13.6: GOOGLE GEOCODING WEB SERVICE

Contributed by Chuck Severance
Clinical Associate Professor (School of Information) at University of Michigan

Google has an excellent web service that allows us to make use of their large database of geographic information. We can submit a geographical search string like "Ann Arbor, MI" to their geocoding API and have Google return its best guess as to where on a map we might find our search string and tell us about the landmarks nearby.

The geocoding service is free but rate limited so you cannot make unlimited use of the API in a commercial application. But if you have some survey data where an end user has entered a location in a free-format input box, you can use this API to clean up your data quite nicely.

When you are using a free API like Google's geocoding API, you need to be respectful in your use of these resources. If too many people abuse the service, Google might drop or significantly curtail its free service.

You can read the online documentation for this service, but it is quite simple and you can even test it using a browser by typing the following URL into your browser:

http://maps.googleapis.com/maps/api/geocode/json?address=Ann+Arbor%2C+MI

Make sure to unwrap the URL and remove any spaces from the URL before pasting it into your browser.

The following is a simple application to prompt the user for a search string, call the Google geocoding API, and extract information from the returned JSON.

CODE 13.6.1 (PYTHON):

```python
import urllib.request, urllib.parse, urllib.error
import json

# Note that Google is increasingly requiring keys
# for this API
serviceurl = 'http://maps.googleapis.com/maps/api/geocode/json?'

while True:
    address = input('Enter location: ')
    if len(address) < 1: break

    url = serviceurl + urllib.parse.urlencode(
        {'address': address})

    print('Retrieving', url)
    uh = urllib.request.urlopen(url)
    data = uh.read().decode()
    print('Retrieved', len(data), 'characters')

    try:
        js = json.loads(data)
    except:
        js = None

    if not js or 'status' not in js or js['status'] != 'OK':
        print('==== Failure To Retrieve ====')
        print(data)
        continue
    print(json.dumps(js, indent=4))

    lat = js["results"][0]["geometry"]["location"]["lat"]
    lng = js["results"][0]["geometry"]["location"]["lng"]
    print('lat', lat, 'lng', lng)
    location = js['results'][0]['formatted_address']
    print(location)

# Code: http://www.py4e.com/code3/geojson.py
```

run restart

The program takes the search string and constructs a URL with the search string as a properly encoded parameter and then uses *urllib* to retrieve the text from the Google geocoding API. Unlike a fixed web page, the data we get depends on the parameters we send and the geographical data stored in Google's servers.

Once we retrieve the JSON data, we parse it with the *json* library and do a few checks to make sure that we received good data, then extract the information that we are looking for.

The output of the program is as follows (some of the returned JSON has been removed):

```
$ python3 geojson.py
Enter location: Ann Arbor, MI
Retrieving http://maps.googleapis.com/maps/api/
  geocode/json?address=Ann+Arbor%2C+MI
Retrieved 1669 characters
```

```
{
    "status": "OK",
    "results": [
        {
            "geometry": {
                "location_type": "APPROXIMATE",
                "location": {
                    "lat": 42.2808256,
                    "lng": -83.7430378
                }
            },
            "address_components": [
                {
                    "long_name": "Ann Arbor",
                    "types": [
                        "locality",
                        "political"
                    ],
                    "short_name": "Ann Arbor"
                }
            ],
            "formatted_address": "Ann Arbor, MI, USA",
            "types": [
                "locality",
                "political"
            ]
        }
    ]
}
lat 42.2808256 lng -83.7430378
Ann Arbor, MI, USA
```

```
Enter location:
```

You can download www.py4e.com/code3/geoxml.py to explore the XML variant of the Google geocoding API.