# Loqate Sample Code in Python

## Overview

This tutorial provides sample code for developers to call Loqate Local API using Python.

## Pre-requisite

- Version of Python: Python 3 and above
- 32-bit/64-bit Windows or Linux

## Software Components

Sub sections below list out software components to be used on Windows and Linux Platforms.

### Windows (32-bit/64-bit)

Distributed components for 32-bit/64-bit Windows Platform include:

- *lqtcr.dll*
- *_pyloqate.pyd*
- *pyloqate.py*

32-bit components and 64-bit components are distributed as different sets, respectively. Those components cannot be mixed otherwise error messages such as below will be displayed:

```
ImportError: DLL load failed while importing _pyloqate: %1 is not a valid Win32 application.
```

The above error message will be produced when using 64-bit "*_pyloqate.pyd*" with 32-bit "*lqtcr.dll*" or using 32-bit "*_pyloqate.pyd*" with 64-bit "*lqtcr.dll*".

### Linux (64-bit)

Under Linux platform, only two components are needed:

- *_pyloqate.so*
- *pyloqate.py*

## Sample Code

Below illustrates a sample of Python code segment that calls Loqate Local API. Note that the sample Python code can be executed on both Windows and Linux platforms.

Once Python 3.0 (or higher version) are installed on a platform, run the following command to check the output:

For example:

1. Windows: (*lqtcr.dll*, *_pyloqate.pyd* and *pyloqate.py* must be in the same directory as *PyLoqateSample.py* which is "*Loqate_Projects*")

   C:\Loqate_Projects>python PyLoqateSample.py

2. Linux: (*_pyloqate.so* and *pyloqate.py* must be in the same directory as *PyLoqateSample.py*)

   $ python3 PyLoqateSample.py

```
"""
A sample code segment using Python to call Loqate Local API. File: PyLoqateSample.py

"""
import platform
import pyloqate

LOQATE_DATA_PATH = ""
if platform.system() == "Linux":
    LOQATE_DATA_PATH = "/mnt/c/Loqate/Releases/2022Q1.0_16013-6062831/data"
elif platform.system() == "Windows":
    LOQATE_DATA_PATH = "C:\\Loqate\\Releases\\2022Q1.0_16013-6062831\\data"

def main():
    # Create a server object
    srv = pyloqate.lqtServer_create()
    srv.init(LOQATE_DATA_PATH)

    # Create process list, options, input record and output result
    lst = pyloqate.lqtProcessList.create()
    opts = pyloqate.lqtProcessOptions.create()
    res = pyloqate.lqtProcessResult.create()
    rec = pyloqate.lqtInputRecord.create()

    # Set the relevant address field values
    rec.set("Address1", "805 Veterans Blvd Ste 320")
    rec.set("Locality", "Redwood City")
    rec.set("AdministrativeArea", "CA")
    rec.set("PostalCode", "94063")
    rec.set("Country", "USA")

    # Use Verify and Geocode processes
    lst.add("Verify", opts)
    lst.add("Geocode", opts)

    # Open a session
    session = srv.open()

    try:
        # Process the input address
        srv.process(rec, lst, res)

        # Display the result
        print("Version                : " + srv.getVersion())
        print("AVC                    : " + res.getAccuracyCode())

        for i in range(res.getCount()):
            print("Organization           : " + res.getField(i, "Organization"))
            print("SubBuilding            : " + res.getField(i, "SubBuilding"))
            print("Premise                : " + res.getField(i, "Premise"))
            print("Building               : " + res.getField(i, "Building"))
            print("PostBox                : " + res.getField(i, "PostBox"))
            print("DependentThoroughfare  : " + res.getField(i, "DependentThoroughfare"))
            print("Thoroughfare           : " + res.getField(i, "Thoroughfare"))
            print("DoubleDependentLocality: " + res.getField(i, "DoubleDependentLocality"))
            print("DependentLocality      : " + res.getField(i, "DependentLocality"))
            print("Locality               : " + res.getField(i, "Locality"))
            print("SubAdministrativeArea  : " + res.getField(i, "SubAdministrativeArea"))
            print("AdministrativeArea     : " + res.getField(i, "AdministrativeArea"))
            print("SuperAdministrativeArea: " + res.getField(i, "SuperAdministrativeArea"))
            print("PostalCode             : " + res.getField(i, "PostalCode"))
            print("Telephone              : " + res.getField(i, "Telephone"))
            print("CountryName            : " + res.getField(i, "CountryName"))
            print("ISO3166-2              : " + res.getField(i, "ISO3166-2"))
            print("Latitude               : " + res.getField(i, "Latitude"))
            print("Longitude              : " + res.getField(i, "Longitude"))
            print("GeoAccuracy            : " + res.getField(i, "GeoAccuracy"))
            print("Address                : " + res.getField(i, "Address"))

    except Exception as e:
        print("Address lookup failed with exception: {}".format(e), flush=True)

    # The memory must be released to avoid memory leak:

    # Destroy the process list, result and options
    pyloqate.lqtProcessList.destroy(lst)
    pyloqate.lqtProcessResult.destroy(res)
    pyloqate.lqtProcessOptions.destroy(opts)

    # Close the session and tidy up
    srv.close(session)
    srv.shutdown()
    pyloqate.lqtServer.destroy(srv)

if __name__ == "__main__":
    main()
```

## Sample Output

```
Version                 : 2.42.0.16028-2ed018b
AVC                     : V55-I55-P7-100
Organization            :
SubBuilding             : Ste 320
Premise                 : 805
Building                :
PostBox                 :
DependentThoroughfare   :
Thoroughfare            : Veterans Blvd
DoubleDependentLocality:
DependentLocality       :
Locality                : Redwood City
SubAdministrativeArea   : San Mateo
AdministrativeArea      : CA
SuperAdministrativeArea:
PostalCode              : 94063-1737
Telephone               :
CountryName             : United States
ISO3166-2               : US
Latitude                : 37.491400
Longitude               : -122.228800
GeoAccuracy             : P4
Address                 : 805 Veterans Blvd Ste 320<BR>Redwood City CA 94063-1737
```