rural route delivery information is: 36 GRANT RD RR 3 ANTIGONISH NS In this address, "RR 3" is the rural route delivery information. **A** Place rural route delivery information on the same line as the address, after the address information. Default. For example, 36 GRANT RD RR 3 **S** Place rural route delivery information on a separate address line. For example, 36 GRANT RD RR 3

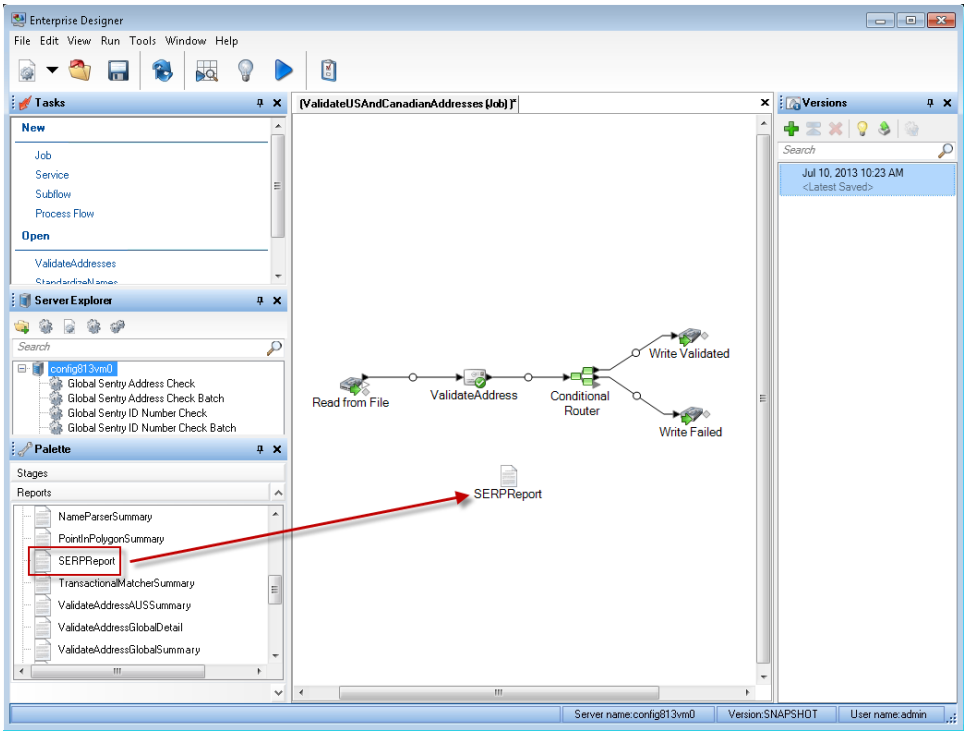
Parameter	Description
Option.CanDeliveryOfficeFormat	Specifies where to place station information. An example of an address with station information is: PO BOX 8625 STN A ST. JOHN'S NL **I** Place station information in the same location as it is in the input address. Default. **A** Place station information on the same line as the address, after the address information. For example, PO BOX 8625 STN A **S** Place station information on a separate address line. For example, PO BOX 8625 STN A

Parameter	Description
Option.CanDualAddressLogic	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: **D** Use DualAddressLogic Global Option. Default. **P** Match to PO Box or other non-street data. **S** Match to street. For example, given the following input address: AddressLine1: 36 GRANT RD AddressLine2: RR 4 City: ANTIGONISH StateProvince: NS ValidateAddress would return one of the following: • If `Option.CanDualAddressLogic` is set to S, ValidateAddress returns the following: AddressLine1: 36 GRANT RD AddressLine2: RR 3 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L1 • If `Option.CanDualAddressLogic` is set to P, ValidateAddress returns the following: AddressLine1: RR 4 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L2 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 301.

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 the 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** checkbox 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 301.

SERP return codes are provided in the following output field.

Table 32: SERP Return Code Output

Response Element	Description
CanadianSERPCode	Validation/correction return code (Canadian addresses only): **V** The input was valid. Canada Post defines a "valid" address as an address that meets all the following requirements: **Note:** There are exceptions. For further information, contact the CPC. • 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. **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. **C** The input was correctable. A "correctable" address is one that can be corrected to match one, and only one, address. **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. **F** The input address was foreign (outside of Canada).

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	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. **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. **N** No, do not process international addresses. **Y** Yes, process international addresses (default).
Option.Database.International	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.

Parameter	Description
Option.InternationalCityStreetSearching	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. 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. 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.
Option.AddressLineSearchOnFail	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. Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields. 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). **N** No, do not search the AddressLine fields. **Y** Yes, search the address line fields. Default.

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 33: Standard Address Output

Response Element	Description
AdditionalInputData	Input data not used by the address validation process. For more information, see **Additional Input Data** on page 374.
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 Data.OutputCountryFormat: • 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 or province abbreviation.
USUrbanName	The validated urbanization name. (U.S. addresses only.) This is used primarily for Puerto Rico addresses.

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 345.

Table 34: Parsed Address Output

Response Element	Description
AdditionalInputData	Input data not used by ValidateAddress. For more information, see **Additional Input Data** on page 374.
AdditionalInputData.Base	Input data that was not output to the standardized address by ValidateAddress. For more information, see **Additional Input Data** on page 374.
AdditionalInputData.Unmatched	Input data passed to the matcher but not used by ValidateAddress for validation. For more information, see **Additional Input Data** on page 374.
ApartmentLabel	Apartment designator (such as STE or APT), for example: 123 E Main St **APT** 3
ApartmentLabel2	Secondary apartment designator, for example: 123 E Main St APT 3, 4th **Floor** **Note:** In this release, this field will always be blank.
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.
CanadianDeliveryInstallationAreaName	Delivery installation name (Canadian addresses only)
CanadianDeliveryInstallationQualifierName	Delivery installation qualifier (Canadian addresses only)
CanadianDeliveryInstallationType	Delivery installation type (Canadian addresses only)

Response Element	Description
City	Validated city name
Country	Country. Format is determined by what you selected in `Data.OutputCountryFormat`: • 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.
PrivateMailbox	Private mailbox indicator.

Response Element	Description
PrivateMailbox.Type	The type of private mailbox. Possible values include: • Standard • Non-Standard **Note:** This replaces PrivateMailboxType (no period in field name). Please modify your API calls accordingly.
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 35: 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 Data.OutputCountryFormat: • 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: • 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 36: Postal Data Output

Response Element	Description
CanadianSERPCode	Validation/correction return code (Canadian addresses only). For more information, see **Obtaining SERP Return Codes** on page 337.

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 309.
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: **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 309.
USCarrierRouteCode	Carrier route code (U.S. addresses only). For more information, see **Obtaining Carrier Route Codes** on page 308.
USCongressionalDistrict	Congressional district (U.S. addresses only). For more information, see **Obtaining Congressional Districts** on page 307.
USCountyName	County name (U.S. addresses only). For more information, see **Obtaining County Names** on page 308.
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 308.
USLACS	Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following: **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. **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: • 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	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: Chantilly VA 20151 Chantilly VA 20152

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 37: Record Level Indicators

Response Element	Description
AddressFormat	The type of address data being returned: **F** French format (for example: 123 Rue Main) **E** English format (for example: 123 Main St)
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 887.
CouldNotValidate	If no match was found, which address component could not be validated: • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber • RuralRoute **Note:** More than one component may be returned, in a comma-separated list.

Response Element	Description
CountryLevel	The category of address matching available. This is always "A" for U.S. and Canadian addresses. One of the following: **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: • Postal code • City name • State/county name • Street address elements • Country name **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: • Postal code • City name • State/county name • Country name **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: • 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	MatchScore provides an indication of the degree to which the output address is correct. It is significantly different from Confidence in that Confidence is 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. 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. 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. This field does not apply to Canadian addresses. 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. **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.
MultimatchCount	If multiple matches were found, indicates the number of records that are possible matches.
MultipleMatches	Indicates which address component had multiple matches, if multiple matches were found: • Firm • LeadingDirectional • PostalCode • StreetName • StreetSuffix • TrailingDirectional • Urbanization **Note:** More than one component may be returned, in a comma-separated list.

Response Element	Description
ProcessedBy	Which address coder processed the address: **USA** U.S. address coder **CAN** Canadian address coder **INT** International address coder
RecordType	Type of address record, as defined by U.S. and Canadian postal authorities (supported for U.S. and Canadian addresses only): • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal
RecordType.Default	Code indicating the "default" match: **Y** The address matches a default record. **null** The address does not match a default record.
Status	Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. For multiple matches, all possible matches is "MultipleMatchesFound." • DisabledCoder • InsufficientInputData • MultipleMatchesFound • UnableToValidate

Response Element	Description
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 301.

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 38: Field-Level Result Indicators

Response Element	Description
AddressRecord.Result	These result codes apply to international 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. **S** — Standardized. This option includes any standard abbreviations. **U** — Unmatched. **V** — Validated. The data was confirmed correct and remained unchanged from input.

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. **D** Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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. **U** Unmatched. Does not apply to Canadian 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.

Response Element	Description
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.
HouseNumber.Result	**A** Appended. The field was added to a blank input field. 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 **Additional Input Data** on page 374. **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. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
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. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
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. **U** Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U. **V** Validated. The data was confirmed correct and remained unchanged from input.
PostalCodeCity.Result	These result codes apply to international 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. **P** Pass-through. The data was not used in the validation process, but it was preserved in the output. **S** Standardized. This option includes any standard abbreviations. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
PostalCode.Source	These result codes apply to U.S. addresses only. **FinanceNumber** The ZIP Code™ in the input was verified by using USPS® Finance Number groupings. **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™.
PostalCode.Type	**P** The ZIP Code™ contains only PO Box addresses. U.S. addresses only. **U** The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only. **M** The ZIP Code™ is for military addresses. U.S. addresses only. **null** The ZIP Code™ is a standard ZIP Code™.
RRHC.Result	**C** Corrected. Canadian addresses only. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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 rural route/highway contract is required but is missing from the input address. U.S. addresses only. **S** Standardized. This option includes any standard abbreviations. 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. and Canadian addresses only.

Response Element	Description
RRHC.Type	These result codes apply to U.S. addresses only. **HC** The address is a Highway Contract address. **RR** The address is a Rural Route address.
StateProvince.Result	**A** Appended. The field was added to a blank input field. U.S. and Canadian addresses only. **C** Corrected. U.S. 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 state 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. Does not apply to Canadian addresses. **V** Validated. The data was confirmed correct and remained unchanged from input.
Street.Result	These result codes apply to international 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. **P** Pass-through. The data was not used in the validation process, but it was preserved in the output. **R** Street corrected. House number is out of range. Applies to French, UK, and Japanese records only. **S** Standardized. This option includes any standard abbreviations. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
StreetName.AbbreviatedAlias.Result	Indicates the result of abbreviated alias processing. One of the following: **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	This result code applies to U.S. addresses only. **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. **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	Indicates the result of preferred alias processing. One of the following: **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	**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 **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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.
USUrbanName.Result	These result codes apply to U.S. addresses only. **A** Appended. The field was added to a blank input field. **C** Corrected. **M** Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

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	Line of Travel sequence code and an indicator denoting USPS® LOT sequence. This field is in the format nnnnY where: **nnnn** The four-digit LOT code. **Y** One of the following: • **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.
USLOTSequence	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	Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following: **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. **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: • 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	Indicates the success or failure of LACS^{Link} processing. (U.S. addresses only.) **A** LACS^{Link} processing successful. Record matched through LACS^{Link} processing. **00** LACS^{Link} processing failed. No matching record found during LACS^{Link} processing. **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. **14** LACS^{Link} processing failed. Match found during LACS^{Link} processing but conversion did not occur due to other USPS® regulations. **92** LACS^{Link} processing successful. Record matched through LACS^{Link} processing. Unit number dropped on input. **null** LACS^{Link} did not process the record, or LACS^{Link} processing was not attempted.

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	Indicates the results of Delivery Point Validation (DPV) processing. **Y** DPV confirmed. Mail can be delivered to the address. **N** Mail cannot be delivered to the address. **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: 424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA **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: 424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA **M** The address matches multiple valid delivery points. **U** The address could not be confirmed because the address did not code at the ZIP + 4® level. **V** The address caused a false-positive violation.
CMRA	Indicates if the address is a Commercial Mail Receiving Agency (CMRA) **Y** Yes, the address is a CMRA. **N** No, the address is not a CMRA. **U** Unconfirmed.

Response Element	Description
DPVFootnote	DPV footnote codes. **AA** Input address matched to the ZIP + 4® file. **A1** Input address not matched to the ZIP + 4® file. **BB** Input address matched to DPV (all components). **CC** Input address primary number matched to DPV but secondary number not match (present but not valid). **F1** Input address is military; DPV bypassed. **G1** Input address is general delivery; DPV bypassed. **M1** Input address primary number missing. **M3** Input address primary number invalid. **N1** Input address primary number matched to DPV but high rise address missing secondary number. **P1** Input address missing RR or HC Box number. **P3** Input address missing PO, RR, or HC Box number **RR** Input address matched to CMRA. **R1** Input address matched to CMRA but secondary number not present. **U1** Input address is unique ZIP; DPV bypassed.
DPVVacant	Indicates whether the building is vacant (unoccupied for 90 days). One of the following: **Y** Yes, the building is vacant. **N** No, the building is not vacant. **null** The `Option.DPVDetermineVacancy` option was not turned on.
DPVNoStat	Indicates whether the building is a "no stat" building and therefore unable to receive mail. One of the following: **Y** Yes, the building is a "no stat" building, which means the building is not receiving mail. **N** No, the building is not a "no stat" building, which means the building does receive mail. **null** The option was not turned on.

Suite[Link] Output

Response Element	Description
SuiteLinkReturnCode	Indicates whether or not ValidateAddress corrected the secondary address information (U.S. addresses only). One of the following: **A** ValidateAddress corrected the secondary address information. **00** ValidateAddress did not correct the secondary address information. **null** Suite[Link] was not performed. **XX** Suite[Link] processing encountered an error. For example, an error would occur if the Suite[Link] database is expired.
SuiteLinkMatchCode	Provides additional information on the Suite[Link] match attempt. (U.S. addresses only) **A** ValidateAddress corrected the secondary address information. **B** ValidateAddress did not correct the secondary address information. No additional detail about the match attempt is available. **C** The words in the FirmName field are all "noise" words. Noise words are defined by the USPS® and are ignored when attempting to mach 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. **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). **E** Suite[Link] processing failed because the Suite[Link] database is expired. **null** Suite[Link] was not performed or there was an error.

Response Element	Description
SuiteLinkFidelity	Indicates how well ValidateAddress matched the firm name to the firm names in the SuiteLink database. **1** The firm name matches the SuiteLink database exactly. **2** Good match. All words in the firm name except one matched the firm name in the SuiteLink database. **3** Poor match. More than one word in the firm name did not match the firm name in the SuiteLink database. **null** SuiteLink could not match the firm name, or was not performed, or there was an error.

VeriMove Output

Response Element	Description
VeriMoveDataBlock	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: **Y** Yes, return the field VeriMoveDataBlock.. **N** No, do not return the field VeriMoveDataBlock.

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 usuall 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 301 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:

Pitney Bowes
4200 Parlament
Lanham MD

ValidateAddress would correct the spelling of the street name and add the suffix, returning "4200 Parliament Pl" 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 314.

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": "006000506000000000060",
   "ElementRelevance": "111010101000000000010",
   "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+View&Data.City=Troy&
Data.Country=USA&Data.StateProvince=NY
```

The XML returned by this request would be:

```
<ns2:xml.ValidateAddressGlobalResponse
xmlns:ns2="http://www.pb.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 39: ValidateAddressGlobal Input

Parameter	Format	Description
Data.AddressLine1 through Data.AddressLine6	String [79]	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: **AddressLine1:** 17413 Blodgett Road **AddressLine2:** PO Box 123 **City:** Mount Vernon **StateProvice:** WA **PostalCode:** 97273 **Country:** USA If the input address is not already parsed into the appropriate address line and City, StateProvice, and PostalCode fields, use the UnformattedLine fields instead of the address line fields.
Data.City	String [79]	City name
Data.StateProvince	String [79]	State or province.
Data.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®.
Data.Contact	String [79]	The name of the addressee. For example, "Mr. Jones".
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

Parameter	Format	Description
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]	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: **UnformattedLine1:** 17413 Blodgett Road **UnformattedLine2:** PO Box 123 **UnformattedLine3:** Mount Vernon WA 97273 **UnformattedLine4:** USA This address would be parsed into these output fields: **AddressLine1:** 17413 Blodgett Road **AddressLine2:** PO Box 123 **City:** Mount Vernon **StateProvice:** WA **PostalCode:** 97273 **Country:** USA **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.

Parameters for Options

Input Options

Table 40: 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 *Spectrum™ Technology Platform Administration Guide*.
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 855.
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 855.
Option.Input.FormatDelimiter	Enables you to use non-standard formatting for multi-line addresses in input files. Acceptable values for this field include the following: • 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) **Note:** The same value must be selected for both the input option and output option.

Output Options

Table 41: 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	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. **TRUE** Include the input data in the output. **FALSE** Do not include the input data in the output (default).
Option.Result.StateProvinceType	Specifies the format for the StateProvince field. One of the following. **ABBREVIATION** Return the abbreviation for the state or province. For example, North Carolina would be returned as "NC". **COUNTRY_STANDARD** Return either the abbreviation or the full name depending on the format used by the country's postal authority. (Default) **EXTENDED** Return the full name of the state or province, not the abbreviation. For example "North Carolina".

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	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. **ASCII_Extended** — ASCII characters with expansion of special characters (e.g. Ã– = OE) **ASCII_Simplified** — ASCII characters **Database** — (default) Latin I or ASCII characters (as per reference database standard) **Latin** — Latin I characters **Latin_Alt** — Latin I characters (alternative transliteration) **Postal_Admin_Alt** — Latin I or ASCII characters (local postal administration alternative) **Postal_Admin_Pref** — Latin I or ASCII characters (as preferred by local postal administration) 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 387.
Option.Result.PreferredLanguage	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. **DATABASE** — Language derived from reference data for each address. Default. **ENGLISH** — English locality and state/province names output, if available.
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.

Parameter	Description
Option.Result.FormatDelimiter	Enables you to use non-standard formatting for multi-line addresses in the output. Acceptable values for this field include the following: • 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) **Note:** The same value must be selected for both the input option and output option.

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

Country	Database			Latin			
TWN	Hanzi	Hanzi	Hanzi	CHINESE_ CANTONESE	CHINESE_ MANDARIN	CHINESE_ CANTONESE + LATIN_SIMPLE	CHINESE_ CANTONESE + 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 42: ValidateAddressGlobal Process Options

Parameter	Description
Option.Process.OptimizationLevel	Use this option to set the appropriate balance between processing speed and quality. One of the following: **NARROW** The parser will honor input assignment strictly, with the exception of separation of House Number from Street information. **STANDARD** The parser will separate address element more actively as follows: • 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 **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. 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. 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.

Parameter	Description
Option.Process.Mode	Specifies the type of processing to perform on the addresses. One of the following: **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. **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. **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. **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. **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. **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.

Parameter	Description
Option.Process.MatchingScope	Specifies how closely an address must match the reference data in order for the address to be validated. One of the following: **Note:** These settings may not have an effect for countries lacking the necessary level of detail in the postal reference data. **ALL** — All address elements must match. **DELIVERYPOINT_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, street, house number, and sub building. **STREET_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, and street. **LOCALITY_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, and City/Locality/Suburb.

Response

Address Data

Table 43: Parsed Address Elements

Response Element	Description
AddressBlock1-9	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: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882

Response Element	Description
AddressLine1-6	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: AddressLine1: 4200 PARLIAMENT PL STE 600 LastLine: LANHAM MD 20706-1882
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, e.g. 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 44: 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 855.
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 45: Result Codes

Response Element	Result Code
AddressType	For United States and Canada addresses only, the AddressType field indicates the type of address. One of the following: **F** The address was validated/corrected to the firm name. **B** The address was validated/corrected to the building name. **G** The address is a general delivery address. **H** The address was validated/corrected to the high-rise default. **L** The address is a large volume receiver. **M** The address is a military address. **P** The address was validated/corrected to PO box. **R** The address was validated/corrected to a rural route. **S** The address was validated/corrected to a street address. **U** The address could not be validated/corrected so the type is unknown.
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.
CountOverflow	Indicates whether the number of candidate addresses exceeds the number returned. One of the following: **Yes** Yes, there are additional candidate addresses. To obtain the additional candidates, increase the value. **No** No, there are no additional candidates.
ElementInputStatus	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 401.
ElementRelevance	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 401.

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 401.
MailabilityScore	An estimate of how likely it is that mail sent to the address would be successful delivered. One of the following: **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 388.
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 (e.g. 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 (e.g. 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—StateProvice
- 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 (e.g. 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 Universal Addressing Module 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 the Universal Addressing Module.

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.pb.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 46: 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	The country code or name, in any of the following formats: • Two-character ISO 3116-1 Alpha-2 country code • Three-character ISO 3116-1 Alpha-3 country code • English country name See **ISO Country Codes and Module Support** on page 855 for a list of ISO codes.
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: Pitney Bowes

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 ValidateAddress Loqate.

