APPLICATION NOTE

Reverse Geo Web Service

Creating a Dealer Locator

INTRODUCTION:

Melissa's Reverse Geo Web Service allows you to find the nearest valid addresses to a latitude and longitude coordinate. One of the available features is the ability to return the distance from your current location to the nearest known location within a provided list. This is sometimes referred to as a dealer locator. This document explores the background and traditional usage of our Reverse Geo Web Service (RGWS), as well as capabilities and common applications.

BACKGROUND:

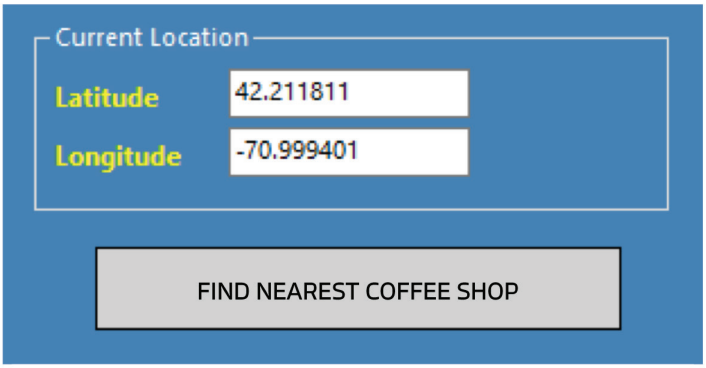
Traditionally, Reverse Geo Web Service (RGWS) returned the nearest valid address to the input (starting location) which was defined as a latitude longitude pair. This is commonly used for disaster relief mapping or emergency location applications. The 'doLookupFromList' feature allows the developer to instead obtain the nearest valid address from a predetermined finite list of valid addresses. Example applications might be to help customers find the closest healthcare clinic, or a particular coffee shop to their current location.

PREREQUISITES:

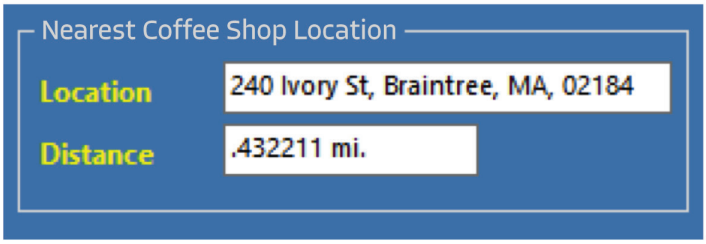
The input starting location can be defined by (in addition to the default set of lat/long coordinates) by a Melissa Address Key (MAK) – a unique identifier given to every valid U.S. address. The provided list of input locations to be looked up must be a list of MAKs and a unique identifier for each. Therefore, to use this feature, one must first convert a list of valid addresses to obtain their respective MAKs. Using this feature requires the request to use the JSON format.

Example: Where's the Nearest Dunkin' Donuts®?:

It may look like this:



And return this:



The above input could have started as a valid address – which would then require another Melissa service request to obtain its MAK, or respective latitude longitude pair, which would then be used for the end user input. But this would require another web service call and the example here demonstrates how the feature could easily be used in tandem with a geolocation API for smart device application development.

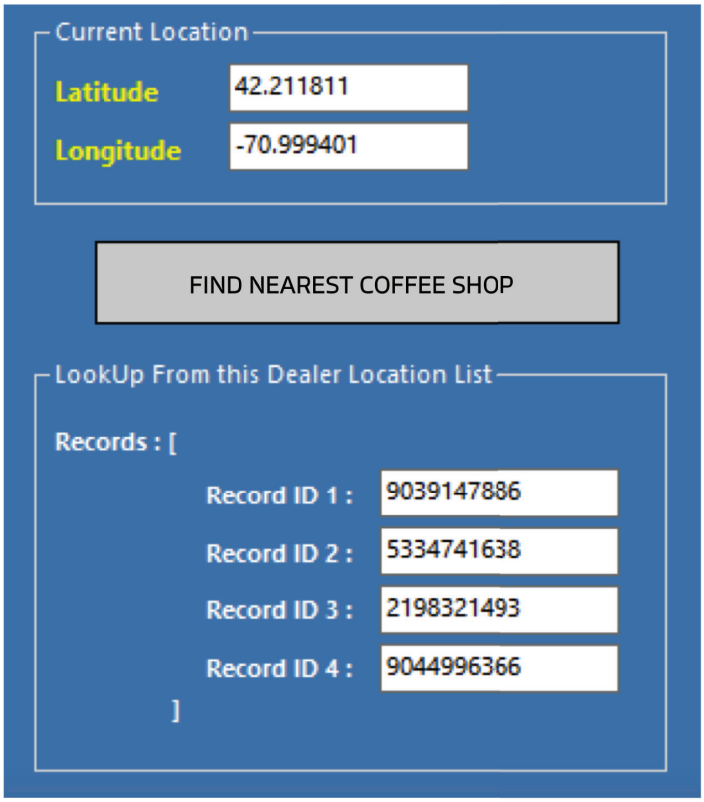
How does the developer create this?

