HOME-TEST

A Python Geocoding service

Abstract

This document provides details on how to use a python based geocoding service as well as details on how to use the service API

Mohamed W. Mehrez Said

m.mehrez.said@mun.ca

Contents

Introduction:

This document provides details on how to run the geocoding service as well as details on how to use the service API. This program is a simple network service that returns latitude and longitude coordinates of a give address or a list of addresses. The service is user interactive and is based on two third party geocoding services, i.e. google maps and here maps.

How to Run the Service?

Here, we show the steps for downloading and running the service.

Download:

The service source code can be downloaded from the following link:

https://github.com/MMehrez/homeTestProject

System Requirements:

Running this service requires Python ver. 3.0 or later and an internet connection.

Running the Service under Windows (tested on Windows 7):

From the windows command prompt navigate to the source code location and run the following command

> python main.py

In case this command does not work, then it is likely that the windows environment variables need to be set. Check the following link to see how to set or change the PATH system variable in windows:

https://www.java.com/en/download/help/path.xml

When the previous command runs without errors, you should see the following introduction message on the command window

```
Welcome to HOME-TEST Geocoding Service!
This service returns the latitude and longitude coordinates for (a) given a
ddress(es).
Press Enter to continue...
```

This is simply a welcome message from the service that describes briefly the service function. In order to continue, you simply press the Enter key. Then the following message appears:

```
Enter the Address(es) separated by a (;)
```

Now the user is required to enter the address(es) of which their latitude and longitude coordinates are required. For more than one address, the user should separate the address by a semicolon (;).

Let's say the user entered the following two addresses:

```
Enter the Address(es) separated by a (;) Vancouver, BC, Canada ; St. jo
hn's, NL, Canada
```

Then, the service will process the two addresses and return the coordinates as the following:

```
The coordinates of Vancouver, BC, Canada  are: (latitude: 49.26039 ) an
d (longitude: -123.11336 )
The coordinates of  St. john's, NL, Canada are: (latitude: 47.5615096 )
and (longitude: -52.7125768 )
```

After returning the above message, the service will stop. In case you would like to check new addresses, you can simply rerun the service.

In some cases, the service may return the following message about a certain address

```
The coordinates of <entered address>  are: (latitude: Error ) and (long
itude: Error )
```

The reason for this returned error could be one or more of the following:

1- The entered address is not clear.
2- The computer running the service is not connected to the internet.
3- No address is entered at all.

Running the Service under Ubuntu:

From the terminal navigate to the source code location and run the following command

$ python3 main.py

Then, follow the same instructions for running the service as in the previous Subsection.

How to Use the Service API?

Here, we present the two functions used in the code, their use, and their response examples.

geocodingGoogle:

Tries to find an address geocode coordinates via the third-party service google maps.

Function call: geocodingGoogle(address)	
Parameter: address	**value:** 'string'

Response:
This function returns either an array containing the lat,lng coordinates upon a successful geocode request, or just an 'Error' string otherwise.
 	
Response example of a successful geocode request:
`{'lat': 49.2827291, 'lng': -123.1207375}`
 	
Response example of unsuccessful geocode request:
`'Error'`	

geocodingHere:

Tries to find an address geocode coordinates via the third-party service Here maps.

Function call: geocodingHere (address)	
Parameter: address	**value:** 'string'

Response:
This function returns either an array containing the lat,lng coordinates upon a successful geocode request, or just an array of error entries otherwise.
 	
Response example of a successful geocode request:
`{'lat': 49.2827291, 'lng': -123.1207375}`
 	
Response example of unsuccessful geocode request:
`{'lat': 'Error', 'lng': 'Error'}`	

Please note that the variable params defined in this function, i.e.

```
params = {'searchtext': requested_address,'app_id':'ZuuaESOoZ3r0TfUpSNLN',\
    'app_code':'cNeC1goF-_A2H5Zbth4vmQ'}
```

needs a careful attention, i.e. the entries `'app_id'` and `'app_code'` should be updated with the user/developer values. This is because these values belong to the original developer (Mohamed W. Mehrez) of this package and are valid only for 90 days.