Table 47: 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 423. **N** No, do not output field-level return codes (default). **Y** Yes, output field-level return codes.

Parameter	Description
Option.OutputFormattedOnFail	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. **N** No, do not format failed addresses (default). **Y** Yes, format failed addresses. 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**. If you check this option, you must select **Include a standard address** and/or **Include address line elements**. 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**. If you specify Y, you must specify "A" and/or "E" for OutputRecordType. 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**. If you specify Y, you must specify "A" and/or "E" for Option.OutputRecordType.

Parameter	Description
Option.OutputAddressBlocks	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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. One of the following: **N** No, do not return address blocks. Default. **Y** Yes, return address blocks.

Parameter	Description
Option.AmasFormatting	Specifies that output address data is to be formatted using Address Matching Approval System (AMAS) conventions. 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. This option modifies the output data as follows. • 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. Valid values are: **N** No, do not format the output data using AMAS conventions (default). **Y** Yes, format the output data using AMAS conventions. **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.
Option.OutputCasing	Specifies the casing of the output data. One of the following: **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

Parameter	Description
Option.HomeCountry	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: 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

Parameter	Description
Option.OutputCountryFormat	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". **E** Use English country names (default). **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.
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.

Parameter	Description
Acceptance level Option.AcceptanceLevel	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": • **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 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 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 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 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 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 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 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.

Parameter	Description
Option.IsDuplicateHandlingMaskEnable	Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options: **S** Selected by default. Pre-process the input and remove duplicates that occur in a single field. **C** Selected by default. Pre-process the input and remove duplicates across all fields. **T** Pre-process the input and remove duplicates in fields that are not standard address fields. **F** Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.
Option.MinimumMatchScore	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.
Option.KeepMultimatch	Specifies whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches. For more information, see **Returning Multiple Matches** on page 414.
Option.FailMultipleMatches	Fails multiple addresses for those input addresses that have more than one possible match.

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 48: 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 and is that a multiple match will return a failure if, whereas a multiple match 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 406. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information**Result Indicators** on page 423, see .

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 49: Standard Address Output

Response Element	Description
AdditionalInputData	Input data that could not be matched to a particular address component. For more information, see **About Additional Input Data**.
AddressLine1-4	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: AddressLine1 through AddressLine4.
City	The validated city name.
Country	The country in the format determined by what you selected in : • ISO Code • UPU Code • English
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.
PostalCode.Base	The 5-digit ZIP Code™; for example 20706.

Response Element	Description
StateProvince	The validated state or province abbreviation.

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 419.

Table 50: Parsed Address Output

Response Element	Description
AddressBlock1-9	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: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882
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**

Response Element	Description
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 : • 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.
PostalCode	Validated postal code. For U.S. addresses, this is the ZIP Code.
Principality *	The largest geographic data element within a country

Response Element	Description
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 51: 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 : • 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.
Principality.Input *	The largest geographic data element within a country
StateProvince.Input	Validated state or province name

Response Element	Description
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 52: Geocode Address Output

Response Element	Description
Geocode.MatchCode	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: **A** Multiple candidate geocodes were found to match the input address, and an average of these was returned **I** A geocode was able to be interpolated from the input addresses location in a range **P** A single geocode was found matching the input address **U** A geocode was not able to be generated for the input address The second byte represents the level of geocoding matching and is one of the following: **5** Delivery point (post box or subbuilding) **4** Premise or building **3** Thoroughfare **2** Locality **1** Administrative area **0** None
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 53: City/Street/Postal Code Centroid Match Codes

Element	Match Code
Address Point	P4

Element	Match Code
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 54: 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 Loqate Confidence Algorithm**.

Response Element	Description
CouldNotValidate	If no match was found, which address component could not be validated: • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber **Note:** More than one component may be returned, in a comma-separated list.
MatchScore	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. 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. **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.
ProcessedBy	Which address coder processed the address: **LOQATE** The Loqate coder processed the address.

Response Element	Description
Status	Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. • UnableToValidate
Status.Description	Description of the problem, if there is one. **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 55: 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.
	U	Unmatched. Does not apply to Canadian 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.
	U	Unmatched.
	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.
	U	Unmatched.
	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.
	U	Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U.
	V	Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description	
PostalCode.Type	**P**	The ZIP Code™ contains only PO Box addresses. U.S. addresses only.
	U	The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.
	M	The ZIP Code™ is for military addresses. U.S. addresses only.
	null	The ZIP Code™ is a standard ZIP Code™.
Principality.Result *	The largest geographic data element within a country	
StateProvince.Result	**A**	Appended. The field was added to a blank input field. U.S. and Canadian addresses only.
	C	Corrected. U.S. 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. U.S. and Canadian addresses only.
	R	The state 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. Does not apply to Canadian addresses.
	V	Validated. The data was confirmed correct and remained unchanged from input.
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's output. An example of an AVC is:

V44-I44-P6-100

AVCs consist of 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 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 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 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 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 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 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 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 ValidateAddressAUS.

Table 56: Output Fields

Response Element	Description	
Barcode	Standard barcode based on the DPID.	
	F	Failure (no barcode found)
	20-digit number	Success
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	

Response Element	Description
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

Universal Name Module

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 multi-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.pb.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 57: Open Name Parser Input

Parameter	Description
Data.CultureCode	The culture of the input name data. The options are listed below. **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 are also valid.
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 58: Open Name Parser Parsing Options

Parameter	Description
Option.ParseNaturalOrderPersonalNames	Specifies whether to parse names where the is in the order Title, First Name, Middle Name, Last Name, and Suffix. **true** Parse personal names that are in natural order. **false** Do not parse names that are in natural order.
Option.ParseReverseOrderPersonalNames	Specifies whether to parse names where the last name is specified first. **true** Parse personal names that are in reverse order. **false** Do not parse names that are in reverse order.
Option.ParseConjoinedNames	Specifies whether to parse conjoined names. **true** Parse conjoined names. **false** Do not parse conjoined names.
Option.SplitConjoinedNames	Specifies whether to separate names containing more than one individual into multiple records, for example, `Bill & Sally Smith`. **true** Split conjoined names. **false** Do not split conjoined names.
Option.ParseBusinessNames	Specifies whether to parse business names. **true** Parse business names. **false** Do not parse business names.

Parameter	Description
Option.OutputAsList	Specifies whether to return the parsed name elements in a list form. **true** Return the parsed elements in a list form. **false** Do not return the parsed elements in a list form.
Option.ShortcutThreshold	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. Specify a value from 0 to 100. The default is 100.

Parameters for Culture Options

The following table lists the options that control name cultures.

Table 59: Open Name Parser Cultures Options

Parameter	Description
Option.DefaultCulture	Specifies which culture(s) you want to include in the parsing grammar. Global Culture is the default selection. 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: `es,ja,,`

Parameters for Advanced Options

The following table lists the advanced options for name parsing.

Table 60: 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	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.
Option.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.ReverseOrderConjoinedPersonalNamesPriority	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. 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.BusinessNamesDomain	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.

Option	Description
Option.BusinessNamesPriority	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. 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.

Response

Table 61: 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, "Pitney Bowes".
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

Enterprise Routing Module

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 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 the Enterprise Routing Module.

> **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://www.pb.com/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.pb.com/"

      xmlns:ns3="http://www.mapinfo.com/midev/service/geometries/v1"
     xmlns:ns4="http://www.pb.com/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://www.pb.com/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.pb.com/"

      xmlns:ns3="http://www.mapinfo.com/midev/service/geometries/v1"
     xmlns:ns4="http://www.pb.com/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 62: 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	(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. Use this field to override the default travel boundary cost on a record-by-record basis. 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.
TravelBoundaryCostUnits	String	(Optional) The type of metric used to calculate the travel boundary. One of the following: • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Yards Use this field to override the default travel boundary cost units on a record-by-record basis.

Parameters for Options

Input

Table 63: 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 *Spectrum™ Technology Platform Spatial Guide*.
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 `http://<server>:<port>/Spatial/GeometryService/DemoPage.html`.
CoordinateFormat	Specifies the format of latitude/longitude coordinates in the input. **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. One of the following: **Decimal** (90.000000, 180.000000) **DecimalAssumed** (90000000, 180000000). Default. **DegreesMinutesSeconds** (90 00 00N, 180 00 00W) **PreZero** (090000000N, 180000000W) **PreZeroDecimal** (090.000000N, 180.000000W)
DefaultTravelBoundaryCost	Number of cost units. This can be any double value (includes decimals). The default is 10.

Parameter	Description
DefaultTravelBoundaryCostUnits	Type of metric you want to use to calculate the travel boundary. One of the following: • Feet • Hours • Kilometers • Meters • Miles • Minutes • Seconds • Milliseconds • Yards
HistoricTrafficTimeBucket	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: **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.

Output

Table 64: GetTravelBoundary Output Options

Parameter	Description
ResultType	Specifies the type of result you want returned. One of the following: **AccessibleNodes** Returns the entire isoChrone. **Geometry** Returns all of the points along the road network that can be reached for the isoChrone calculation. **StartNodes** Returns the location specified by the entered address.
SimplificationFactor	Specifies what percentage of the original points should be returned or upon which the resulting polygon should be based.
BandingStyle	Specifies the style of banding to be used in the result. Banding styles are the types of multiple isoChrone 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.
ReturnIslands	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.

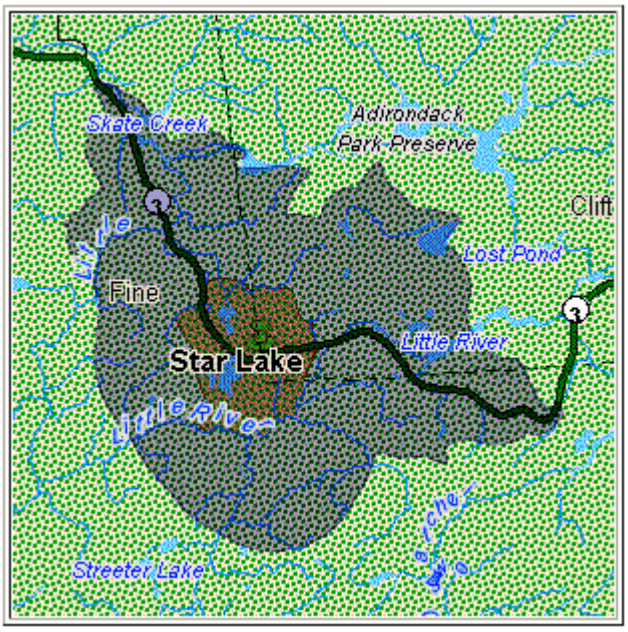
Travel

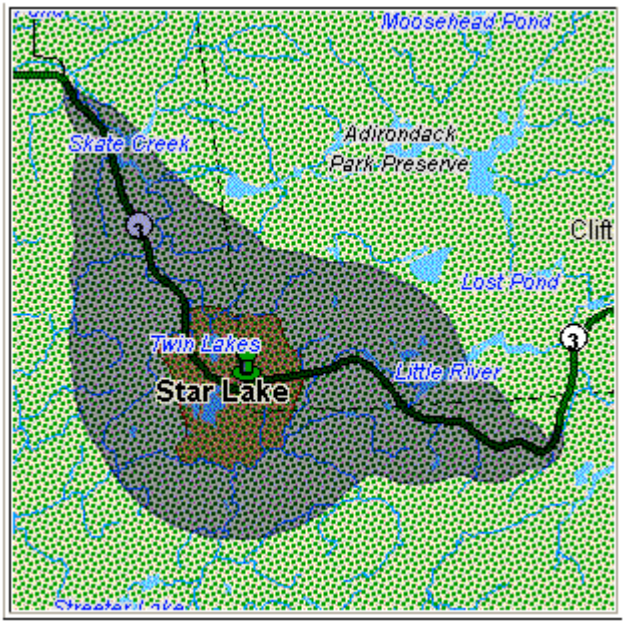
Travel options specify assumptions to make about travel speed off network roads and whether to use only major roads when calculating the travel boundary. Most travel options have to do with ambient speed.

Table 65: 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.
Units	The units of measure in which you want the data returned. One of the following: • Kilometer (default) • Meter • Mile

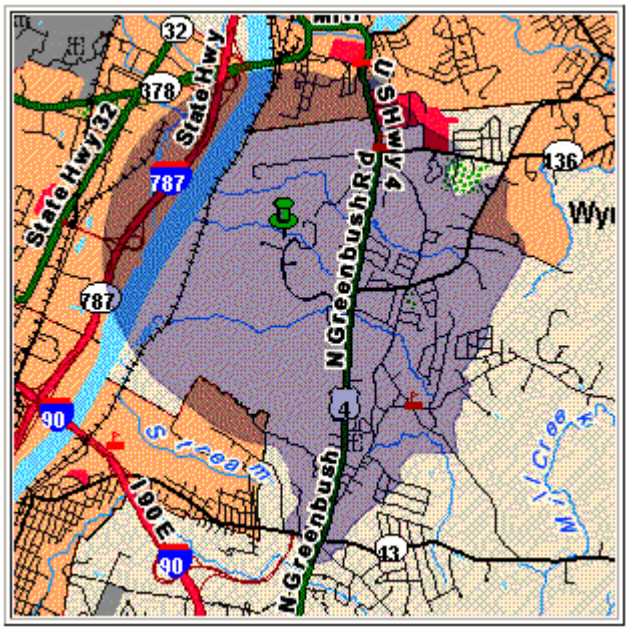
Parameter	Description
MajorRoads	

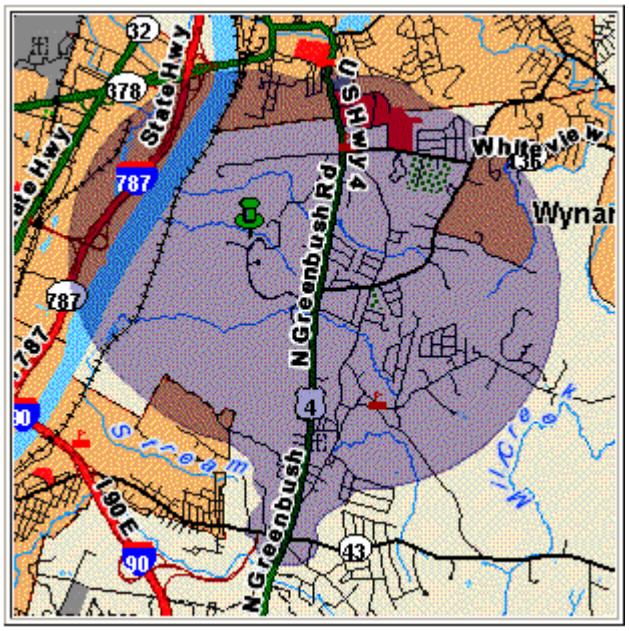
Parameter	Description
	Specifies whether to include all roads in the calculation or just major roads. By default, the Get Travel Boundary is calculated with major roads set to true. This improves performance but the accuracy may decrease. This map represents a travel boundary with travel allowed on all roads:  This map represents a travel boundary with travel restricted to major roads only:

Parameter	Description
	 One of the following: **Y** Include only major roads in the calculation. Default. **N** Include all roads in the calculation.
DefaultAmbientSpeed	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. 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. 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. **Note:** If you are calculating pedestrian travel boundaries we recommend that you change the default ambient speed to 3 MPH (5 KPH).

Parameter	Description
AmbientSpeedUnit	The unit of measure to use with the value specified in the DefaultAmbientSpeed option. **KPH** Kilometers per hour. **MPH** MILES per hour. Default. **MTPS** Meters per second. **MTPM** Meters per minute.

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
	• 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

Response

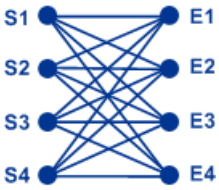
Get Travel Boundary returns the following fields:

Table 66: GetTravelBoundary Outputs

Response Element	Format	Description
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: • 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.

GetTravelCostMatrix

GetTravelCostMatrix calculates the travel time and distances between an array of start and end locations. You can use GetTravelCostMatrix to find the shortest or fastest paths between a number of start points and a number of end points, and determine the total time and distance of the individual routes (the route costs). For example if you input four start points (S1 through S4) and four end points (E1 through E4), a total of 16 routes will be returned as illustrated in the following diagram:



Matrix routing is often used to determine service response time and coverages for specific services such as a fire house or police station. You may require these calculations to ensure they can provide adequate coverage for service level agreements such as having one or more people who can respond to an incident within 20 minutes of first notification. The calculation can be used an analysis tool to 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://www.pb.com/spectrum/services/GetTravelCostMatrix"

    xmlns:spec="http://spectrum.pb.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.pb.com/"
       xmlns:ns3="http://www.g1.com/services/erm/types"
       xmlns:ns4="http://www.g1.com/services/GetTravelCostMatrix"

xmlns:ns5="http://www.pb.com/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 67: 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 CoordinateFormat option.
Longitude	The longitude for a start or end point. Specify the longitude using the format selected in the CoordinateFormat 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 StartPointID or EndPointID field in output.
MatrixTransientUpdate	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). For transient update options and example, see **Transient Options** on page 477.

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.

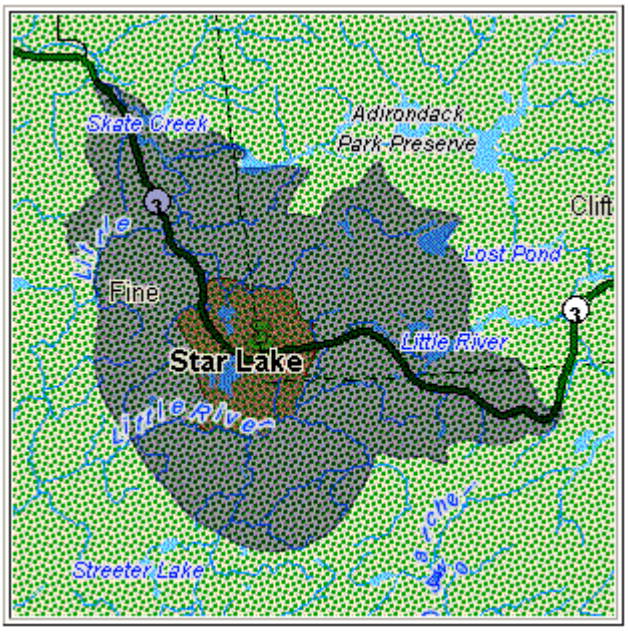
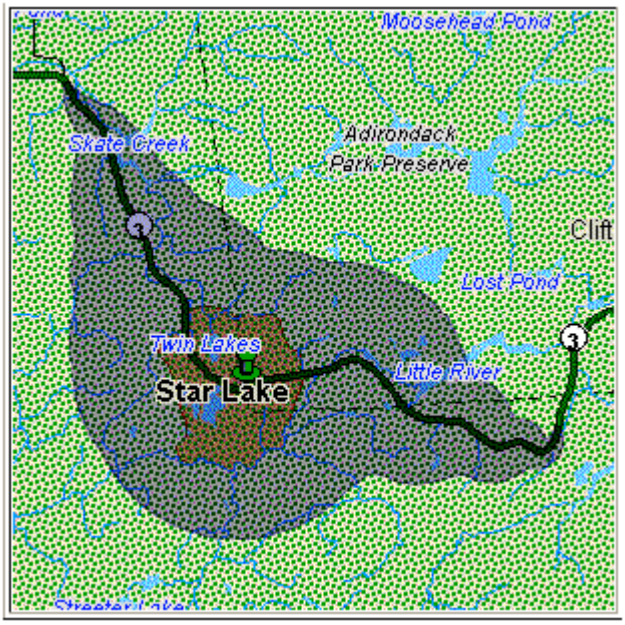
Table 68: Travel Preferences Options

Parameter	Description
RoadType_<type>	

Parameter	Description
	Specifies the priority to give to different types of roads when determining the route. • 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 • mormal 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	Description
	• secondary highway suburban • secondary highway urban For each road type you can specify one of the following: **Avoid** Exclude the road type from routes if possible. **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. **High** Prefer the road type over other road types. **Low** Prefer other road types over this road type. **Medium** Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium.

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
	One of the following: **Y** Include only major roads in the calculation. Default. **N** Include all roads in the calculation.

Routing

Table 69: 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 *Spectrum™ Technology Platform Spatial Guide*.
OptimizeBy	Specifies whether to find the shortest distance or the shortest time. One of the following: **Time** Optimize by fastest time traveled. Default. **Distance** Optimize by shortest distance traveled.
ReturnOptimalRoutesOnly	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. 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. **Y** Yes, return only optimal routes. Default. **N** No, return all routes not just the optimal routes.

Parameter	Description
RouteCostMatrixFormat	Specifies the data format of the route cost matrix. One of the following: **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. **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.
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 `http://<server>:<port>/Spatial/GeometryService/DemoPage.html`.
CoordinateFormat	Specifies the format of latitude/longitude coordinates in the input. **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. One of the following: **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	Specifies how GetTravelCostMatrix should return distance values. One of the following: **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	Specifies how to return time. One of the following: **Hours** Return time in hours. **Minutes** Return time in minutes. Default. **Seconds** Return time in seconds. **Milliseconds** Return time in milliseconds.
HistoricTrafficTimeBucket	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: **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.