First, you'll have to create a list of your locations that can be found by the user request. We recommend that the list be no larger than 100 entries, with the optimal number being much smaller for practical request / processing / response throughput efficiency. Then use another Melissa service, such as Personator, to obtain each location's MAK. For example:

STORE ID	ADDRESS	MAK
0001	551 Granite St, Braintree, MA 02184	9039147886
0002	375 Washington St, Braintree, MA 02184	5334741638
0003	240 Ivory St, Braintree, MA 02184	2198321493
0004	238 Grove St. Flr 1,02184	9044996366
0005	77 Washington St, Weymouth, MA 02188	5469049002
0006	1172 Washington St, Braintree, MA 02184	7810969005
0007	29 Hayward St, Braintree, MA 02184	7102530317
0008	366 Centre St, Quincy, MA 02169	6437599306

It's this MAK and its respective identifier which will be passed into each request.

So, behind the scenes, the previous customer interface is really doing this:



For simplicity, we've built the lookup list with just the first 4 locations. Taking a look at the actual JSON call:

https://reversegeo.melissadata.net/V3/WEB/ReverseGeoCode/doLookupFromList

```
"CustomerID": "LICENSE",
"Latitude": "42.211811",
"Longitude": "-70.999401",
"MelissaAddressKey": "",
"MaxDistance": "10",
"MaxRecords": "2",
"TransmissionReference": "tagToIdentifyThisRequest",
"Records": [
  {
    "RecordID": "0001",
    "MelissaAddressKey": "9039147886"
  },
  {
    "RecordID": "0002",
    "MelissaAddressKey": "5334741638"
  },
  {
    "RecordID": "0003",
    "MelissaAddressKey": "2198321493"
  },
  {
    "RecordID": "0004",
    "MelissaAddressKey": "9044996366"
```

Each request requires a valid license. Again, although our example inputs use geolocation as the lookup starting point, the request shows how a previously obtained MAK could have been used to override the latitude longitude. The developer may also set the distance to search from, and the number of found responses. Depending on the nature of the application and the building density of the business model, setting the MaxDistance too large may not be necessary in a suburban setting, whereas a search in a rural area might require a larger MaxDistance setting. Also, where many applications may just require returning a single 'closest' record, others may want a short list of sorted locations and therefore set the MaxRecords to 4 for example.

The above request would return a response like this:


```
"Version": "1.0.0.5164",
"TransmissionReference": "rgFROMLIST",
"TransmissionResults": "",
"Results": "GS07",
"TotalRecords": 2,
"Records": [
  {
    "AddressLine1": "240 Ivory St",
    "SuiteName": "",
    "SuiteCount": "4",
    "City": "Braintree",
    "State": "MA",
    "PostalCode": "02184",
    "AddressKey": "02184654099",
    "Latitude": "42.205253",
    "Longitude": "-71.000811",
    "Distance": "0.432211",
    "MelissaAddressKey": "2198321493",
    "MelissaAddressKeyBase": "0",
    "RecordID": "3"
  },
  {
    "AddressLine1": "375 Washington St",
    "SuiteName": "",
    "SuiteCount": "0",
    "City": "Braintree",
    "State": "MA",
    "PostalCode": "02184",
    "AddressKey": "02184470575",
    "Latitude": "42.219724",
    "Longitude": "-71.004037",
    "Distance": "0.607225",
    "MelissaAddressKey": "5334741638",
    "MelissaAddressKeyBase": "0",
    "RecordID": "2"
```


Notice that for this request, we were returned two records in the response. We had set the MaxDistance search to 10 miles, and this business does have more than two locations within that radius, but we also had configured the number of response records (the engine automatically returns them in sort or from closest location to furthest) to two, so that is the maximum number of records returned. Configuring the number of request Records, MaxDistance, and MaxRecords will have an impact on throughput efficiency due to the size of the request and the response, as well as the number of searches the algorithm has to perform to locate and return the correct number of response records.

Next Steps

To get started with Reverse Geo Web Service, you can:

Schedule a Demo

View the Quick Start Guide

Download Sample Guide

ABOUT MELISSA

Our 35+ years of address expertise started with ZIP+4 and turned into so much more. Melissa is a single-source vendor of global address management, data quality and identity verification solutions that help organizations harness accurate data for a more compelling customer view. Our industry-leading solutions have processed over 1 trillion address, name, phone and email records, making it clear why thousands of businesses worldwide trust Melissa with their data quality needs. For more information, visit www.melissa.com or call 1-800-Melissa.