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 70: Transient Update Options

Parameter	Description
PointUpdate	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 one of the following: **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). **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 `Exclude` parameter defined as Y. Valid values are Y (yes) and N (no).

Parameter	Description
SegmentUpdate	

Parameter	Description
	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 `RoutingSegmentID`, then specify one of the following: **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). **RoadType** This is a string value to change the value of the road type for the segment for the route calculation. The `RoadType` can be one of the following: • 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

Parameter	Description
	• 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 • mormal 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	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 `Exclude` parameter defined as Y. Valid values are Y (yes) and N (no).

Parameter	Description
RoadTypeUpdate	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` to update (see the above road types in the segment update), then specify one of the following: **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).

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://www.pb.com/spectrum/services/GetTravelCostMatrix"

    xmlns:spec="http://spectrum.pb.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: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 71: 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: **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. **Note:** This field is active only when the **Route Cost Matrix Format** field is set to "Hierarchy."
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.
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."

Response Element	Format	Description
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: **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. **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 **Get 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: Enterprise Routing Module - <Country>. Each database also has its own country code. For example, the name of the Austrian database is "Enterprise Routing Module - 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 the Enterprise Routing Module.

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://www.pb.com/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: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.pb.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://www.pb.com/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: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 72: 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 *Spectrum™ Technology Platform Dataflow Designer's Guide*.

Parameter	Format	Description
Language	String	Specifies the language in which GetTravelDirections should return directions. One of the following: **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. **Note:** An entry in this input field will override an entry in the Default Language option.

Parameter	Format	Description
TravelDirectionTransientUpdate	List	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). For transient update options and example, see **Transient Options** on page 502.

Parameters for Options

Routing

The following table lists the configuration options for GetTravelDirections.

Table 73: Get Travel Directions Configuration 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 *Spectrum™ Technology Platform Spatial Guide*.
OptimizeBy	Specifies whether GetTravelDirections should find the shortest distance or the shortest time. One of the following: **Time** — Optimize by shortest time traveled. Default. **Distance** — Optimize by shortest distance traveled.
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 `http://<server>:<port>/Spatial/GeometryService/DemoPage.html`.

Parameter	Description
CoordinateFormat	Specifies the format of latitude/longitude coordinates in the input. **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. One of the following: **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	Specifies how directions should be returned. One of the following: **Normal** Return directions in a normal format. Default. **Terse** Return directions in a terse format. Terse directions are more suitable for wireless devices.
DistanceUnits	Specifies how GetTravelDirections should return distance values. One of the following: **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	Specifies how GetTravelDirections should return time. One of the following: **Hours** Return time in hours. **Minutes** Return time in minutes. Default. **Seconds** Return time in seconds. **Milliseconds** Return time in milliseconds.

Parameter	Description
FocusOfRoute	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. One of the following: **Start** Return just the start of the route. **End** Return just the end of the route. **None** Return the whole route. Default.
HistoricTrafficTimeBucket	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: **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.
OptimizeIntermediatePoints	Specifies whether GetTravelDirections re-orders the intermediate points in an optimal manner during route computation. **Y** Yes, optimize intermediate points. Directions optimally orders the intermediate points. Default. **N** No, do not optimize intermediate points. Directions preserves the specified order of the points.

Directions

Table 74: Get Travel Directions Direction Options

Parameter	Description
ShowDistance	Specifies whether to return the distance for the route. **Y** Yes, return the distance for the route. Default. **N** No, do not return the distance for the route.
ReturnRouteDirections	Specifies whether to return turn-by-turn directions for the route. This option is enabled by default. **Y** Yes, return textual turn-by-turn directions. Default. **N** No, do not return textual turn-by-turn directions.
ReturnSegmentGeometry	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 496.
ReturnRouteGeometry	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. One of the following: **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	Specifies the language in which GetTravelDirections should return directions. One of the following: **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	Specifies whether GetTravelDirections should return the time it takes to follow a direction within the route. **Y** Yes, return the time for the route. Default. **N** No, do not return the time for the route.

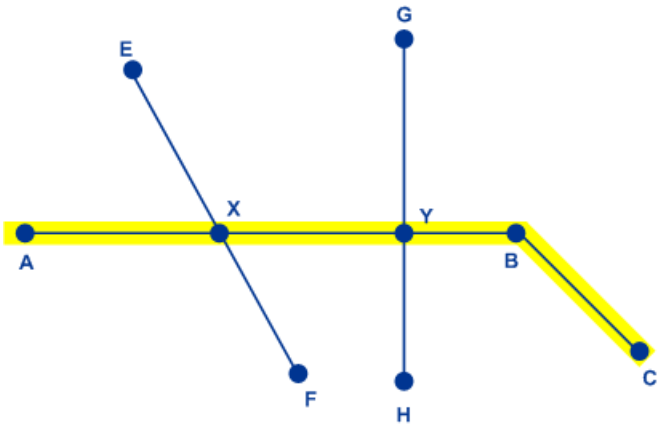
Parameter	Description
ShowPrimaryNameOnly	Specifies whether GetTravelDirections should return all names for a given street in the directions or if it should return just the primary name for a street. **Y** Yes, return just the primary name. **N** Return all names for a street. Default.

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 75: Travel Preferences Options

Parameter	Description
RoadType_<type>	

Parameter	Description
	Specifies the priority to give to different types of roads when determining the route. • 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 • mormal 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

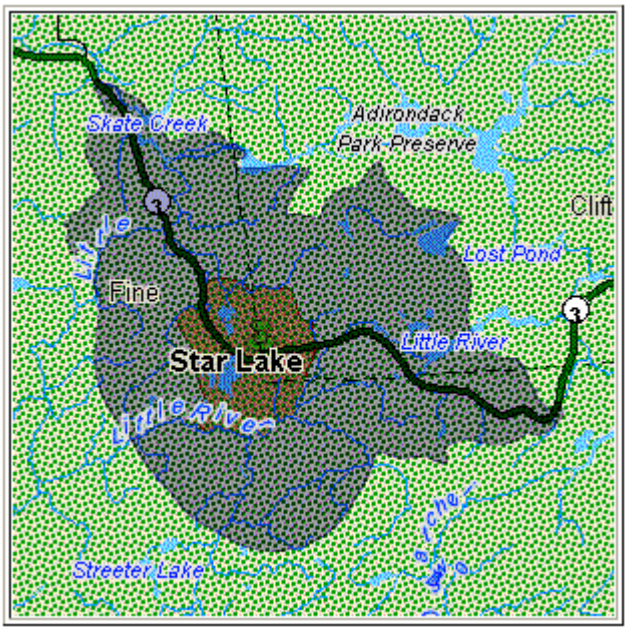
Parameter	Description
	• secondary highway suburban • secondary highway urban For each road type you can specify one of the following: **Avoid** Exclude the road type from routes if possible. **Note:** It is not always possible to exclude a road type from the travel directions. Depending on the situation, 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. **High** Prefer the road type over other road types. **Low** Prefer other road types over this road type. **Medium** Give this road type equal preference with other road types. If no preference is specified for a road type, the default is Medium.

Parameter	Description
MajorRoads	

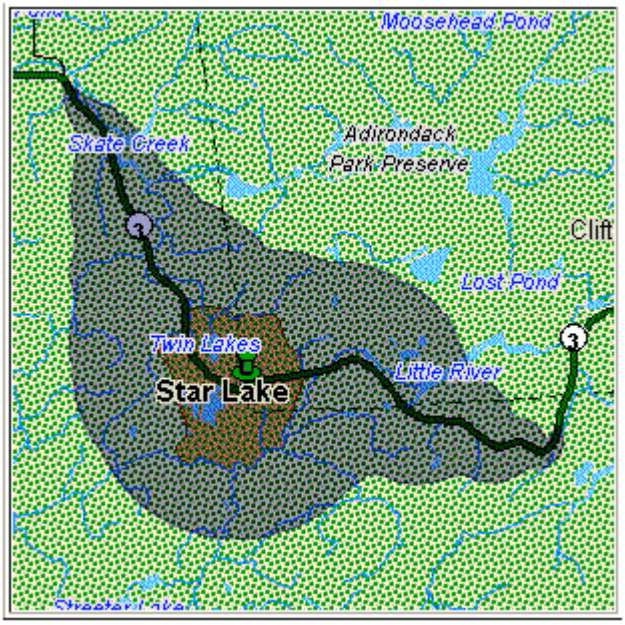
Parameter	Description

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
	One of the following: **Y** Include only major roads in the calculation. Default. **N** Include all roads in the calculation.

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 76: Transient Update Options

Parameter	Description
PointUpdate	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 one of the following: **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). **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 `Exclude` parameter defined as Y. Valid values are Y (yes) and N (no).

Parameter	Description
SegmentUpdate	

Parameter	Description
	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 `RoutingSegmentID`, then specify one of the following: **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). **RoadType** This is a string value to change the value of the road type for the segment for the route calculation. The `RoadType` can be one of the following: • 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

Parameter	Description
	• 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 • mormal 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	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 `Exclude` parameter defined as Y. Valid values are Y (yes) and N (no).

Parameter	Description
RoadTypeUpdate	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` to update (see the above road types in the segment update), then specify one of the following: **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).

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://www.pb.com/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: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>
```

```
                                    <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:TravelDirectionTransientUpdate>
               </get:PointToPointRequest>
            </get:input_port>
         </get:GetTravelDirectionsRequest>
      </soapenv:Body>
</soapenv:Envelope>
```

Response

GetTravelDirections returns the following fields:

Table 77: GetTravelDirections Outputs

Response Element	Format	Description
Distance	String	The distance of each segment in the route.
DistanceUnits	String	The unit of measurement for distance.
Format	String	Value of the format used to generate directions.
Language	String	Language of the directions.
RouteDirections	List	Turn-by-turn directions for the route.

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 496 .
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: • 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 the Enterprise Routing Module.

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>
                                 <Lattitude>40.024421</Lattitude>
                              </RouteDataPoint>
                              <RouteDataPoint>
                                 <Longitude>-74.824133</Longitude>
                                 <Lattitude>40.024149</Lattitude>
                              </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 78: 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 Enterprise Routing Module Routing Database Resource tool. For more information, see the *Spectrum™ Technology Platform Administration Guide*.
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 `http://<server>:<port>/Spatial/GeometryService/DemoPage.html`.

Options

Table 79: Get Route Data Options

Parameter	Description
CoordinateFormat	This is astring value to specify the format of the coordinates for the point data returned in the response. The options are `Decimal`, `DecimalAssumed`, `PreZero`, `PreZeroDecimal`, and `DegMinSec`.
DistanceUnits	This is astring 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 astring 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`.

Parameter	Description
VelocityUnits	This is astring 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 astring 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 astring 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 astring 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 80: GetRouteData Outputs

Response Element	Format	Description
RoutingSegmentID	String	One or more route segment IDs

Response Element	Format	Description
SegmentData	String	All information for the route segmenet. This information includes: • PrimaryName • PrimaryNameLanguage • AlternateNameList • SegmentLength • SegmentLengthUnit • TimeTaken • TimeUnit • TurnAngle • TurnAngleUnit • speedOfTravel • speedOfTravelUnit • RoadType • SegmentDirection • StartJunctionType • IsRoundabout • IsTollRoad • PointsInSegment The route geometries (points) for the segment is returned in the `PointsInSegment` parameter.

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 the Enterprise Routing Module.

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 81: 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 Enterprise Routing Module Routing Database Resource tool. For more information, see the *Spectrum™ Technology Platform Administration Guide*.
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 `http://<server>:<port>/Spatial/GeometryService/DemoPage.html`.
RestoreDefaults	All persistent updates made to the `DataSetResourceName` will be reverted to their original state. Specify `Y` 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 82: 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 astring value to exclude the specified point from all route calculations. 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).
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 83: Segment Update Options

Parameter	Description
RoutingSegmentID	The route segment ID to perform the persistent update.
Exclude	This is astring value to exclude the specified segment from all route calculations. To exclude a segment you need to specify the segment ID and include the `Exclude` parameter defined as Y. Valid values are Y (yes) and N (no).

Parameter	Description
RoadType	

Parameter	Description
	This is a string value to change the value of the road type for the segment for the route calculation. The `RoadType` can be one of the following: • 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 • mormal 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
	• 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 84: Road Type Update Options

Parameter	Description
RoadType	

Parameter	Description
	This is a string value to change the speed value of the road type for the route calculation. The road type can be one of the following: • 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
	• 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 85: 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 `Latitude` and `Longitude`, and specify the type of update to be reset for the point. The options for the `Reset Type` are `Exclude` and `Speed`.

Parameter	Description
SegmentReset	The segment that has the persistent update to be reset. You must specify the `RoutingSegmentID` and specify the type of update to be reset for the segment. The options for the `Reset Type` are `Exclude`, `Road Type`, and `Speed`.
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`. Similarily 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 86: PersistentUpdate Outputs

Response Element	Format	Description
SuccessMessage	String	Returns if the update or reset was completed successfully.

Enterprise Tax Module

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.
- **Property tax districts**—Areas designated for the collection of taxes imposed on property owners to support local government facilities and services based on the property's location. Examples include taxes collected for districts to pay for schools, police, or other services. Property tax districts are created by local governments.
- **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 Pitney Bowes sales representative for more information.
- **Sales and Use Tax Rates**—Using the optional Pitney Bowes 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 the Enterprise Tax Module.

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.pb.com/spectrum/services/AssignGeoTAXInfo"
xmlns:spec="http://spectrum.pb.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>
```

This would be the response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <ns3:AssignGeoTAXInfoResponse xmlns:ns2="http://spectrum.pb.com/"

      xmlns:ns3="http://www.pb.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>000000000</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

Parameter	Format	Description
BufferWidth	String [10]	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 852. 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.
CandidateID	String [1]	For a multi-match outcome, indicates which match result should be returned; valid values are 1-9.
City	String [50]	City name
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 852. 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. **Y** Use the GeoTAX Auxiliary file for matching. (default) **N** Do not use the GeoTAX Auxiliary file for matching.
UseAuxiliaryFile	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. **Y** Use the User Auxiliary file for matching. **N** Do not use the User Auxiliary file for matching. (default)
UseStateProvidedFile	Specifies whether to attempt a match to the state-supplied file. Use this option in combination with `FileSearchOrder` to specify a state-supplied file to use. 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. 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. 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. **Y** Use the State-supplied file for matching. **N** Do not use the State-supplied file for matching. (default) **Note:** You must install the appropriate State-supplied file to use these options. For instructions, see the *Spectrum™ Technology Platform Installation Guide*.

Parameter	Description
FileSearchOrder	Specifies which state-supplied file to use. This option only takes effect if you specify `UseStateProvidedFile=Y`. One of the following: **FLOnly** Use only the Florida-native formatted file. (default) **TSOnly** Use only the TS-158 formatted file.
UseRelaxedSecondaryMatching	Specifies whether input addresses with secondary information are matched to records without secondary information. This option applies only to Florida-native files. **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.	
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. **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.

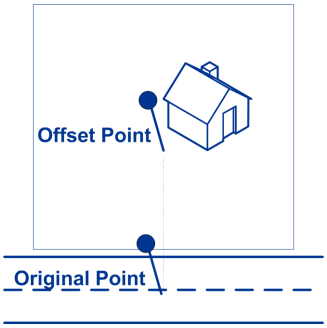
Parameter	Description
GsSearchArea	The search area options allow for searching the address' finance area or an expanded area specified by distance. **1** Searches the entire finance area for a match. A finance area is a region defined by the U.S. Postal Service and typically consists of a set of contiguous 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.
GsSearchRadius	Radius for search area. **1-99 miles** Search radius. (default = 0 miles)
GsEnableFirstLetterExpanded	Looks for the correct first letter of a street address if the first letter is missing or incorrect. The Enterprise Tax Module searches through the alphabet looking for possible correct first letters to complete the street address. **Note:** This feature is disabled by default and cannot be enabled in Exact mode. **Y** Enable first letter change matches. **N** Do not allow first letter change matches. (default)
GsEnableRangedAddress	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. **Note:** This feature is disabled by default and cannot be enabled in Exact mode. **Y** Allow address range matches. **N** Do not allow address range matches. (default)

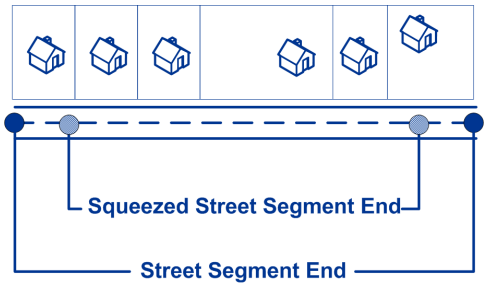
Parameter	Description
GsAlternateLookup	This option allows specifying the preferred way to match when both an address and firm name are provided. The matching method can be set to match to the address rather than the firm name or vice versa. If neither are specified, the default matching method is to match to the address line only. **1** Searches for street name, but if there isn't a match, will use the firm name. **2** Looks up the firm name, but if there isn't a match, will use the street name. **3** Searches only street records. (default)
GsMultiMatchResolution	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: **N** No matches are returned. (default) **R** Return the first match candidate in the list. **A** The information for all the match candidates is returned.

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	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. **60** Sixty feet offset from street centerline.

Parameter	Description
Squeeze	Specifies if the street end points should be "squeezed" when determining the geocode of an address in street-level matching. When `Squeeze` 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.  **Y** Apply squeeze. **N** Do not apply squeeze. (default)
LatLongFormat	Indicates the desired format for the returned latitude/longitude. One of the following: **PreZero** Decimal degrees using directional indicator (no decimal point). For example, 090000000N180000000W. **PreZeroDecimal** Decimal degrees using directional indicator. For example, 090.000000N180.000000W. (default) **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. **DegMinSec** Degrees, minutes, seconds. For example, 90 00 00N180 00 00W.
Expanded Geocoding options: These options enable additional geocoding functionality.	

Parameter	Description
GsEnableAddressPointInterpolation	Address point interpolation uses a patented process that improves upon regular street segment interpolation by inserting point data into the interpolation process. **Note:** This feature is only for use with point-level geocoding. 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, the Enterprise Tax Module attempts to find high and low boundary address points to use for address point interpolation. **Y** Enable address point interpolation. **N** Disable address point interpolation. (default)
Minimum geocode quality	
GsEnableGeographicFallback	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. **Y** If a definitive match cannot be made, then return the next higher level geographic centroid. **N** Disable geographic fallback feature. (default)

Parameter	Description
GsEnableStreetCentroid	If an input street address cannot be found using the street number and name, the Enterprise Tax Module then searches the input ZIP Code or city/state for the closest match. If the Enterprise Tax Module 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. 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 Enterprise Tax Module does not change the street name on the output address. **Y** If a street or point match cannot be made, then return a street level centroid. **N** Do not return a street level centroid if a match cannot be made. (default) **Note:** This feature should only be used for exception processing or research. It should not be used in a production process.
Boundary matching: These 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: **Feet** Distances are measured in feet. (default) **Meters** Distances are measured in meters.
DefaultBufferWidth	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, the Personal Property Tax District (PTD) file and Place and MCD files. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. For more information about buffers, see **Buffering** on page 852.

Parameter	Description
DefaultUserBufferWidth	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 852. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. **Note:** To use buffers, the user-defined boundary file must support buffers.

Output Data Options

Data options control the data returned by AssignGeoTAXInfo.

Parameter	Description
GeoTAXOutputRecordType	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 `OutputFields` option. • **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 You can also specify one, and only one, of the following: **I** Insurance Premium Tax District (IPD) **P** Property Tax District (PTD) **R** Payroll Tax District (PAY) **S** Special Purpose Tax District (SPD) For a description of the fields in each output group, see **Response** on page 134. **Note:** If you specify `W`, to obtain the best payroll system tax code match possible.

Parameter	Description
TaxKey	If you use third-party tax compliance software from Vertex or Taxware, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following: **N** Do not return either the Taxware or Vertex jurisdiction codes (default). **T** Return the Taxware jurisdiction code for the address. **Y** Return the Vertex jurisdiction code for the address. Select this option if you obtained a MatchMaster file from Vertex. **V** Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Pitney Bowes. **Note:** The Special Purpose District data is needed to achieve the best results from this option.
TaxRate	Indicates the sales and use tax rate type to return or none: **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:** The Special Purpose District data is needed to achieve the best results from this option.
OutputFields	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. For a list of the fields included in each data type, see **Response** on page 134.

Output Format

Output format options control how AssignGeoTAXInfo formats output data.

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

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	Indicates the file from which the match was obtained. One of the following: • 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	General Return Code indicating the type of match. **5** ZIP Code match **9** ZIP+4 Code 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
AddressMatch.Lastline*	61	The complete matched last address line (city, state, and postal code).
AddressMatch.LocationCode*	5	The Location Code indicates the methodology used to complete the geocode and may also provide some information about the quality of the geocode. For the list of location codes, see **Location Codes.**
AddressMatch.MatchCode*	5	The Match Code indicates the portions of the address that matched or did not match to the reference file. For the list of match codes, see **Match Codes.**
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	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. The AddressMatch.PBKey field has "P" as the leading character, for example: P00001XSF1IF.
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	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

Response Element	Max. Field Length with null terminator	Description

Confidence*	4	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:
		
		Census.MatchCode + LatLong.StreetMatchCode + LatLong.MatchCode
		
		The maximum confidence score is 100, so if this calculation results in a value greater than 100, the Confidence score is returned as 100.
		
		The following values are used:
		
		• Census.MatchCode
		  • 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
		
		• LatLong.StreetMatchCode
		  • H = 5
		  • L = 15
		  • S = -10
		  • Z = -5
		  • null = 0

Response Element	Max. Field Length with null terminator	Description

		• LatLong.MatchCode
		  • 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 151.		
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	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.txb file. For more information on buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
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.txb file.
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.

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 852. **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	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.txb file. For more information on buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
MCD.DistanceToBorder*	10	Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb 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 `AddressMatch.GenRC` 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 `LatLong.MatchCode` 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
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

Response Element	Max. Field Length with null terminator	Description
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.

Response Element	Max. Field Length with null terminator	Description
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 address 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

Response Element	Max. Field Length with null terminator	Description
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 119. 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.
PTC*n*.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.

Response Element	Max. Field Length with null terminator	Description
PTC*n*.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*n*.PayrollDescription	41 per PTC	A description of the purpose of this payroll code.
PTC*n*.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 jurisdiciton data in the output. Contents returned when available; otherwise, left blank.

Response Element	Max. Field Length with null terminator	Description
Confidence.SurfaceType	3	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. **0** Undefined **1** The search failed - the address was not found. **2** Intersection confidence surface generated. **3** Interpolated street segment. **4** Point-level match. **5** State confidence surface generated. **6** County confidence surface generated. **7** City confidence surface generated. **8** Reserved **9** A ZIP Code confidence surface generated. **10** A ZIP+2 confidence surface generated. **11** A ZIP+4 confidence surface generated. **12** Reserved **13** A street centroid confidence surface generated.
GeoTAXKey	10	The value in this field varies depending on the option you specified in the TaxKey option: If you specified `T`, GeoTAXKey contains the proprietary codes used in Taxware tax compliance software. You can use this code in your Taxware application to find out the tax rate for the jurisdiction. If you specified `Y` or `V`, GeoTAXKey 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.

Response Element	Max. Field Length with null terminator	Description
GeoTAXKey.MatchCode	2	Return code denoting the level of match obtained against the Vertex or Taxware cross reference files. **E** Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name. **P** Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code. **A** Alternate match using two fields: ZIP Code, FIPS place name. **N** Record is default coded based on valid state code. **null** No matching record found.
GeoTAXKey.MatchLevel	12	A description of the value returned in the `GeoTAXKey.MatchCode` field. **Exact** Exact match. See description in GeoTAXKey.MatchCode. **Partial** Partial match. See description in GeoTAXKey.MatchCode. **Alternate** Alternate match. See description in GeoTAXKey.MatchCode. **DefaultCode** Record is default coded. See description in GeoTAXKey.MatchCode. **NoMatch** No matching record found.
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	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 852. **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.
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*n*.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	Indicates where in the polygon the address resides in relation to the edge of the area. Buffer width is specified by the option DefaultUserBufferWidth or by the input field BufferWidth . One of the following: **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. For more information, see **Buffering** on page 852.
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 DistanceUnits.
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, PTDs, 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*n*.BoundaryBuffer.BufferRelation	2 per IPD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
IPD*n*.BoundaryBuffer.DistanceToBorder	10 per IPD	Indicates the distance from the address to the border of the district.
IPD*n*.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*n*.DistrictID	11 per IPD	IPD ID.
IPD*n*.DistrictName	61 per IPD	IPD name.
IPD*n*.DistrictType	7 per IPD	IPD district type.
IPD*n*.UpdateDate	7 per IPD	IPD update date (MMYYYY).

Response Element	Max. Field Length with null terminator	Description
IPD*n*.VersionDate	7 per IPD	IPD compiled date (MMYYYY).
IPD*n*.Notes	21 per IPD	Tax code descriptions. For example: 01, 33, A, B
IPD*n*.ChangeDate	7 per IPD	IPD change date.
IPD*n*.EffectiveDate	7 per IPD	MMDDYY - Identifies when district becomes active - State supplied For example: 010108
IPD*n*.ExpirationDate	7 per IPD	MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009
IPD*n*.FireRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.CasualtyRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.VehicleRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MarineRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.HealthRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.LifeRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.OtherRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MinimumRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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 119. To include this data in the output, set `GeoTAXOutputRecordType` = `R`.

Note: AssignGeoTAXInfo returns multiple districts for IPDs, SPDs, PTDs, 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*n*.BoundaryBuffer.BufferRelation	2 per PAY	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option DefaultBufferWidth or by the input field BufferWidth . One of the following: **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. For more information, see **Buffering** on page 852.
PAY*n*.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 DistanceUnits.
PAY*n*.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*n*.DistrictID	11 per PAY	PAY district ID.
PAY*n*.DistrictName	61 per PAY	PAY district name.
PAY*n*.DistrictType	7 per PAY	PAY district type.
PAY*n*.ID	11 per PAY	PAY ID.

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

Property Tax Districts

The table below lists the output fields that contain Property Tax Districts (PTD) data. For more information about property tax districts, see **AssignGeoTAXInfo** on page 119. To include this data in the output, set `GeoTAXOutputRecordType` = `P`.

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

Response Element	Max. Field Length with null terminator	Description
NumberPTDsFound	3	Number of PTDs returned.
PTD*n*.BoundaryBuffer.BufferRelation	2 per PTD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
PTD*n*.BoundaryBuffer.DistanceToBorder	10 per PTD	Indicates the distance from the address to the border of the district. The distance is in the units specified by the option `DistanceUnits`.
PTD*n*.BoundaryConfidence	4 per PTD	Provides the percentage overlap of the geocode to the PTD boundary polygon layer. The returned percentage value describes the probability that the point falls in the PTD boundary area.
PTD*n*.DistrictID	11 per PTD	PTD district ID.

Response Element	Max. Field Length with null terminator	Description
PTD*n*.DistrictName	61 per PTD	PTD district name.
PTD*n*.DistrictType	2 per PTD	PTD district type. Only returned for Indiana. **R** Reporting district **B** Billing district
PTD*n*.JurisdictionID	11 per PTD	PTD jurisdiction ID.
PTD*n*.UpdateDate	7 per PTD	PTD update date.

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 119. To include this data in the output, set `GeoTAXOutputRecordType` = `S`.

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

Response Element	Max. Field Length with null terminator	Description
NumberSPDsFound	3	Number of SPDs returned.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.BoundaryBuffer.BufferRelation	2 per SPD	Indicates where in the district the address resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
SPD*n*.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 `DistanceUnits`.
SPD*n*.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*n*.CompiledDate	7 per SPD	SPD compiled date.
SPD*n*.DistrictCode	4 per SPD	3-digit district type code.
SPD*n*.DistrictName	61 per SPD	SPD name.
SPD*n*.DistrictNumber	6 per SPD	SPD district number.
SPD*n*.EffectiveDate	7 per SPD	SPD effective date.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.UpdateDate	7 per SPD	SPD update date.
SPD*n*.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 Pitney Bowes 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 Pitney Bowes 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.

Response Element	Max. Field Length with null terminator	Description
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).
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	This field contains a return code if the GeoTAX engine experiences an abnormal termination. **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. The first character indicates the file (or set of files affected). **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 The second position is one of the following: **E** Fatal issue, program terminating **F** Expired database **I** Informational

Response Element	Max. Field Length with null terminator	Description
GTX.ErrorDescription*	81	

Response Element	Max. Field Length with null terminator	Description
		If the GeoTAX engine experiences an abnormal termination, this field contains a text description of the reason. It is blank if GeoTAX terminated normally. **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. 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" 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" 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 = "

Response Element	Max. Field Length with null terminator	Description
		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 `WN` 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.
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.
Status	2	Reports the success or failure of the match attempt. **null** Success **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.
Status.Code	12	If AssignGeoTAXInfo could not process the address, this field will show the reason. Currently there is one possible value for this field: Invalid Address.

Response Element	Max. Field Length with null terminator	Description
Status.Description	64	If AssignGeoTAXInfo could not process the address, this field will show a description of the failure. One of the following: 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 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 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 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 the Enterprise Tax Module.

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.pb.com/spectrum/services/CalculateDistance"
xmlns:spec="http://spectrum.pb.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: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.pb.com/"

     xmlns:ns3="http://www.pb.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. **N** North **S** South
FirstLongitude	Longitude of the first point for which you want distance returned.
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.

Parameter	Description
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** (090000000N180000000W). (default) **PreZeroDecimal** (090.000000N180.000000W) **Decimal** (90.000000-180.000000) **DecimalAssumed** (90000000-180000000)
ReturnUnits	Indicates the measurement units returned for distance calculation: • 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 Pitney Bowes 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 ReverseGeoTAXInfo Lookup 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.
- **Property tax districts**—Areas designated for the collection of taxes imposed on property owners to support local government facilities and services, based on the property's location. Examples include taxes collected for districts to pay for schools, police, or other services. Property tax districts are created by local governments.
- **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 Pitney Bowes sales representative for more information.

Using the optional Pitney Bowes 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 the Enterprise Tax Module.

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.pb.com/spectrum/services/ReverseGeoTAXInfoLookup"
xmlns:spec="http://spectrum.pb.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.pb.com/"

xmlns:ns3="http://www.pb.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.

Parameter	Description
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: **Feet** Distances are measured in feet. (default) **Meters** Distances are measured in meters.
Default buffer widths	
DefaultBufferWidth	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, the Property Tax Districts (PTD) file, Place and MCD files. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. For more information about buffers, see **Buffering** on page 852.
DefaultUserBufferWidth	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 852. The default buffer width that you specify here can be overridden on a record-by-record basis using the BufferWidth input field. **Note:** To use buffers, the user-defined boundary file must support buffers.

Output Data Options

Data options control the data returned by ReverseGeoTAXInfoLookup.

Parameter	Description
GeoTAXOutputRecordType	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 checkbox, and list only those fields you want returned in `OutputFields`. • **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 You can also specify one, and only one, of the following: **I** Insurance Premium Tax District (IPD) **P** Property Tax District (PTD) **R** Payroll Tax District (PAY) **S** Special Purpose Tax District (SPD) For a description of the fields in each output group, see **Response** on page 185. **Note:** If you specify `W`, also specify `R` to obtain the best payroll system tax code match possible.
TaxKey	If you integrate ReverseGeoTAXInfo Lookup with third-party tax compliance software from Vertex or Taxware, select which vendor you use. This controls the value returned in the GeoTAXKey output field. One of the following: **N** Do not return either the Taxware or Vertex jurisdiction codes (default). **T** Return the Taxware jurisdiction code for the address. **Y** Return the Vertex jurisdiction code for the address. Select this option if you obtained a MatchMaster file from Vertex. **V** Return the Vertex jurisdiction code for the address. Select this option if you obtained a Vertex file from Pitney Bowes.
TaxRate	Select the desired Sales tax rate type or none: **N** Do not return sales tax rates. (default) **G** Return the General sales tax rates. **A** Return the Automotive sales tax rates. **C** Return the Construction sales tax rates. Return the Medical sales tax rates.

Parameter	Description
OutputFields	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. For a list of the fields included in each data type, see **Response** on page 185.

Output Format

Output format options control how ReverseGeoTAXInfo Lookup formats output data.

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

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.
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	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.txb file. For more information about buffers, see **Buffering** on page 852. **Note:** The buffer distance for Cousub.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.
MCD.DistanceToBorder	10	Returns the distance in feet between the matched address point to the polygon defined by the Cousub.txb file.
StateCode	3	Two-digit Federal Information Processing Standards (FIPS) state 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.

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 `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.
PTC*n*.MatchCode	2 per PTC	Indicates the level of match obtained for the location. 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*n*.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.
PTC*n*.PayrollDescription	41 per PTC	A description of the purpose of this payroll code.

Response Element	Max. Field Length with null terminator	Description
PTC*n*.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	The value in this field varies depending on the option you specified in the `TaxKey` option: If you specified `T`, GeoTAXKey contains the proprietary codes used in Taxware tax compliance software. You can use this code in your Taxware application to find out the tax rate for the jurisdiction. If you specified `Y` or `V`, GeoTAXKey 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.

Response Element	Max. Field Length with null terminator	Description
GeoTAXKey.MatchCode	2	Return code denoting the level of match obtained against the Vertex or Taxware cross reference files. **E** Exact match using five fields: FIPS state code, FIPS county code, FIPS or GNIS place code, ZIP Code, and FIPS place name. **P** Partial match using four fields: FIPS state code, FIPS county code, FIPS or GNIS place code, and ZIP Code. **A** Alternate match using two fields: ZIP Code, FIPS place name. **N** Record is default coded based on valid state code. **null** No matching record found.
GeoTAXKey.MatchLevel	12	A description of the value returned in the GeoTAXKey.MatchCode field. **Exact** Exact match. See description in GeoTAXKey.MatchCode. **Partial** Partial match. See description in GeoTAXKey.MatchCode. **Alternate** Alternate match. See description in GeoTAXKey.MatchCode. **DefaultCode** Record is default coded. See description in GeoTAXKey.MatchCode. **NoMatch** No matching record found.
GNISCode	10	Unique nine-digit Geographic Names Information System (GNIS) code. **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
Place.ClassCode	3	Place class code. Place class codes are used to determine the proper taxing jurisdictions **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output. **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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.

Response Element	Max. Field Length with null terminator	Description
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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.LastVerifiedDate	8	Last verified date, in the format MM/YYYY, representing the month and year that TomTom verified municipality change information. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.Name	41	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. **Note:** This field is always included in the output regardless of whether or not you choose to include tax jurisdiction data in the output.
Place.PointStatus	2	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 852. **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.
Place.DistanceToBorder	10	Returns the distance in feet between the matched address point to the polygon defined by the Place.txb file.

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*n*.BoundaryDescription	51 per User Boundary	A description of the polygon.
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	Indicates where in the polygon the location resides in relation to the edge of the area. Buffer width is specified by the option `DefaultUserBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
UserBoundary*n*.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 179. To include IPD data in the output, set `GeoTAXOutputRecordType` = `I`.

> **Note:** ReverseGeoTAXInfoLookup returns multiple districts for IPDs, SPDs, PTDs, 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	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
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
IPD*n*.DistrictID	11 per IPD	IPD ID.
IPD*n*.DistrictName	61 per IPD	IPD name.
IPD*n*.DistrictType	7 per IPD	IPD district type.
IPD*n*.UpdateDate	7 per IPD	IPD update date (MMYYYY).
IPD*n*.VersionDate	7 per IPD	IPD compiled date (MMYYYY).
IPD*n*.Notes	21 per IPD	Tax code descriptions. For example: 01, 33, A, B
IPD*n*.ChangeDate	7 per IPD	IPD change date.
IPD*n*.EffectiveDate	7 per IPD	MMDDYY - Identifies when district becomes active - State supplied For example: 010108
IPD*n*.ExpirationDate	7 per IPD	MMDDYY - Identifies when district becomes inactive - State supplied For example: 063009
IPD*n*.FireRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.CasualtyRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.VehicleRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MarineRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.HealthRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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
IPD*n*.LifeRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.OtherRate	21 per IPD	Format is dependent on associated flag Possible Values: .13, 15.00 or 3;7
IPD*n*.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%"
IPD*n*.MinimumRate	21 per IPD	Format is dependent on associated flag For example: .13, 15.00 or 3;7
IPD*n*.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, PTDs, and PAYs.

Response Element	Max. Field Length with null terminator	Description
NumberPAYsFound	3	Number of payroll tax districts found for the location.
PAY*n*.BoundaryBuffer.BufferRelation	2 per PAY	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
PAY*n*.BoundaryBuffer.DistanceToBorder	10 per PAY	Indicates the distance in feet from the location to the border of the district.
PAY*n*.DistrictID	11 per PAY	PAY district ID.
PAY*n*.DistrictName	61 per PAY	PAY district name.
PAY*n*.DistrictType	7 per PAY	PAY district type.
PAY*n*.ID	11 per PAY	PAY ID.

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

Property Tax District

The following table lists the output fields that contain Property Tax District (PTD) data. For more information on property tax districts, see **ReverseGeoTAXInfoLookup** on page 179. To include this data in the output, set `GeoTAXOutputRecordType` = `P`.

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

Response Element	Max. Field Length with null terminator	Description
NumberPTDsFound	3	The number of Property Tax Districts found for the location.
PTD*n*.BoundaryBuffer.BufferRelation	2 per PTD	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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 address is in the buffer area. For more information, see **Buffering** on page 852.
PTD*n*.BoundaryBuffer.DistanceToBorder	10 per PTD	Indicates the distance in feet from the location to the border of the district.
PTD*n*.DistrictID	11 per PTD	PTD district ID.
PTD*n*.DistrictName	61 per PTD	PTD district name.

Response Element	Max. Field Length with null terminator	Description
PTD*n*.DistrictType	2 per PTD	PTD district type. Only returned for Indiana. **R** Reporting district **B** Billing district
PTD*n*.JurisdictionID	11 per PTD	PTD jurisdiction ID.
PTD*n*.UpdateDate	7 per PTD	PTD update date.

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, PTDs, and PAYs.

Response Element	Max. Field Length with null terminator	Description
NumberSPDsFound	3	Number of Special Purpose Tax Districts found for the location.

Response Element	Max. Field Length with null terminator	Description
SPD*n*.BoundaryBuffer.BufferRelation	2 per SPD	Indicates where in the district the location resides in relation to the edge of the district. Buffer width is specified by the option `DefaultBufferWidth` or by the input field `BufferWidth`. One of the following: **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. For more information, see **Buffering** on page 852.
SPD*n*.BoundaryBuffer.DistanceToBorder	10 per SPD	Indicates the distance in feet from the address to the border of the district.
SPD*n*.CompiledDate	7 per SPD	SPD compiled date.
SPD*n*.DistrictCode	4 per SPD	3-digit district type code.
SPD*n*.DistrictName	61 per SPD	SPD name.
SPD*n*.DistrictNumber	6 per SPD	SPD district number.
SPD*n*.EffectiveDate	7 per SPD	SPD effective date.
SPD*n*.UpdateDate	7 per SPD	SPD update date.
SPD*n*.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 Pitney Bowes 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 Pitney Bowes 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.

Response Element	Max. Field Length with null terminator	Description
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).
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	This field contains a return code if the GeoTAX engine experiences an abnormal termination. **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. The first character indicates the file (or set of files affected). **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 The second position is one of the following: **E** Fatal issue, program terminating **F** Expired database **I** Informational

Response Element	Max. Field Length with null terminator	Description
GTX.ErrorDescription	81	

Response Element	Max. Field Length with null terminator	Description
		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. **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. 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" 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" 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 = "

Response Element	Max. Field Length with null terminator	Description
		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 `WN` 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.
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 624). 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.
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 620.

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

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

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

Extended 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 617).

> **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 619
> - 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 `AddressLine2`.
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.

Input Addressline	Output Addressline	Extended Code	Description
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.
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 AddressLine2.
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 AddressLine2.

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.

Code	In 3rd hex position means:
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 `AddressLine2`.
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 `AddressLine2`.
4	Unit Number changed or ignored.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.
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 `AddressLine2`.

Code	In 3rd hex position means:
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 `AddressLine2`.

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.

Code	"nnn" values	Description
	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.
	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 the Enterprise Tax Module 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 the Enterprise Tax Module did not standardize the address; therefore, the Enterprise Tax Module 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.

1st character	Always an "A" indicating an address location.		
2nd character	May be one of the following:		
		C	Interpolated address point location
		G	Auxiliary file data location
		I	Application infers the correct segment from the candidate records
		P	Point-level data location
		R	Location represents a ranged address

	S	Location on a street range
	X	Location on an intersection of two streets
3rd and 4th character	Digit indicating other qualities about the location.	

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:

Code	Description
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.
nn = 05	Structure centroid Indicates the center of a building footprint polygon, where the building receives mail or has telephone service. Usually a residential address consists of a single building. For houses with outbuildings (detached garages, shed, barns, etc.), only the residences have a structure point. Condominiums and duplexes have multiple points for each building. Larger buildings, such as apartment complexes, typically receive mail at one address for each building and therefore individual apartments are not represented as discrete structure points. Shopping malls, industrial complexes, and academic or medical center campuses where one building accepts mail for the entire complex are represented as one point. When addresses are assigned to multiple buildings within one complex, each addressed structure is represented by a point. If the center of a structure falls outside of its polygon, the center is manually repositioned to fall inside the polygon.
nn = 07	Manually placed Address points are manually placed to coincide with the midpoint of a parcel's street frontage at a distance from the center line.

Code	Description
nn = 08	Front door point Represents the designated primary entrance to a building. If a building has multiple entrances and there is no designated primary entrance or the primary entrance cannot readily be determined, the primary entrance is chosen based on proximity to the main access street and availability of parking.
nn = 09	Driveway offset point Represents a point located on the primary access road (most commonly a driveway) at a perpendicular distance of between 33-98 feet (10-30 meters) from the main roadway.
nn = 10	Street access point Represents the primary point of access from the street network. This address point type is located where the driveway or other access road intersects the main roadway.
nn = 21	Base parcel point When unable to match to an input unit number, or when the unit number is missing from an address location with multiple units, the "base" parcel information is returned, the address is not standardized to a unit number, and additional information, such as an Assessor's Parcel Number, is not returned.
nn = 22	Backfill address point The precise parcel centroid is unknown. The address location assigned is based on two known parcel centroids.
nn = 23	Virtual address point The precise parcel centroid is unknown. The address location assigned is relative to a known parcel centroid and a street segment end point.

Code	Description
nn = 24	Interpolated address point The precise parcel centroid is unknown. The address location assigned is based on street segment end points.
AIn	The correct segment is inferred from the candidate records at match time.
ASn	House range address geocode. This is the most accurate street interpolated geocode available.
AIn, ASn and ACn share the same values for the 3rd 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	Indicates one or both of the following: • The address is interpolated onto a TIGER segment that did not initially contain address ranges. • The original segment name changed to match the USPS spelling. This specifically refers to street type, predirectional, and postdirectional. **Note:** Only the second case is valid for non-TIGER data because segment range interpolation is only completed for TIGER data.
n = 3	Both 1 and 2.
n = 7	Placeholder. Used when starting and ending points of segments contain the same value and shape data is not available.

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

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

The table below contains the values and descriptions for the 2nd - 3rd characters in the street centroid location codes.

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

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

ZIP + 4 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.

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

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

Character Position	Code	Description
2nd Character		
	B	Block Group accuracy (most accurate).
	T	Census Tract accuracy.

Character Position	Code	Description
	C	Unclassified Census accuracy. Normally accurate to at least the County level.
3rd Character		
	5	Location of the Post Office that delivers mail to the address, a 5-digit ZIP Code centroid, or a location based upon locale (city). See the 4th character for a precise indication of locational accuracy.
	7	Location based upon a ZIP + 2 centroid. These locations can represent a multiple block area in urban locations, or a slightly larger area in rural settings.
	9	Location based upon a ZIP + 4 centroid. These are the most accurate centroids and normally place the location on the correct block face. For a small number of records, the location may be the middle of the entire street on which the ZIP + 4 falls. See the 4th character for a precise indication of locational accuracy.
4th Character		
	A	Address matched to a single segment. Location assigned in the middle of the matched street segment, offset to the proper side of the street.

Character Position	Code	Description
	a	Address matched to a single segment, but the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	B	Address matched to multiple segments, all segments have the same Block Group. Location assigned to the middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	b	Same as methodology B except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.
	C	Address matched to multiple segments, with all segments having the same Census Tract. Returns the Block Group representing the most households in this ZIP + 4. Location assigned to t he middle of the matched street segment with the most house number ranges within this ZIP + 4. Location offset to the proper side of the street.
	c	Same as methodology C except the correct side of the street is unknown. Location assigned in the middle of the matched street segment, offset to the left side of the street, as address ranges increase.

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

Character Position	Code	Description
	I	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, and b. All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.
	J	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, b, C, and c. All centroids in this ZIP + 2 cluster have the same Census Tract. Location assigned to the ZIP + 2 centroid.
	K	Created ZIP + 2 cluster centroid as defined by methodologies A, a, B, b, C, c, D, and d. Location assigned to the ZIP + 2 centroid.
	L	Created ZIP + 2 cluster centroid as defined by methodology E. All centroids in this ZIP + 2 cluster have the same Block Group. Location assigned to the ZIP + 2 centroid.
	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.

Character Position	Code	Description
	V	Over 95% of addresses in this ZIP Code are in a single Census Tract. Location assigned to the ZIP Code centroid.
	W	Over 80% of addresses in this ZIP Code are in a single Census Tract. Reasonable Census Tract accuracy. Location assigned to the ZIP Code centroid.
	X	Less than 80% of addresses in this ZIP Code are in a single Census Tract. Census ID is uncertain. Location assigned to the ZIP Code centroid.
	Y	Rural or sparsely populated area. Census code is uncertain. Location based upon the USGS places file.
	Z	P.O. Box or General Delivery addresses. Census code is uncertain. Location based upon the Post Office location that delivers the mail to that address.

Geographic Centroid Location Codes

Location codes that begin with "G" are geographic centroid location codes. Geographic centroids may be returned if the 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.

1st character	Always "G" indicating a location derived from a geographic centroid.

2nd character	Geographic area type. One of the following: **M** Municipality (for example, a city) **C** County **S** State

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)**
- **Property Tax Districts (PTD)**

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

Type	Descriptions
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
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

Type	Descriptions
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
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 Distrit
DE	FIRE	Fire District
FL	FIRE	Fire District
FL	POLICE	Police District
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

State	Type	Descriptions
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
UNI	School District

Property Tax Districts (PTD)

Type	Descriptions
B	Billing
R	Reporting
X	Reporting and Billing

Class Codes

This appendix lists definitions for the FIPS Class Codes.

Class C—Incorporated Places

Class Code	Description
C1	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. 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.
C2	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. 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.
C3	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. 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.

Class Code	Description
C4	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. For example, the entity in California whose legal name is San Buenaventura (subclass C1) is commonly known as Ventura, which is coded C4.
C5	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.
C6	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.
C7	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.
C8	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.
C9	Identifies an inactive or nonfunctioning incorporated place.

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

Type	Descriptions
U1	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.
U2	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.
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.

Type	Descriptions
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.

GeoConfidence Module

GeoConfidenceSurface

GeoConfidenceSurface returns geoconfidence polygons (also called surfaces) based on the quality of the geocode information generated by the Enterprise Geocoding Module. 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 GeoConfidence Module's FloodZoneAnalysis dataflow template.

> **Note:** GeoConfidence uses services provided by the Enterprise Geocoding and Location Intelligence 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	The value returned in this field indicates which geoconfidence surface type has been returned. The possible values are: **INTERSECTION** A geocode point for the intersection of two streets. **ADDRESS** An array of street segment points representing the street segment where the address is located. **POINT** If the geocoder was able to match the address using point data, the point geometry where the address is located. **POSTAL1** A geocode point for the ZIP centroid. **POSTAL2** An array of points for all street segments in the ZIP + 2 in which the address is located. **POSTAL3** An array of points for street segments in the ZIP + 4 in which the address is located. **ERROR** An error has occurred.
StreetSegmentPoints	1024	An array of latitude/longitude values that represent the street segment points. **Note:** This field contains values only if the `GeoConfidenceCode` field returns a value of `ADDRESS`, `POSTAL2`, or `POSTAL3`.
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.

Global Sentry Module

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.pb.com/spectrum/services/GlobalSentry">
   <soapenv:Header/>
   <soapenv:Body>
      <glob:GlobalSentryRequest>
         <glob:options/>
         <glob:Input>
            <glob:Row>
               <glob:FirstName>Miguel</glob:FirstName>
               <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.pb.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 87: 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.
AddressLine3	The third address line. Recommended if provided.
Country	Full country name. Required if address lines are used.

Parameter	Description
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 88: 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	

Response Element	Description
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`.
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.

Response Element	Description
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	
InputIDNumber	Input ID Number from the original data source.
IDNumber	ID Number returned from database.
IDNumberScore	ID Number match score. 0-100.

Response Element	Description
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	Identifiesif 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`.

Response Element	Description
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 `Yes` or `No`.
CitizenshipProvided	Identifies if the Citizenship is provided in the input data. Values are `Yes` or `No`.
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 `Yes` or `No`.
NationalityProvided	Identifies if the Nationality is provided in the input data. Values are `Yes` or `No`.
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.

Response Element	Description
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 250.
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 89: 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

Input	No Data Provided	Matched	Did Not Match
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 the Spectrum™ Technology Platform, Universal Name, Data Normalization and Advanced Matching modules.

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

Information Extraction Module

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.pb.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.pb.com/"
xmlns:ns3="http://www.pb.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 90: 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.).
EntityList	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. **Address** **CreditCard** **Date** **Email** **HashTag** **ISBN** **Location** **Mention** **Organization** **Person** **Phone** **ProperNouns** **SSN** **WebAddress** **ZipCode**
OutputEntityCount	Specifies whether to return a count of how many times a particular entity occurred in the output. **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.

Location Intelligence Module

Where to Find Documentation?

The Location Intelligence Module 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

To learn about the Location Intelligence Module services, see the *Spectrum Spatial Guide* on **support.pb.com**.

Universal Addressing Module

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.pb.com/spectrum/services/AutoCompleteLoqate"
xmlns:spec="http://spectrum.pb.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.pb.com/"
 xmlns:ns3="http://www.pb.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 91: 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	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
FirmName	The company or firm name.
PostalCode	The postal code for the address.
StateProvince	The state or province.

Parameters for Options

Table 92: 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	Specifies the casing of the output data. One of the following: **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

Parameter	Description
HomeCountry	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: 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
Option.OutputCountryFormat	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". **E** Use English country names (default). **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
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.
MaximumResults	The maximum number of addresses that AutoCompleteLoqate should return. The default is 10.
isPowersearchEnable	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. **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. 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. Powersearch cannot be used for the following countries: Botswana, Ethiopia, India, Kazakhstan, Malaysia, Mongolia, Saint Kitts and Nevis, and San Marino. **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.

Parameter	Description
IsDuplicateHandlingMaskEnable	Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options: **S** Selected by default. Pre-process the input and remove duplicates that occur in a single field. **C** Selected by default. Pre-process the input and remove duplicates across all fields. **T** Pre-process the input and remove duplicates in fields that are not standard address fields. **F** Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.
FailJobOnDataLicenseError	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

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 93: 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 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
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 state or province abbreviation.
Status	Reports the success or failure of the match attempt. **null** Success **F** Failure
Status.Code	The reason for failure, if there is one. • DisabledCoder • RequestFailed • NoLookupAddressFound

Response Element	Description	
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. AutoCompleteLoqate returns data only if multiple possible matches were found.
	Not able to look up the address pattern	AutoCompleteLoqate is not able to process the partial address.

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 the Universal Addressing Module.

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.pb.com/spectrum/services/GetCandidateAddresses"
xmlns:spec="http://spectrum.pb.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.pb.com/"
xmlns:ns3="http://www.pb.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 94: 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	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name • French country name • German country name • Spanish country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
FirmName	The company or firm name.
USUrbanName	U.S. address urbanization name. Used primarily for Puerto Rico addresses.

Parameters for Options

Table 95: GetCandidateAddresses Options

Parameter	Description
PerformUSProcessing	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. **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. **Y** Yes, process U.S. addresses (default). **N** No, do not process U.S. addresses.

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	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. **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. **Y** Yes, process Canadian addresses (default). **N** No, do not process Canadian addresses.
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	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. **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. **Y** Yes, process international addresses (default). **N** No, do not process international addresses.
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.
OutputCasing	Specifies the casing of the output data. One of the following: **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
MaximumResults	The maximum number of candidate addresses that GetCandidateAddresses should return. The default is 10. The maximum is 10.

Parameter	Description
OutputShortCityName	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. **Y** Yes, return the short city name. **N** No, do not return the short city name.
DualAddressLogic	(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 314. **N** (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority: **1.** PO Box **2.** Firm **3.** Highrise **4.** Street **5.** Rural Route **6.** General Delivery **S** Return a street match, regardless of the address line. **P** Return a PO Box match, regardless of the address line.
StreetMatchingStrictness	The strictness of the street name match (U.S. addresses only). **E** The input street name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
FirmMatchingStrictness	The strictness of the firm name match (U.S. addresses only). **E** The input firm name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."

Parameter	Description
DirectionalMatchingStrictness	The strictness of the directional match. **E** The input directional must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
PerformESM	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. **Y** Yes, perform ESM processing. **N** No, do not perform ESM processing (default).
AddressLineSearchOnFail	Specifies whether ValidateAddress will search address lines for the city, state/province, and postal code. 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. Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields. 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). **Y** Yes, search the address line fields (default). **N** No, do not search the AddressLine fields.

Response

GetCandidateAddresses returns the following output.

Table 96: 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.
City	The city name.
Country	The three-character ISO 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
FirmName	The firm name.
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	Indicates the numbering scheme for the house numbers between HouseNumberLow and HouseNumberHigh, as follows: **E** Only even values **O** Only odd values **B** Both

Response Element	Description
MatchLevel	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: **A** The candidate matches the input address at the street level. **B** The candidate matches the input address at the state/province level.
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	The type of address record, as defined by U.S. and Canadian postal authorities (U.S. and Canadian addresses only): • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal
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 state or province abbreviation.
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: • DisabledCoder • RequestFailed

Response Element	Description
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.
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: **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 the Universal Addressing Module.

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.pb.com/spectrum/services/GetCandidateAddressesLoqate"
xmlns:spec="http://spectrum.pb.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.pb.com/"

xmlns:ns3="http://www.pb.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 97: 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	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855. **Note:** This field is required. If this field is blank, no output will be returned.
FirmName	The company or firm name.
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 98: 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.

Parameter	Description
OutputCasing	Specifies the casing of the output data. One of the following: **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: **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	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: 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
Option.OutputCountryFormat	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". **E** Use English country names (default). **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
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.
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 99: 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 3116-1 Alpha-3 code for the country. For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.

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 state or province abbreviation.
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 the Universal Addressing Module.

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.pb.com/spectrum/services/GetCityStateProvince"
xmlns:spec="http://spectrum.pb.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.pb.com/"
xmlns:ns3="http://www.pb.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 100: 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 101: GetCityStateProvince Options

Parameter Name	Description
PerformUSProcessing	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. **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. **Y** Yes, process U.S. addresses (default). **N** No, do not process U.S. addresses.
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	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. **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. **Y** Yes, process Canadian addresses (default). **N** No, do not process Canadian addresses.
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.
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).
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 102: 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 state or province abbreviation.
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: • DisabledCoder • UnrecognizedPostalCode

Response Element	Description	
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 Universal Addressing Module.

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.pb.com/spectrum/services/GetCityStateProvinceLoqate"
xmlns:spec="http://spectrum.pb.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.pb.com/"

xmlns:ns3="http://www.pb.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 103: GetCityStateProvinceLoqate Input

Parameter	Description
Country	The country code or name, in any of the following formats: • 2-digit ISO country code • 3-digit UPU Country code • English country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
PostalCode	The postal code for the address.

Options

Table 104: 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	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 105: GetCityStateProvinceLoqate Output

Response Element	Description
City	The matched city name.
Country	The country in the format determined by what you selected in: • ISO Code • UPU Code • English
PostalCode	The input postal code.
ProcessedBy	Indicates which address coder processed the address. **LOQATE** The Loqate coder processed the address.
StateProvince	The state or province abbreviation.
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: • UnrecognizedPostalCode
Status.Description	The description of the failure. The only valid value is: **Postal code not found** This value will appear if Status.Code=UnrecognizedPostalCode.

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 Universal Addressing Module.

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.pb.com/spectrum/services/GetPostalCodes"
xmlns:spec="http://spectrum.pb.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.pb.com/"
xmlns:ns3="http://www.pb.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: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 106: GetPostalCodes Input

Parameter	Description
City	The city whose postal codes you want to look up. You may put the city and state in the City field. If you do this, you must leave the StateProvince field blank. The total length of the City and StateProvince fields cannot exceed 100 characters.
StateProvince	The state or province of the city whose postal codes you want to look up. You may also put the state in the City field instead of the StateProvince field. The total length of the City and StateProvince fields cannot exceed 100 characters.
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 107: 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 108: GetPostalCodes Output

Response Element	Description
City.Type	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. This field column is only populated if `OutputCityType=Y`. The possible values are: **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	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	Reports the success or failure of the match attempt. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. One of the following: • CountryNotSupported • UnableToLookup
Status.Description	Description of failure. • 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. ValidateAddress 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 ValidateAddress 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, ValidateAddress 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.pb.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.pb.com/"
```

```
      xmlns:ns3="http://www.pb.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 301 for important information about how address line data is processed for U.S. addresses.

Table 109: Input Format

Parameter	Format	Description
AddressLine1	String [50]	The first address line.
AddressLine2	String [50]	The second address line.
AddressLine3	String [50]	The third address line. Does not apply to Canadian addresses.

Parameter	Format	Description
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]	The country code or name, in any of the following formats: • Two-character ISO 3116-1 Alpha-2 country code • Three-character ISO 3116-1 Alpha-3 country code • English country name • French country name • German country name • Spanish country name For a list of ISO codes, see **ISO Country Codes and Module Support** on page 855.
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	For Canadian addresses only, indicates whether the address is in English or French, if the option `CanFrenchFormat=T` 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: Pitney Bowes

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 329.

Table 110: Output Data Options

Parameter	Description
OutputRecordType	Type of output record. For more than one, provide a list. **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 **Response** on page 340. 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 `OutputRecordType=A`. **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 342. Note that if you specify "E" and specify `OutputFormattedOnFail=Y`, the parsed address elements will contain the input address for addresses that could not be validated. **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 `OutputRecordType=E` and `OutputFormattedOnFail=Y` 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 345. **P** Postal data. Output addresses contain additional data for each validated address. For more information, see **Postal Data Output** on page 347. **Blank** Do not return any address data or postal data.

Parameter	Description
OutputFieldLevelReturnCodes	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 354. **N** No, do not output field-level return codes (default). **Y** Yes, output field-level return codes.

Parameter	Description
OutputFormattedOnFail	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. **Note:** This option applies only to U.S. and Canadian addresses. Formatted data will not be returned for any other address. **N** No, do not format failed addresses (default). **Y** Yes, format failed addresses. 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 ValidateAddress could validate the address, specify `OutputRecordType=I`. 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 ValidateAddress could validate the address, specify `Option.OutputRecordType=I`. 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 ValidateAddress could validate the address, select **Include standardized input address elements**. If you specify Y, you must specify "A" and/or "E" for OutputRecordType. If you specify Y, you must specify "A" and/or "E" for `Option.OutputRecordType`. If you check this option, you must select **Include a standard address** and/or **Include address line elements**.

Parameter	Description
OutputStreetNameAlias	For U.S. addresses only, specifies whether or not to use a street's alias in the output. A street alias is an alternate name for a street and 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. **N** No, do not return street name aliases in the output. **Y** Yes, return street name aliases in the output if there is an alias for the street (default).
OutputStreetNameAlias	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 typically applies only to a specific range of addresses on the street. 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: • If a preferred alias is used in input the preferred alias will always be used in output. • Changed aliases used in input are always converted to the base street name in output. This is one of three options that control how ValidateAddress handles street name aliases. The other two are `OutputPreferredAlias` and `OutputAbbreviatedAlias`. **Note:** If `OutputAbbreviatedAlias` is enabled, the abbreviated alias will always appear in the output even if you have `OutputStreetNameAlias` disabled. **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).

Parameter	Description
OutputAddressBlocks	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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 `OutputCountryFormat` does not affect the country name in the address block, it only affects the name returned in the **Country** output field. 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. **N** No, do not return address blocks. Default. **Y** Yes, return address blocks.

Parameter	Description
OutputAMAS	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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 `OutputCountryFormat` does not affect the country name in the address block, it only affects the name returned in the **Country** output field. 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. **N** No, do not return address blocks. Default. **Y** Yes, return address blocks.

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 301.

Table 111: 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 301.

Table 112: 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 301.

Table 113: 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 301.

Table 114: 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 301.

Table 115: 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 the apply to all addresses. Some of these options can be overridden for Canadian addresses. For more information, see **Canadian Address Options** on page 329.

Table 116: Default Options

Parameter	Description
OutputCasing	Specifies the casing of the output address. One of the following: **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
OutputPostalCodeSeparator	Specifies whether to use separators (spaces or hyphens) in ZIP™ Codes or Canadian postal codes. 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. **Y** Yes, use separator (default). **N** No, do not use separator. **Note:** Spaces are used in Canadian postal codes and hyphens in U.S. ZIP + 4® Codes.
OutputMultinationalCharacters	Specifies whether or not to return multinational characters, including diacritical marks such as umlauts or accents. (Not supported for U.S. addresses). **N** No, do not use multinational characters in the output (default). Only standard ASCII characters is returned. **Y** Yes, use multinational characters in the output.

Parameter	Description
KeepMultimatch	Indicates whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches. For more information, see **Returning Multiple Matches** on page 315.
StandardAddressFormat	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". Apt 10E 424 Washington Blvd Springfield MI 49423 **C** Place both primary and secondary address information in AddressLine1 (default). **S** Place the primary address information in AddressLine1 and the secondary address information in AddressLine2. **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 314.

Parameter	Description
OutputShortCityName	Specifies how to format city names that have short city name or non-mailing city name alternatives. Applies to U.S. and Canadian addresses. **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	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". **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	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: 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

Parameter	Description
DualAddressLogic	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.) **N** (Default) USPS® CASS™ regulations determine the address returned based on the following order of priority: **1.** PO Box **2.** Firm **3.** Highrise **4.** Street **5.** Rural Route **6.** General Delivery **S** Return a street match, regardless of the address line. **P** Return a PO Box match, regardless of the address line. For more information, see **About Dual Address Logic** on page 314.

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 310. 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 310. 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 310 .

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 117: Multiple Match Option

Parameter	Description
KeepMultimatch	Indicates whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches.
MaximumResults	A number between 1 and 10 that indicates the maximum number of addresses to return. The default value is 1. **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.
OutputFieldLevelReturnCodes	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.

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 301. 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 354.

U.S. Address Options

Parameter	Description
PerformUSProcessing	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. **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. **N** No, do not process U.S. addresses. **Y** Yes, process U.S. addresses. Default.
Database.US	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.
PerformLOT	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. To perform eLOT processing you must have the eLOT database installed. **N** No, do not perform Line of Travel Processing. Default. **Y** Yes, perform Line of Travel processing. For a listing of the output fields returned by this option, see **Enhanced Line of Travel Output** on page 368.
PerformRDI	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. 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. **N** No, do not perform Residential Delivery Indicator processing. Default. **Y** Yes, perform Residential Delivery Indicator processing.

Parameter	Description
PerformESM	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. **N** No, do not perform enhanced street matching. Default. **Y** Yes, perform enhanced street matching.
PerformASM	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. **N** No, do not perform all street matching. **Y** Yes, perform all street matching. Default.
PerformDPV	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). To perform DPV and CMRA processing, you must have the DPV database installed. The DPV database contains both DPV and CMRA data. **N** No, do not perform Delivery Point Validation or CMRA processing. Default. **Y** Yes, perform Delivery Point Validation and CMRA processing. For a listing of the output fields returned by this option, see **DPV and CMRA Output** on page 371.
PerformLACSLink	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. To perform LACS^{Link} processing, you must have the LACS^{Link} database installed. **N** No, do not attempt LACS^{Link} conversion. Default. **Y** Yes, attempt LACS^{Link} conversion. For a listing of the output fields returned by this option, see **LACSLink Output** on page 369

Parameter	Description
PerformEWS	The Early Warning System (EWS) uses the USPS® EWS File to validate addresses that are not in the ZIP + 4® database. To perform EWS processing, you must have the EWS database installed. If an input address matches an address in the EWS file, the following record-level result indicators are returned: • Status="F" • Status.Code="EWSFailure" • Status.Description="Address found in EWS table" **N** No, do not perform EWS processing. Default. **Y** Yes, perform EWS processing.

Parameter	Description
ExtractFirm	

Parameter	Description

Specifies whether to extract the firm name from AddressLine1 through AddressLine4 and place it in the FirmName output field. This option works in cases where the input record's FirmName field is blank and there is more than one address line.

Y	Yes, extract the firm name.
N	No, do not extract the firm name. Default.

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:

- 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,

  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes
  AddressLine4: <blank>

- 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.

Here are some examples of firm name extraction:

- In this example, AddressLine2 would get extracted into the FirmName output field

  FirmName: <blank>
  AddressLine1: 4200 Parliament Place Suite 600
  AddressLine2: International Goose Feathers inc.

- In this example, AddressLine3 would get extracted into the FirmName output field.

  FirmName: <blank>
  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes

- 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.

  FirmName: International Goose Feathers Inc.
  AddressLine1: 4200 Parliament Place
  AddressLine2: Suite 600
  AddressLine3: Pitney Bowes

- In this example, no firm name would be extracted because there is only one

Parameter	Description
	non-blank address line, which is always treated as the primary address element. **FirmName:** <blank> **AddressLine1:** 4200 Parliament Place Suite 600 • 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. **FirmName:** <blank> **AddressLine1:** 4200 Parliament Place Suite 600 **AddressLine2:** 1 Stop Software
ExtractUrb	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. **Y** Yes, extract the urbanization name. **N** No, do not extract the urbanization name. Default. 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, **AddressLine1:** A1 Calle A **AddressLine2:** **AddressLine3:** URB Alamar **AddressLine4:** Pitney Bowes

Parameter	Description
PerformSuiteLink	Specifies whether to perform Suite$^{Link™}$ processing. 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. For example, Firm Name: Pitney Bowes Address Line 1: 4200 Parliament Place Address Line 2: STE 1 Postal Code: 20706 In this case, Suite^{Link} processing would change the suite number to the correct suite number: Firm Name: Pitney Bowes Address Line 1: 4200 Parliament Place Address Line 2: **STE 600** Postal Code: 20706-1844 To perform Suite$^{Link™}$ processing, you must have the Suite$^{Link™}$ database installed. This option takes one of the following values: **N** No, do not use Suite$^{Link™}$. Default. **Y** Yes, use Suite$^{Link™}$ processing. For a listing of fields returned by this option, see **SuiteLink Output** on page 373.

Parameter	Description
OutputPreferredAlias	Specifies whether to use a street's preferred alias in the output. Street name aliases in the United States are alternative names given to sections of a street. There are four types of street name aliases: • **Preferred**—A preferred alias is the street name preferred locally. It typically applies only to a specific range of addresses on the street. • **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. • **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. The non-alias version of the street name is called the base street name. 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. This is one of three options that control how ValidateAddress handles street name aliases. The other two are OutputStreetNameAlias and OutputAbbreviatedAlias. 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. **Y** Yes, perform preferred alias processing. **N** No, do not perform preferred alias processing. Default. **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.

Parameter	Description
OutputAbbreviatedAlias	Specifies whether to use a street's abbreviated alias in the output if the output address line is longer than 31 characters. This is one of three options that control how ValidateAddress handles street name aliases. The other two are OutputStreetNameAlias and OutputPreferredAlias. **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. **Y** Yes, perform abbreviated alias processing. **N** No, do not perform abbreviated alias processing. Default. **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.
DPVDetermineNoStat	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. **N** No, do not determine "no stat" status. Default. **Y** Yes, determine "no stat" status. **Note:** You must enable DPV processing to use this option. The result is returned in the DPVNoStat field. For more information see **LACSLink Output** on page 369
DPVDetermineVacancy	Determines if the location has been unoccupied for at least 90 days. **N** No, do not determine vacancy. Default. **Y** Yes, determine vacancy. **Note:** You must enable DPV processing to use this option. The result is returned in the DPVVacant field. For more information see **LACSLink Output** on page 369
ReturnVerimove	Returns VeriMove detail data in output. **N** No, do not return VeriMove detail data. Default. **Y** Yes, return VeriMove detail data.

Parameter	Description
SuppressZplusPhantomCarrierR777	Specifies whether to supress 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: • No ZIP + 4 code is assigned • Address is not counted on the USPS Form 3553 (CASS Summary Report) • DPV Footnote of R7 is returned **N** No, do not suppress addresses with Carrier Route R777. **Y** Yes, suppress addresses with Carrier Route R777.
StreetMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input street name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
FirmMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input firm name must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium" (default). **L** The matching algorithm is "loose."
DirectionalMatchingStrictness	Specifies the algorithm to use when determining if an input address matches an address in the postal database. One of the following: **E** The input directionals, such as the "N" in 123 N Main St., must match the database exactly. **T** The matching algorithm is "tight." **M** The matching algorithm is "medium". Default. **L** The matching algorithm is "loose."

Parameter	Description
DPVSuccessfulStatusCondition	Select the match condition where a DPV result does NOT cause a record to fail. **F** Full match **P** Partial match **A** Always. Default. **Note:** You must enable DPV processing to use this option.
FailOnCMRAMatch	Treat Commercial Mail Receiving Agency (CMRA) matches as failures? **N** No, do not treat CMRA matches as failures. Default. **Y** Yes, treat CMRA matches as failures. **Note:** You must enable DPV processing to use this option.
StandardAddressPMBLine	Specifies where Private Mailbox (PMB) information is placed. **N** Do not include the PMB information in Standard Address output (default). **1** Place the PMB information in AddressLine1. If you specify 1, you must set StandardAddressFormat to either C or D. **2** Place the PMB information in AddressLine2.
PreferredCity	Specifies whether the preferred last line city name should be stored. **Z** Store the Preferred Last Line City Name from the USPS ZIP+4 File (Override City Name). **Note:** If you select this option, Validate Address generates a CASS-certified configuration and the USPS 3553 Report. **C** Store the USPS-preferred City Name from USPS City/State File. **Note:** If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report. **P** Store the Primary City Name from the USPS City/State File. **Note:** If you select this option, Validate Address does not generate a CASS-certified configuration and does not generate the USPS 3553 Report.

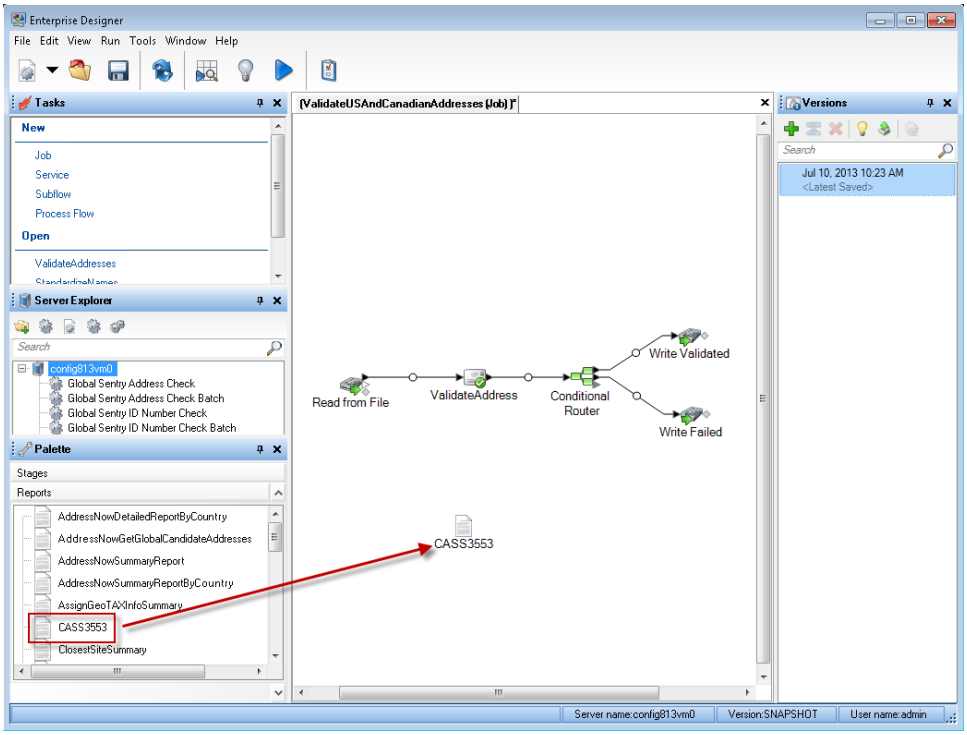
CASS Certified Processing

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 the **CASS3553** report 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** checkbox. 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	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. **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. **N** No, do not process Canadian addresses. **Y** Yes, process Canadian addresses (default).
Database.Canada	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.

Parameter	Description
CanFrenchFormat	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: English: 123 Main St W French: 123 Rue Main O The parameter controls the formatting of the address. It also affects the spelling of the directional but not spelling of the suffix. **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.
CanEnglishApartmentLabel	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 `CanStandardAddressFormat=F`. **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	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 `CanStandardAddressFormat=F`. **App** Use "App" as the label. Default. **Appartement** Use "Appartement" as the label. **Bureau** Use "Bureau" as the label. **Suite** Use "Suite" as the label. **Unite** Use "Unite" as the label.
ForceCorrectionLVR	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). **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.. **Y** Change the civic and/or suite information to match the LVR or single-single record. **Note:** If you check this box, the Statement of Address Accuracy will not be printed because this is **not** a SERP-recognized setting.
CanPreferHouseNum	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 `CanPreferHouseNum=Y`. If you do not select this option the house number is changed to match the postal code. **N** Change the house number to match the postal code. Default. **Y** Change the postal code to match the house number.

Parameter	Description
CanOutputCityAlias	Specifies whether or not to return the city alias when the alias is in the input address. This option is disabled when you specify `CanOutputCityFormat=D`. **Y** Output the city alias when the city alias is in the input. Default. **N** Never output the city alias even if it is in the input.
CanNonCivicFormat	Specifies whether or not non-civic keywords are abbreviated in the output. For example, Post Office Box vs. PO Box. **A** Abbreviate non-civic keywords. Default. **F** Do not abbreviate non-civic keywords. The full keyword is used.
EnableSERP	Specifies whether or not to use SERP options. **Y** Enable SERP options. **N** Do not enable SERP options. Default.
CanStandardAddressFormat	Specifies where to place secondary address information in the output address. Secondary address information refers to apartment numbers, suite numbers, and similar designators. **D** Place apartment information in the location specified in the. Default. **B** Place apartment information at the at the end of the AddressLine1 field. **F** Place the apartment number only (no label) at the beginning of the AddressLine1 field. For example, 400-123 Rue Main **E** Place the apartment number and label at the beginning of the AddressLine1 field. For example, Apt 400 123 Rue Main **S** Place apartment information on a separate line. **S** Place apartment information in the same location as the input address.

Parameter	Description
CanOutputCityFormat	Specifies whether to use the long, medium, or short version of the city if the city has a long name. For example, Long: BUFFALO HEAD PRAIRIE Medium: BUFFALO-HEAD-PR Short: BUFFALO-HD-PR **D** Use the default option specified by the `OutputShortCityName` option. Default. If you specify `OutputShortCityName=V`, the city is formatted as if you select for this option (see below) and`Y` for **CanOutputCityAlias**. **S** Output short city name. **L** Output the long city name. **M** Output the medium city name. **I** Use the same city format as used in the input address. Output is L, M, or S.
CanRuralRouteFormat	Specifies where to place rural route delivery information. An example of an address with rural route delivery information is: 36 GRANT RD RR 3 ANTIGONISH NS In this address, "RR 3" is the rural route delivery information. **A** Place rural route delivery information on the same line as the address, after the address information. Default. For example, 36 GRANT RD RR 3 **S** Place rural route delivery information on a separate address line. For example, 36 GRANT RD RR 3

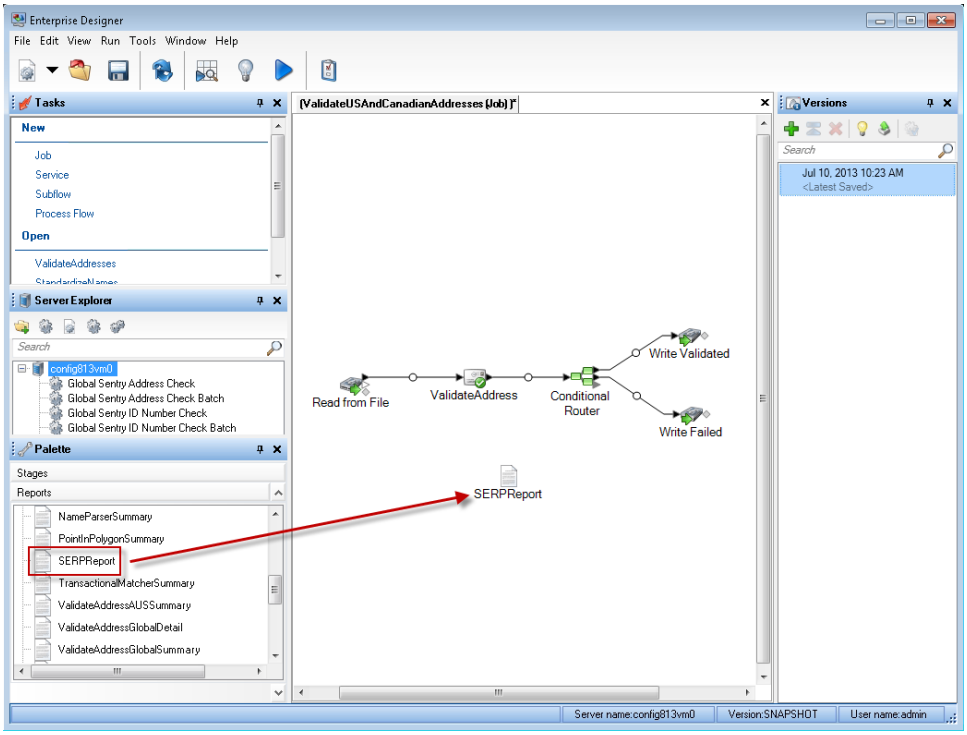
Parameter	Description
CanDeliveryOfficeFormat	Specifies where to place station information. An example of an address with station information is: PO BOX 8625 STN A ST. JOHN'S NL **I** Place station information in the same location as it is in the input address. Default. **A** Place station information on the same line as the address, after the address information. For example, PO BOX 8625 STN A **S** Place station information on a separate address line. For example, PO BOX 8625 STN A

Parameter	Description
CanDualAddressLogic	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: **D** Use DualAddressLogic Global Option. Default. **P** Match to PO Box or other non-street data. **S** Match to street. For example, given the following input address: AddressLine1: 36 GRANT RD AddressLine2: RR 4 City: ANTIGONISH StateProvince: NS ValidateAddress would return one of the following: • If `CanDualAddressLogic` is set to S, ValidateAddress returns the following: AddressLine1: 36 GRANT RD AddressLine2: RR 3 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L1 • If `CanDualAddressLogic` is set to P, ValidateAddress returns the following: AddressLine1: RR 4 City: ANTIGONISH StateProvince: NS PostalCode: B2G 2L2 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 301.

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 the 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** checkbox 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 301.

SERP return codes are provided in the following output field.

Table 118: SERP Return Code Output

Response Element	Description
CanadianSERPCode	Validation/correction return code (Canadian addresses only): **V** The input was valid. Canada Post defines a "valid" address as an address that meets all the following requirements: **Note:** There are exceptions. For further information, contact the CPC. • 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. **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. **C** The input was correctable. A "correctable" address is one that can be corrected to match one, and only one, address. **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. **F** The input address was foreign (outside of Canada).

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	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. **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. **N** No, do not process international addresses. **Y** Yes, process international addresses (default).
Database.International	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.

Parameter	Description
InternationalCityStreetSearching	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. 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. 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.
AddressLineSearchOnFail	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. Consider enabling this option if your input addresses have the city, state/province, and postal code information in the AddressLine fields. 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). **N** No, do not search the AddressLine fields. **Y** Yes, search the address line fields. Default.

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 119: Standard Address Output

Response Element	Description
AdditionalInputData	Input data not used by the address validation process. For more information, see **Additional Input Data** on page 374.
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: • 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 or province abbreviation.
USUrbanName	The validated urbanization name. (U.S. addresses only.) This is used primarily for Puerto Rico addresses.

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 345.

Table 120: Parsed Address Output

Response Element	Description
AdditionalInputData	Input data not used by ValidateAddress. For more information, see **Additional Input Data** on page 374.
AdditionalInputData.Base	Input data that was not output to the standardized address by ValidateAddress. For more information, see **Additional Input Data** on page 374.
AdditionalInputData.Unmatched	Input data passed to the matcher but not used by ValidateAddress for validation. For more information, see **Additional Input Data** on page 374.
ApartmentLabel	Apartment designator (such as STE or APT), for example: 123 E Main St **APT** 3
ApartmentLabel2	Secondary apartment designator, for example: 123 E Main St APT 3, 4th **Floor** **Note:** In this release, this field will always be blank.
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.
CanadianDeliveryInstallationAreaName	Delivery installation name (Canadian addresses only)
CanadianDeliveryInstallationQualifierName	Delivery installation qualifier (Canadian addresses only)
CanadianDeliveryInstallationType	Delivery installation type (Canadian addresses only)

Response Element	Description
City	Validated city name
Country	Country. Format is determined by what you selected in `OutputCountryFormat`: • 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.
PrivateMailbox	Private mailbox indicator.

Response Element	Description
PrivateMailbox.Type	The type of private mailbox. Possible values include: • Standard • Non-Standard **Note:** This replaces PrivateMailboxType (no period in field name). Please modify your API calls accordingly.
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 121: 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: • 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: • 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 122: Postal Data Output

Response Element	Description
CanadianSERPCode	Validation/correction return code (Canadian addresses only). For more information, see **Obtaining SERP Return Codes** on page 337.

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 309.
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: **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 309.
USCarrierRouteCode	Carrier route code (U.S. addresses only). For more information, see **Obtaining Carrier Route Codes** on page 308.
USCongressionalDistrict	Congressional district (U.S. addresses only). For more information, see **Obtaining Congressional Districts** on page 307.
USCountyName	County name (U.S. addresses only). For more information, see **Obtaining County Names** on page 308.
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 308.
USLACS	Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following: **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. **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: • 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	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: Chantilly VA 20151 Chantilly VA 20152

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 123: Record Level Indicators

Response Element	Description
AddressFormat	The type of address data being returned: **F** French format (for example: 123 Rue Main) **E** English format (for example: 123 Main St)
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 887.
CouldNotValidate	If no match was found, which address component could not be validated: • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber • RuralRoute **Note:** More than one component may be returned, in a comma-separated list.

Response Element	Description
CountryLevel	The category of address matching available. This is always "A" for U.S. and Canadian addresses. One of the following: **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: • Postal code • City name • State/county name • Street address elements • Country name **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: • Postal code • City name • State/county name • Country name **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: • 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	MatchScore provides an indication of the degree to which the output address is correct. It is significantly different from Confidence in that Confidence is 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. 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. 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. This field does not apply to Canadian addresses. 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. **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.
MultimatchCount	If multiple matches were found, indicates the number of records that are possible matches.
MultipleMatches	Indicates which address component had multiple matches, if multiple matches were found: • Firm • LeadingDirectional • PostalCode • StreetName • StreetSuffix • TrailingDirectional • Urbanization **Note:** More than one component may be returned, in a comma-separated list.

Response Element	Description
ProcessedBy	Which address coder processed the address: **USA** U.S. address coder **CAN** Canadian address coder **INT** International address coder
RecordType	Type of address record, as defined by U.S. and Canadian postal authorities (supported for U.S. and Canadian addresses only): • FirmRecord • GeneralDelivery • HighRise • PostOfficeBox • RRHighwayContract • Normal
RecordType.Default	Code indicating the "default" match: **Y** The address matches a default record. **null** The address does not match a default record.
Status	Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. For multiple matches, all possible matches is "MultipleMatchesFound." • DisabledCoder • InsufficientInputData • MultipleMatchesFound • UnableToValidate

Response Element	Description
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 `OutputFieldLevelReturnCodes=Y`. For more information, see **Output Data Options** on page 301.

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 124: Field-Level Result Indicators

Response Element	Description
AddressRecord.Result	These result codes apply to international 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. **S** — Standardized. This option includes any standard abbreviations. **U** — Unmatched. **V** — Validated. The data was confirmed correct and remained unchanged from input.

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. **D** Dropped. The field provided on input was removed. U.S. and Canadian addresses only. For more information, see **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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. **U** Unmatched. Does not apply to Canadian 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.

Response Element	Description
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.
HouseNumber.Result	**A** Appended. The field was added to a blank input field. 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 **Additional Input Data** on page 374. **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. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
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. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
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. **U** Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U. **V** Validated. The data was confirmed correct and remained unchanged from input.
PostalCodeCity.Result	These result codes apply to international 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. **P** Pass-through. The data was not used in the validation process, but it was preserved in the output. **S** Standardized. This option includes any standard abbreviations. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
PostalCode.Source	These result codes apply to U.S. addresses only. **FinanceNumber** The ZIP Code™ in the input was verified by using USPS® Finance Number groupings. **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™.
PostalCode.Type	**P** The ZIP Code™ contains only PO Box addresses. U.S. addresses only. **U** The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only. **M** The ZIP Code™ is for military addresses. U.S. addresses only. **null** The ZIP Code™ is a standard ZIP Code™.
RRHC.Result	**C** Corrected. Canadian addresses only. **D** Dropped. The field provided on input was removed. U.S. addresses only. For more information, see **Additional Input Data** on page 374. **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 rural route/highway contract is required but is missing from the input address. U.S. addresses only. **S** Standardized. This option includes any standard abbreviations. 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. and Canadian addresses only.

Response Element	Description
RRHC.Type	These result codes apply to U.S. addresses only. **HC** The address is a Highway Contract address. **RR** The address is a Rural Route address.
StateProvince.Result	**A** Appended. The field was added to a blank input field. U.S. and Canadian addresses only. **C** Corrected. U.S. 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 state 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. Does not apply to Canadian addresses. **V** Validated. The data was confirmed correct and remained unchanged from input.
Street.Result	These result codes apply to international 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. **P** Pass-through. The data was not used in the validation process, but it was preserved in the output. **R** Street corrected. House number is out of range. Applies to French, UK, and Japanese records only. **S** Standardized. This option includes any standard abbreviations. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description
StreetName.AbbreviatedAlias.Result	Indicates the result of abbreviated alias processing. One of the following: **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	This result code applies to U.S. addresses only. **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. **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	Indicates the result of preferred alias processing. One of the following: **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	**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 **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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	**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 **Additional Input Data** on page 374. **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.
USUrbanName.Result	These result codes apply to U.S. addresses only. **A** Appended. The field was added to a blank input field. **C** Corrected. **M** Multiple. The input address matched multiple records in the postal database, and each matching record has a different value in this field. **U** Unmatched. **V** Validated. The data was confirmed correct and remained unchanged from input.

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	Line of Travel sequence code and an indicator denoting USPS® LOT sequence. This field is in the format nnnnY where: **nnnn** The four-digit LOT code. **Y** One of the following: • **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.
USLOTSequence	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	Indicates whether or not the address is a candidate for LACS^{Link} conversion (U.S. addresses only). One of the following: **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. **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: • 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	Indicates the success or failure of LACS^{Link} processing. (U.S. addresses only.) **A** LACS^{Link} processing successful. Record matched through LACS^{Link} processing. **00** LACS^{Link} processing failed. No matching record found during LACS^{Link} processing. **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. **14** LACS^{Link} processing failed. Match found during LACS^{Link} processing but conversion did not occur due to other USPS® regulations. **92** LACS^{Link} processing successful. Record matched through LACS^{Link} processing. Unit number dropped on input. **null** LACS^{Link} did not process the record, or LACS^{Link} processing was not attempted.

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	Indicates the results of Delivery Point Validation (DPV) processing. **Y** DPV confirmed. Mail can be delivered to the address. **N** Mail cannot be delivered to the address. **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: 424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA **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: 424 Washington Blvd. Apt. 12 Oak Park IL 60302 USA **M** The address matches multiple valid delivery points. **U** The address could not be confirmed because the address did not code at the ZIP + 4® level. **V** The address caused a false-positive violation.
CMRA	Indicates if the address is a Commercial Mail Receiving Agency (CMRA) **Y** Yes, the address is a CMRA. **N** No, the address is not a CMRA. **U** Unconfirmed.

Response Element	Description
DPVFootnote	DPV footnote codes. **AA** Input address matched to the ZIP + 4® file. **A1** Input address not matched to the ZIP + 4® file. **BB** Input address matched to DPV (all components). **CC** Input address primary number matched to DPV but secondary number not match (present but not valid). **F1** Input address is military; DPV bypassed. **G1** Input address is general delivery; DPV bypassed. **M1** Input address primary number missing. **M3** Input address primary number invalid. **N1** Input address primary number matched to DPV but high rise address missing secondary number. **P1** Input address missing RR or HC Box number. **P3** Input address missing PO, RR, or HC Box number **RR** Input address matched to CMRA. **R1** Input address matched to CMRA but secondary number not present. **U1** Input address is unique ZIP; DPV bypassed.
DPVVacant	Indicates whether the building is vacant (unoccupied for 90 days). One of the following: **Y** Yes, the building is vacant. **N** No, the building is not vacant. **null** The `DPVDetermineVacancy` option was not turned on.
DPVNoStat	Indicates whether the building is a "no stat" building and therefore unable to receive mail. One of the following: **Y** Yes, the building is a "no stat" building, which means the building is not receiving mail. **N** No, the building is not a "no stat" building, which means the building does receive mail. **null** The `DPVDetermineNoStat` option was not turned on.

Suite[Link] Output

Response Element	Description
SuiteLinkReturnCode	Indicates whether or not ValidateAddress corrected the secondary address information (U.S. addresses only). One of the following: **A** ValidateAddress corrected the secondary address information. **00** ValidateAddress did not correct the secondary address information. **null** Suite[Link] was not performed. **XX** Suite[Link] processing encountered an error. For example, an error would occur if the Suite[Link] database is expired.
SuiteLinkMatchCode	Provides additional information on the Suite[Link] match attempt. (U.S. addresses only) **A** ValidateAddress corrected the secondary address information. **B** ValidateAddress did not correct the secondary address information. No additional detail about the match attempt is available. **C** The words in the FirmName field are all "noise" words. Noise words are defined by the USPS® and are ignored when attempting to mach 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. **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). **E** Suite[Link] processing failed because the Suite[Link] database is expired. **null** Suite[Link] was not performed or there was an error.

Response Element	Description
SuiteLinkFidelity	Indicates how well ValidateAddress matched the firm name to the firm names in the SuiteLink database. **1** The firm name matches the SuiteLink database exactly. **2** Good match. All words in the firm name except one matched the firm name in the SuiteLink database. **3** Poor match. More than one word in the firm name did not match the firm name in the SuiteLink database. **null** SuiteLink could not match the firm name, or was not performed, or there was an error.

VeriMove Output

Response Element	Description
VeriMoveDataBlock	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: **Y** Yes, return the field VeriMoveDataBlock.. **N** No, do not return the field VeriMoveDataBlock.

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 usuall 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 301 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:

Pitney Bowes
4200 Parlament
Lanham MD

ValidateAddress would correct the spelling of the street name and add the suffix, returning "4200 Parliament Pl" 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 314.

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.pb.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.pb.com/"

xmlns:ns3="http://www.pb.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>0060605060000000060</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 125: ValidateAddressGlobal Input

Parameter	Format	Description
AddressLine1 through AddressLine6	String [79]	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: **AddressLine1:** 17413 Blodgett Road **AddressLine2:** PO Box 123 **City:** Mount Vernon **StateProvice:** WA **PostalCode:** 97273 **Country:** USA If the input address is not already parsed into the appropriate address line and City, StateProvice, and PostalCode fields, use the UnformattedLine fields instead of the address line fields.
City	String [79]	City name
StateProvince	String [79]	State or province.

Parameter	Format	Description
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]	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: **UnformattedLine1:** 17413 Blodgett Road **UnformattedLine2:** PO Box 123 **UnformattedLine3:** Mount Vernon WA 97273 **UnformattedLine4:** USA This address would be parsed into these output fields: **AddressLine1:** 17413 Blodgett Road **AddressLine2:** PO Box 123 **City:** Mount Vernon **StateProvice:** WA **PostalCode:** 97273 **Country:** USA **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.

Parameters for Options

Input Options

Table 126: 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 *Spectrum™ Technology Platform Administration Guide*.
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 855.

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 855.
Input.FormatDelimiter	Enables you to use non-standard formatting for multi-line addresses in input files. Acceptable values for this field include the following: • 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) **Note:** The same value must be selected for both the input option and output option.

Output Options

Table 127: 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	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. **TRUE** Include the input data in the output. **FALSE** Do not include the input data in the output (default).

Parameter	Description
Result.StateProvinceType	Specifies the format for the StateProvince field. One of the following. **ABBREVIATION** Return the abbreviation for the state or province. For example, North Carolina would be returned as "NC". **COUNTRY_STANDARD** Return either the abbreviation or the full name depending on the format used by the country's postal authority. (Default) **EXTENDED** Return the full name of the state or province, not the abbreviation. For example "North Carolina".
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
Result.PreferredScript	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. **ASCII_Extended** — ASCII characters with expansion of special characters (e.g. Ã– = OE) **ASCII_Simplified** — ASCII characters **Database** — (default) Latin I or ASCII characters (as per reference database standard) **Latin** — Latin I characters **Latin_Alt** — Latin I characters (alternative transliteration) **Postal_Admin_Alt** — Latin I or ASCII characters (local postal administration alternative) **Postal_Admin_Pref** — Latin I or ASCII characters (as preferred by local postal administration) 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 791.
Result.PreferredLanguage	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. **DATABASE** — Language derived from reference data for each address. Default. **ENGLISH** — English locality and state/province names output, if available.
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.

Parameter	Description
Result.FormatDelimiter	Enables you to use non-standard formatting for multi-line addresses in the output. Acceptable values for this field include the following: • 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) **Note:** The same value must be selected for both the input option and output option.

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

Country	Database			Latin			
TWN	Hanzi	Hanzi	Hanzi	CHINESE_ CANTONESE	CHINESE_ MANDARIN	CHINESE_ CANTONESE + LATIN_SIMPLE	CHINESE_ CANTONESE + 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 128: ValidateAddressGlobal Process Options

Parameter	Description
Process.OptimizationLevel	Use this option to set the appropriate balance between processing speed and quality. One of the following: **NARROW** The parser will honor input assignment strictly, with the exception of separation of House Number from Street information. **STANDARD** The parser will separate address element more actively as follows: • 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 **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. 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. 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.

Parameter	Description
Process.Mode	Specifies the type of processing to perform on the addresses. One of the following: **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. **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. **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. **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. **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. **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.

Parameter	Description
Process.MatchingScope	Specifies how closely an address must match the reference data in order for the address to be validated. One of the following: **Note:** These settings may not have an effect for countries lacking the necessary level of detail in the postal reference data. **ALL** — All address elements must match. **DELIVERYPOINT_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, street, house number, and sub building. **STREET_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, City/Locality/Suburb, and street. **LOCALITY_LEVEL** — Validate Global Address must achieve a match on StateProvince, PostalCode, and City/Locality/Suburb.

Response

Address Data

Table 129: Parsed Address Elements

Response Element	Description
AddressBlock1-9	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: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882

Response Element	Description
AddressLine1-6	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: AddressLine1: 4200 PARLIAMENT PL STE 600 LastLine: LANHAM MD 20706-1882
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, e.g. 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 130: 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 855.
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 131: Result Codes

Response Element	Result Code
AddressType	For United States and Canada addresses only, the AddressType field indicates the type of address. One of the following: **F** The address was validated/corrected to the firm name. **B** The address was validated/corrected to the building name. **G** The address is a general delivery address. **H** The address was validated/corrected to the high-rise default. **L** The address is a large volume receiver. **M** The address is a military address. **P** The address was validated/corrected to PO box. **R** The address was validated/corrected to a rural route. **S** The address was validated/corrected to a street address. **U** The address could not be validated/corrected so the type is unknown.
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.
CountOverflow	Indicates whether the number of candidate addresses exceeds the number returned. One of the following: **Yes** Yes, there are additional candidate addresses. To obtain the additional candidates, increase the value. **No** No, there are no additional candidates.
ElementInputStatus	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 805.
ElementRelevance	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 805.

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 805.
MailabilityScore	An estimate of how likely it is that mail sent to the address would be successful delivered. One of the following: **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 388.
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 (e.g. 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 (e.g. 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—StateProvice
- 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 (e.g. 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 Universal Addressing Module 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 the Universal Addressing Module.

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.pb.com/spectrum/services/ValidateAddressLoqate"
xmlns:spec="http://spectrum.pb.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.pb.com/"

xmlns:ns3="http://www.pb.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 132: 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	The country code or name, in any of the following formats: • Two-character ISO 3116-1 Alpha-2 country code • Three-character ISO 3116-1 Alpha-3 country code • English country name See **ISO Country Codes and Module Support** on page 855 for a list of ISO codes.
FirmName	String	The company or firm name.
PostalCode	String	The postal code for the address in one of these formats: 99999 99999-9999 A9A9A9 A9A 9A9 9999 999
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: Pitney Bowes

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 ValidateAddress Loqate.

Table 133: 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 423. **N** No, do not output field-level return codes (default). **Y** Yes, output field-level return codes.

Parameter	Description
OutputFormattedOnFail	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. **N** No, do not format failed addresses (default). **Y** Yes, format failed addresses. 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**. If you check this option, you must select **Include a standard address** and/or **Include address line elements**. 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**. If you specify Y, you must specify "A" and/or "E" for OutputRecordType. 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**.

Parameter	Description
OutputAddressBlocks	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**. For example, this input address: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882 AddressBlock3: UNITED STATES OF AMERICA 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. One of the following:

Parameter	Description
AmasFormatting	Specifies that output address data is to be formatted using Address Matching Approval System (AMAS) conventions. 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. This option modifies the output data as follows. • 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. Valid values are: **N** No, do not format the output data using AMAS conventions (default). **Y** Yes, format the output data using AMAS conventions. **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.
OutputCasing	Specifies the casing of the output data. One of the following: **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

Parameter	Description
HomeCountry	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: 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

Parameter	Description
OutputCountryFormat	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". **E** Use English country names (default). **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.
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.

Parameter	Description
Acceptance level AcceptanceLevel	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": • **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 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 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 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 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 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 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 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.

Parameter	Description
IsDuplicateHandlingMaskEnable	Enables the duplicate handling mask and specifies how duplicate records are processed and removed. Select one or more of the following options: **S** Selected by default. Pre-process the input and remove duplicates that occur in a single field. **C** Selected by default. Pre-process the input and remove duplicates across all fields. **T** Pre-process the input and remove duplicates in fields that are not standard address fields. **F** Selected by default. Post-process the output from verification and remove duplicates from non-verified fields.
MinimumMatchScore	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.
KeepMultimatch	Specifies whether or not to return multiple address for those input addresses that have more than one possible match. **Y** Yes, return multiple matches (default). **N** No, do not return multiple matches. For more information, see **Returning Multiple Matches** on page 414.
FailMultipleMatches	Fails multiple addresses for those input addresses that have more than one possible match.

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 134: 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 and is that a multiple match will return a failure if, whereas a multiple match 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 406. To identify which records are the candidate addresses, look for multiple "M" values in the field-level result indicators. For more information**Result Indicators** on page 423, see .

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 135: Standard Address Output

Response Element	Description
AdditionalInputData	Input data that could not be matched to a particular address component. For more information, see **About Additional Input Data**.
AddressLine1-4	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: AddressLine1 through AddressLine4.
City	The validated city name.
Country	The country in the format determined by what you selected in : • ISO Code • UPU Code • English
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.
PostalCode.Base	The 5-digit ZIP Code™; for example 20706.

Response Element	Description
StateProvince	The validated state or province abbreviation.

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 419.

Table 136: Parsed Address Output

Response Element	Description
AddressBlock1-9	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: AddressLine1: 4200 Parliament Place AddressLine2: Suite 600 City: Lanham StateProvince: MD PostalCode: 20706 Results in this address block output: AddressBlock1: 4200 PARLIAMENT PL STE 600 AddressBlock2: LANHAM MD 20706-1882
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**

Response Element	Description
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 : • 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.
PostalCode	Validated postal code. For U.S. addresses, this is the ZIP Code.
Principality *	The largest geographic data element within a country

Response Element	Description
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 137: 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 : • 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.
Principality.Input *	The largest geographic data element within a country
StateProvince.Input	Validated state or province name

Response Element	Description
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 138: Geocode Address Output

Response Element	Description
Geocode.MatchCode	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: **A** Multiple candidate geocodes were found to match the input address, and an average of these was returned **I** A geocode was able to be interpolated from the input addresses location in a range **P** A single geocode was found matching the input address **U** A geocode was not able to be generated for the input address The second byte represents the level of geocoding matching and is one of the following: **5** Delivery point (post box or subbuilding) **4** Premise or building **3** Thoroughfare **2** Locality **1** Administrative area **0** None
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 139: City/Street/Postal Code Centroid Match Codes

Element	Match Code
Address Point	P4

Element	Match Code
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 140: 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 Loqate Confidence Algorithm**.

Response Element	Description
CouldNotValidate	If no match was found, which address component could not be validated: • ApartmentNumber • HouseNumber • StreetName • PostalCode • City • Directional • StreetSuffix • Firm • POBoxNumber **Note:** More than one component may be returned, in a comma-separated list.
MatchScore	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. 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. **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.
ProcessedBy	Which address coder processed the address: **LOQATE** The Loqate coder processed the address.

Response Element	Description
Status	Reports the success or failure of the match attempt. For multiple matches, this field is "F" for all the possible matches. **null** Success **F** Failure
Status.Code	Reason for failure, if there is one. • UnableToValidate
Status.Description	Description of the problem, if there is one. **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 141: 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.
	U	Unmatched. Does not apply to Canadian 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.
	U	Unmatched.
	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.
	U	Unmatched.
	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.
	U	Unmatched. For example, if the street name does not match the postal code, both StreetName.Result and PostalCode.Result will contain U.
	V	Validated. The data was confirmed correct and remained unchanged from input.

Response Element	Description	
PostalCode.Type	**P**	The ZIP Code™ contains only PO Box addresses. U.S. addresses only.
	U	The ZIP Code™ is a unique ZIP Code™ assigned to a specific company or location. U.S. addresses only.
	M	The ZIP Code™ is for military addresses. U.S. addresses only.
	null	The ZIP Code™ is a standard ZIP Code™.
Principality.Result *	The largest geographic data element within a country	
StateProvince.Result	**A**	Appended. The field was added to a blank input field. U.S. and Canadian addresses only.
	C	Corrected. U.S. 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. U.S. and Canadian addresses only.
	R	The state 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. Does not apply to Canadian addresses.
	V	Validated. The data was confirmed correct and remained unchanged from input.
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's output. An example of an AVC is:

V44-I44-P6-100

AVCs consist of 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 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 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 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 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 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 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 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 ValidateAddressAUS.

Table 142: Output Fields

Response Element	Description
Barcode	Standard barcode based on the DPID. **F** Failure (no barcode found) **20-digit number** Success
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

Response Element	Description
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

Universal Name Module

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 multi-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.pb.com/spectrum/services/OpenNameParser"
xmlns:spec="http://spectrum.pb.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.pb.com/"
xmlns:ns3="http://www.pb.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:output_port>
      </ns3:OpenNameParserResponse>
   </soap:Body>
</soap:Envelope>
```

Request

Parameters for Input Data

Table 143: Open Name Parser Input

Parameter	Description
CultureCode	The culture of the input name data. The options are listed below. **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 are also valid.
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 144: Open Name Parser Parsing Options

Parameter	Description
ParseNaturalOrderPersonalNames	Specifies whether to parse names where the is in the order Title, First Name, Middle Name, Last Name, and Suffix. **true** Parse personal names that are in natural order. **false** Do not parse names that are in natural order.

Parameter	Description
ParseReverseOrderPersonalNames	Specifies whether to parse names where the last name is specified first. **true** Parse personal names that are in reverse order. **false** Do not parse names that are in reverse order.
ParseConjoinedNames	Specifies whether to parse conjoined names. **true** Parse conjoined names. **false** Do not parse conjoined names.
SplitConjoinedNames	Specifies whether to separate names containing more than one individual into multiple records, for example, `Bill & Sally Smith`. **true** Split conjoined names. **false** Do not split conjoined names.
ParseBusinessNames	Specifies whether to parse business names. **true** Parse business names. **false** Do not parse business names.
OutputAsList	Specifies whether to return the parsed name elements in a list form. **true** Return the parsed elements in a list form. **false** Do not return the parsed elements in a list form.
ShortcutThreshold	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. Specify a value from 0 to 100. The default is 100.

Parameters for Culture Options

The following table lists the options that control name cultures.

Table 145: Open Name Parser Cultures Options

Parameter	Description
DefaultCulture	Specifies which culture(s) you want to include in the parsing grammar. Global Culture is the default selection. 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: `es,ja,,`

Parameters for Advanced Options

The following table lists the advanced options for name parsing.

Table 146: Open Name Parser Advanced Options

Option	Description
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.
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	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	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. 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.
BusinessNamesDomain	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.
BusinessNamesPriority	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. 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.

Response

Table 147: 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, "Pitney Bowes".
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. **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.

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 851
Country Codes 854
Validate Address Confidence Algorithm 886

A - Buffering

In this section

Buffering 852

Buffering

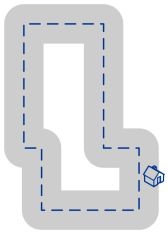
Use buffering to define areas that are close to the edges of a polygon, line, or point.

Buffered Polygon (zone)

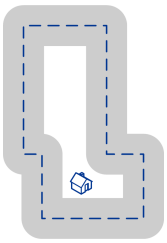
Buffered Line (corridor)

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.

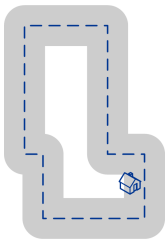


The buffer area extends on both sides of the boundary (inside and outside). When you use buffering, the output field BufferRelation indicates whether or not the point is in the buffered zone, and whether the point is inside or outside of the polygon, as shown in the following illustrations.



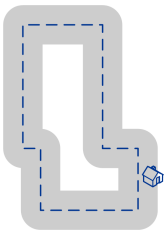
The point is inside the polygon and not in the buffer area.

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 855

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 Enterprise Geocoding Module 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 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Afghanistan	AF	AFG	Universal Addressing Module
Aland Islands	AX	ALA	Universal Addressing Module
Albania	AL or SQ (Routing)	ALB	Universal Addressing Module Enterprise Geocoding Module Enterprise Routing Module
Algeria	DZ	DZA	Enterprise Geocoding Module (Africa) Universal Addressing Module
American Samoa	AS	ASM	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Andorra	AD	AND	Enterprise Geocoding Module. (Andorra is covered by the Spain geocoder) Universal Addressing Module GeoComplete Module
Angola	AO	AGO	Enterprise Geocoding Module (Africa) Universal Addressing Module
Anguilla	AI	AIA	Universal Addressing Module
Antarctica	AQ	ATA	Universal Addressing Module
Antigua And Barbuda	AG	ATG	Universal Addressing Module
Argentina	AR	ARG	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module
Armenia	AM	ARM	Universal Addressing Module
Aruba	AW	ABW	Enterprise Geocoding Module (Latin America) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Australia	AU	AUS	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Austria	AT	AUT	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Azerbaijan	AZ	AZE	Universal Addressing Module
Bahamas	BS	BHS	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module
Bahrain	BH	BHR	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Bangladesh	BD	BGD	Universal Addressing Module
Barbados	BB	BRB	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Belarus	BY	BLR	Universal Addressing Module Enterprise Routing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Belgium	BE	BEL	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Belize	BZ	BLZ	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Benin	BJ	BEN	Enterprise Geocoding Module (Africa) Universal Addressing Module
Bermuda	BM	BMU	Universal Addressing Module Enterprise Routing Module
Bhutan	BT	BTN	Universal Addressing Module
Bolivia, Plurinational State Of	BO	BOL	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Bonaire, Saint Eustatius And Saba	BQ	BES	Universal Addressing Module
Bosnia And Herzegovina	BA	BIH	Universal Addressing Module Enterprise Routing Module Enterprise Geocoding Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Botswana	BW	BWA	Enterprise Geocoding Module (Africa) Universal Addressing Module
Bouvet Island	BV	BVT	Universal Addressing Module
Brazil	BR	BRA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
British Indian Ocean Territory	IO	IOT	Universal Addressing Module
Brunei Darussalam	BN	BRN	Enterprise Geocoding Module Universal Addressing Module
Bulgaria	BG	BGR	Universal Addressing Module
Burkina Faso	BF	BFA	Enterprise Geocoding Module (Africa) Universal Addressing Module
Burundi	BI	BDI	Enterprise Geocoding Module (Africa) Universal Addressing Module
Cambodia	KH	KHM	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Cameroon	CM	CMR	Enterprise Geocoding Module (Africa) Universal Addressing Module
Canada	CA	CAN	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Cape Verde	CV	CPV	Universal Addressing Module
Cayman Islands	KY	CYM	Universal Addressing Module
Central African Republic	CF	CAF	Universal Addressing Module
Chad	TD	TCD	Universal Addressing Module
Chile	CL	CHL	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module GeoComplete Module
China	CN or zh_CN (Routing)	CHN	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Christmas Island	CX	CXR	Universal Addressing Module
Cocos (Keeling) Islands	CC	CCK	Universal Addressing Module
Colombia	CO	COL	Universal Addressing Module
Comoros	KM	COM	Universal Addressing Module
Congo	CG	COG	Enterprise Geocoding Module (Africa) Universal Addressing Module
Congo, The Democratic Republic Of The	CD	COD	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module
Cook Islands	CK	COK	Universal Addressing Module
Costa Rica	CR	CRI	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Côte d'Ivoire	CI	CIV	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Croatia	HR	HRV	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Cuba	CU	CUB	Enterprise Geocoding Module (Latin America) Enterprise Routing Module Universal Addressing Module
Curacao	CW	CUW	Universal Addressing Module
Cyprus	CY	CYP	Enterprise Geocoding Module Universal Addressing Module
Czech Republic	CZ or CS (Routing)	CZE	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module GeoComplete Module
Denmark	DK	DNK	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Djibouti	DJ	DJI	Universal Addressing Module
Dominica	DM	DMA	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Dominican Republic	DO	DOM	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Ecuador	EC	ECU	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Egypt	EG	EGY	Enterprise Geocoding Module (Middle East) Universal Addressing Module
El Salvador	SV	SLV	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Equatorial Guinea	GQ	GNQ	Universal Addressing Module
Eritrea	ER	ERI	Universal Addressing Module
Estonia	EE	EST	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Ethiopia	ET	ETH	Universal Addressing Module
Falkland Islands (Malvinas)	FK	FLK	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Faroe Islands	FO	FRO	Universal Addressing Module
Fiji	FJ	FJI	Universal Addressing Module
Finland	FI	FIN	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
France	FR	FRA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
French Guiana	GF	GUF	Enterprise Geocoding Module (*French Guiana is covered by the France geocoder*.) Universal Addressing Module
French Polynesia	PF	PYF	Universal Addressing Module
French Southern Territories	TF	ATF	Universal Addressing Module
Gabon	GA	GAB	Enterprise Geocoding Module (Africa) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Gambia	GM	GMB	Universal Addressing Module
Georgia	GE	GEO	Universal Addressing Module
Germany	DE	DEU	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Ghana	GH	GHA	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module
Gibraltar	GI	GIB	Enterprise Geocoding Module (*Gibraltar is covered by the Spain geocoder.*) Universal Addressing Module
Greece	GR	GRC	Enterprise Geocoding Module Universal Addressing Module
Greenland	GL	GRL	Universal Addressing Module
Grenada	GD	GRD	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Guadeloupe	GP	GLP	Enterprise Geocoding Module (*Guadeloupe is covered by the France geocoder.*) Universal Addressing Module
Guam	GU	GUM	Universal Addressing Module
Guatemala	GT	GTM	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Guernsey	GG	GGY	Universal Addressing Module
Guinea	GN	GIN	Universal Addressing Module
Guinea-Bissau	GW	GNB	Universal Addressing Module
Guyana	GY	GUY	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Haiti	HT	HTI	Universal Addressing Module
Heard Island and McDonald Islands	HM	HMD	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Holy See (Vatican City State)	VA	VAT	Enterprise Geocoding Module (*The Vatican is covered by the Italy geocoder*.) Universal Addressing Module
Honduras	HN	HND	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Hong Kong	HK	HKG	Enterprise Geocoding Module Universal Addressing Module
Hungary	HU	HUN	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Iceland	IS	ISL	Enterprise Geocoding Module Universal Addressing Module
India	IN	IND	Enterprise Geocoding Module Universal Addressing Module
Indonesia	ID	IDN	Enterprise Geocoding Module Universal Addressing Module
Iran, Islamic Republic Of	IR	IRN	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Iraq	IQ	IRQ	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Ireland	IE	IRL	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Isle Of Man	IM	IMN	Universal Addressing Module
Israel	IL	ISR	Universal Addressing Module Enterprise Routing Module
Italy	IT	ITA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Jamaica	JM	JAM	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Japan	JP	JPN	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Jersey	JE	JEY	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Jordan	JO	JOR	Universal Addressing Module Enterprise Geocoding Module (Middle East) Enterprise Routing Module
Kazakhstan	KZ	KAZ	Universal Addressing Module
Kenya	KE	KEN	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module
Kiribati	KI	KIR	Universal Addressing Module
Korea, Democratic People's Republic Of	KP	PRK	Universal Addressing Module
Korea, Republic Of	KR	KOR	Universal Addressing Module
Kosovo	KS	KOS	Enterprise Geocoding Module Universal Addressing Module GeoComplete Module
Kuwait	KW	KWT	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Kyrgyzstan	KG	KGZ	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Lao People's Democratic Republic	LA	LAO	Universal Addressing Module
Latvia	LV	LVA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Lebanon	LB	LBN	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Lesotho	LS	LSO	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module
Liberia	LR	LBR	Universal Addressing Module
Libyan Arab Jamahiriya	LY	LBY	Universal Addressing Module
Liechtenstein	LI	LIE	Enterprise Geocoding Module (*Liechtenstein is covered by the Switzerland geocoder.*) Enterprise Routing Module Universal Addressing Module GeoComplete Module
Lithuania	LT	LTU	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Luxembourg	LU	LUX	Enterprise Geocoding Module (*Luxembourg is covered by the Belgium geocoder*.) Enterprise Routing Module Universal Addressing Module GeoComplete Module
Macao	MO	MAC	Enterprise Geocoding Module Universal Addressing Module
Macedonia, Former Yugoslav Republic Of	MK	MKD	Enterprise Geocoding Module Universal Addressing Module
Madagascar	MG	MDG	Universal Addressing Module
Malawi	MW	MWI	Enterprise Geocoding Module (Africa) Universal Addressing Module
Malaysia	MY	MYS	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Maldives	MV	MDV	Universal Addressing Module
Mali	ML	MLI	Enterprise Geocoding Module (Africa) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Malta	ML	MLT	Enterprise Geocoding Module Universal Addressing Module
Marshall Islands	MH	MHL	Universal Addressing Module
Martinique	MQ	MTQ	Enterprise Geocoding Module (*Martinique is covered by the France geocoder.*) Universal Addressing Module
Mauritania	MR	MRT	Enterprise Geocoding Module (Africa) Universal Addressing Module
Mauritius	MU	MUS	Enterprise Geocoding Module (Africa) Universal Addressing Module
Mayotte	YT	MYT	Enterprise Geocoding Module (*Mayotte is covered by the France geocoder.*) Universal Addressing Module
Mexico	MX	MEX	Enterprise Geocoding Module Universal Addressing Module
Micronesia, Federated States Of	FM	FSM	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Moldova, Republic Of	MD	MDA	Universal Addressing Module Enterprise Routing Module
Monaco	MC	MCO	Enterprise Geocoding Module (*Monaco is covered by the France geocoder*.) Universal Addressing Module
Mongolia	MN	MNG	Universal Addressing Module
Montenegro	ME	MNE	Enterprise Geocoding Module Universal Addressing Module
Montserrat	MS	MSR	Universal Addressing Module
Morocco	MA	MAR	Enterprise Geocoding Module (Africa) Universal Addressing Module
Mozambique	MZ	MOZ	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module
Myanmar	MM	MMR	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Namibia	NA	NAM	Enterprise Geocoding Module (Africa) Universal Addressing Module
Nauru	NR	NRU	Universal Addressing Module
Nepal	NP	NPL	Universal Addressing Module
Netherlands	NL	NLD	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
New Caledonia	NC	NCL	Universal Addressing Module
New Zealand	NZ	NZL	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Nicaragua	NI	NIC	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Niger	NE	NER	Enterprise Geocoding Module (Africa) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Nigeria	NG	NGA	Enterprise Geocoding Module (Africa) Universal Addressing Module
Niue	NU	NIU	Universal Addressing Module
Norfolk Island	NF	NFK	Universal Addressing Module
Northern Mariana Islands	MP	MNP	Universal Addressing Module
Norway	NO	NOR	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Oman	OM	OMN	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Pakistan	PK	PAK	Universal Addressing Module
Palau	PW	PLW	Universal Addressing Module
Palestinian Territory, Occupied	PS	PSE	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Panama	PA	PAN	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Papua New Guinea	PG	PNG	Universal Addressing Module
Paraguay	PY	PRY	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Peru	PE	PER	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Philippines	PH	PHL	Enterprise Geocoding Module Universal Addressing Module Enterprise Routing Module
Pitcairn	PN	PCN	Universal Addressing Module
Poland	PL	POL	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Portugal	PT	PRT	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Puerto Rico	PR	PRI	Universal Addressing Module
Qatar	QA	QAT	Enterprise Geocoding Module (Middle East) Universal Addressing Module
Reunion	RE	REU	Enterprise Geocoding Module (*Reunion is covered by the France geocoder.*) Universal Addressing Module
Romania	RO	ROU	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Russian Federation	RU	RUS	Enterprise Geocoding Module Universal Addressing Module GeoComplete Module
Rwanda	RW	RWA	Enterprise Geocoding Module (Africa) Universal Addressing Module
Saint Barthelemy	BL	BLM	Universal Addressing Module
Saint Helena, Ascension and Tristan Da Cunha	SH	SHE	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Saint Kitts and Nevis	KN	KNA	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Saint Lucia	LC	LCA	Universal Addressing Module
Saint Martin (French Part)	MF	MAF	Universal Addressing Module
Saint Pierre and Miquelon	PM	SPM	Universal Addressing Module
Saint Vincent and the Grenadines	VC	VCT	Universal Addressing Module
Samoa	WS	WSM	Universal Addressing Module
San Marino	SM	SMR	Enterprise Geocoding Module (*San Marino is covered by the Italy geocoder*.) Universal Addressing Module
Sao Tome and Principe	ST	STP	Universal Addressing Module
Saudi Arabia	SA	SAU	Enterprise Geocoding Module (Middle East) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Senegal	SN	SEN	Enterprise Geocoding Module (Africa) Universal Addressing Module
Serbia	RS	SRB	Enterprise Geocoding Module Universal Addressing Module
Seychelles	SC	SYC	Universal Addressing Module
Sierra Leone	SL	SLE	Universal Addressing Module
Singapore	SG	SGP	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Sint Maarten (Dutch Part)	SX	SXM	Universal Addressing Module
Slovakia	SK	SVK	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Slovenia	SI	SVN	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Solomon Islands	SB	SLB	Universal Addressing Module
Somalia	SO	SOM	Universal Addressing Module
South Africa	ZA	ZAF	Enterprise Geocoding Module Universal Addressing Module GeoComplete Module
South Georgia And The South Sandwich Islands	GS	SGS	Enterprise Geocoding Module Universal Addressing Module
South Sudan	SS	SSD	Universal Addressing Module
Spain	ES	ESP	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Sri Lanka	LK	LKA	Universal Addressing Module
Sudan	SD	SDN	Universal Addressing Module
Suriname	SR	SUR	Enterprise Geocoding Module (Latin America) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Svalbard And Jan Mayen	SJ	SJM	Universal Addressing Module
Swaziland	SZ	SWZ	Enterprise Geocoding Module (Africa) Universal Addressing Module
Sweden	SE	SWE	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Switzerland	CH	CHE	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
Syrian Arab Republic	SY	SYR	Universal Addressing Module
Taiwan, Province of China	TW or zh_TW (Routing)	TWN	Universal Addressing Module Enterprise Routing Module
Tajikistan	TJ	TJK	Universal Addressing Module
Tanzania, United Republic Of	TZ	TZA	Enterprise Geocoding Module (Africa) Universal Addressing Module Enterprise Routing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Thailand	TH	THA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module
Timor-Leste	TL	TLS	Universal Addressing Module
Togo	TG	TGO	Enterprise Geocoding Module (Africa) Universal Addressing Module
Tokelau	TK	TKL	Universal Addressing Module
Tonga	TO	TON	Universal Addressing Module
Trinidad and Tobago	TT	TTO	Enterprise Geocoding Module (Latin America) Universal Addressing Module
Tunisia	TN	TUN	Enterprise Geocoding Module (Africa) Universal Addressing Module
Turkey	TR	TUR	Enterprise Geocoding Module Universal Addressing Module GeoComplete Module
Turkmenistan	TM	TKM	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Turks And Caicos Islands	TC	TCA	Universal Addressing Module
Tuvalu	TV	TUV	Universal Addressing Module
Uganda	UG	UGA	Enterprise Geocoding Module (Africa) Universal Addressing Module
Ukraine	UA	UKR	Enterprise Geocoding Module Universal Addressing Module
United Arab Emirates	AE	ARE	Enterprise Geocoding Module (Middle East) Universal Addressing Module
United Kingdom	GB	GBR	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
United States	US	USA	Enterprise Geocoding Module Enterprise Routing Module Universal Addressing Module GeoComplete Module
United States Minor Outlying Islands	UM	UMI	Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Uruguay	UY	URY	Enterprise Geocoding Module Universal Addressing Module
Uzbekistan	UZ	UZB	Universal Addressing Module
Vanuatu	VU	VUT	Universal Addressing Module
Venezuela, Bolivarian Republic Of	VE	VEN	Enterprise Geocoding Module Universal Addressing Module
Viet Nam	VN	VNM	Universal Addressing Module
Virgin Islands, British	VG	VGB	Universal Addressing Module
Virgin Islands, U.S.	VI	VIR	Universal Addressing Module
Wallis and Futuna	WF	WLF	Universal Addressing Module
Western Sahara	EH	ESH	Universal Addressing Module
Yemen	YE	YEM	Enterprise Geocoding Module (Middle East) Universal Addressing Module

ISO Country Name	ISO 3116-1 Alpha-2	ISO 3116-1 Alpha-3	Supported Modules
Zambia	ZM	ZMB	Enterprise Geocoding Module (Africa) Universal Addressing Module
Zimbabwe	ZW	ZWE	Enterprise Geocoding Module (Africa) Universal Addressing Module

C - Validate Address Confidence Algorithm

In this section

Introduction to the Validate Address Confidence Algorithm 887
Confidence Algorithm for U.S. and Canadian Addresses 887
Confidence Algorithm for International Addresses 888

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
- State
- 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 148: 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%
State	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 149: 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	11.1111111111111	100%	0%	0%
City	11.1111111111111	50%	75%[5]	0%
State	16.6666666666667	100%	100	80%
PostalCode	16.6666666666667	100%	100%	80%
StreetName	16.6666666666667	50%	75%	50%
HouseNumber	16.6666666666667	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 150: 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	13.3333333333333	100%	0%	0%
City	13.3333333333333	50%	75%[8]	0%
State	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 151: 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	

Notices

© 2017 Pitney Bowes Software Inc. All rights reserved. MapInfo and Group 1 Software are trademarks of Pitney Bowes Software Inc. All other marks and trademarks are property of their respective holders.

USPS® Notices

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

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

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

Data Provider and Related Notices

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

© Copyright United States Postal Service. All rights reserved.

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

© 2016 HERE

Fuente: INEGI (Instituto Nacional de Estadística y Geografía)

Based upon electronic data © National Land Survey Sweden.

© Copyright United States Census Bureau

© Copyright Nova Marketing Group, Inc.

Portions of this program are © Copyright 1993-2007 by Nova Marketing Group Inc. All Rights Reserved

© Copyright Second Decimal, LLC

© Copyright Canada Post Corporation

This CD-ROM contains data from a compilation in which Canada Post Corporation is the copyright owner.

© 2007 Claritas, Inc.

The Geocode Address World data set contains data licensed from the GeoNames Project (**www.geonames.org**) provided under the Creative Commons Attribution License ("Attribution

License") located at **http://creativecommons.org/licenses/by/3.0/legalcode**. Your use of the GeoNames data (described in the Spectrum™ Technology Platform User Manual) is governed by the terms of the Attribution License, and any conflict between your agreement with Pitney Bowes Software, Inc. and the Attribution License will be resolved in favor of the Attribution License solely as it relates to your use of the GeoNames data.

3001 Summer Street
Stamford CT 06926-0700
USA

www.pitneybowes.com

© 2017 Pitney Bowes Software Inc.
All rights reserved