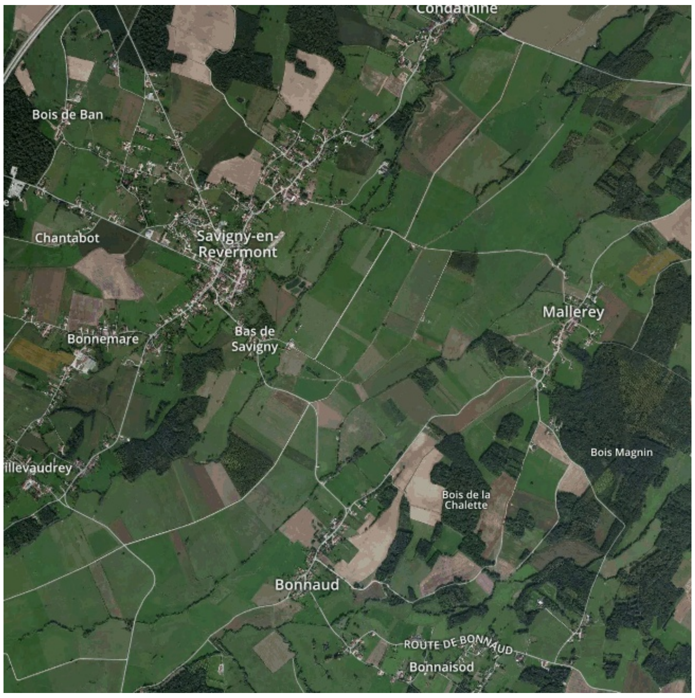
//

Geocoding

maXbox Starter93 – Code with Geocoding

In the beginning there was nothing, which exploded.
    — Terry Pratchett

Geocoding is the processing of single addresses into geographic coordinates and can be performed by a number of free online web sites. Many of these same sites can also perform reverse geocoding, or the use of geographic coordinates to determine addresses or place names. The web site links below provide access to free single address or place name geocoding services. I want to show 2 of them, a commercial and a free one.



Pic:93_5_mX4mapbox_bonnaud0(2).png

Just like every actual house has its address (which includes the number, the name of the street, city), every single point on the surface of earth can be specified by the latitude and longitude coordinates (lat & lng).
For example you see the location **Bonnaud** on the map above, we convert them to lat and long:
Coords: lat 46.621 lng 5.434 :**Bonnaud**, Val-Sonnette, Lons-le-Saunier, Jura, Bourgogne-Franche-Comté, France métropolitaine, 39190, France importance: 0.7350

Therefore, by using latitude and longitude we can specify virtually any point on earth, also like Delphi:

```
Geographic coordinates of Delphi, Greece
Latitude: 38°28'45" N
Longitude: 22°29'36" E
Elevation above sea level: 560 m = 1837 ft
```

OpenStreetMap

So how I did this? With the OpenStreetMap API Nominatim.
The OSM Nominatim is a search engine for OpenStreetMap data. From this site you may search for a name or address (like Route de Bonnaud, **Bonnaud**, France), or look up place names by geographic coordinates. Each result will link to details page where you can inspect what data about the object is saved in the database and investigate how the address of the object has been computed (URI and JSON for example):

```
function TAddressGeoCodeOSM(faddress: string): string;
var
  url, res, display: string;
  jo, location: TJSONObject;
  urlid: TIduri; windown: TWinApiDownload;
 begin
  urlid:= TIdURI.create('');
  url:= urlid.URLEncode('https://nominatim.openstreetmap.org/search?
                                          format=json&q='+fAddress);
  windown:= TWinApiDownload.create;
  windown.useragent:= 'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1';
  windown.url:= url;
  windown.download1(res);
  //windown.OnWorkStart
  StrReplace(res, '[{', '{');
  jo:= TJSONObject.create4(res);
  try
    if jo.getString('place_id') <> ' ' then
      display:= jo.getstring('display_name');
    result:= Format('Coords: lat %2.3f  lng %2.3f  :%s  importance: %2.4f',
                   [jo.getdouble('lat'),jo.getdouble('lon'),display,
                                        jo.getdouble('importance')]);
  except
    writeln('E: '+ExceptiontoString(exceptiontype, exceptionparam));
  finally
    jo.Free;
    urlid.free;
    windown.free;
  end;
end;
```

I use an URI, a JSON and a HTTPGet object and call the function:

```
writeln('res back_: '+TAddressGeoCodeOSM('Bonnaud, France'));
```

You should in particular verify that you have set a custom HTTP referrer or HTTP user agent (*windown.useragent:=*) that identifies your application, and that you are not overusing the service with massive bulk requests. Otherwise you get following message:

<p>You have been blocked because you have violated the<a href="https://operations.osmfoundation.org/policies/nominatim/">usage policy</a>
of OSM's Nominatim geocoding service. Please be aware that OSM's resources are limited **and** shared between many users. The usage policy **is** there **to** ensure that the service remains usable **for** everybody.</p>

In case you missed the user agent, I passed it already with the constructor:

```
constructor TWinApiDownload.Create;
begin
  inherited;
  fUserAgent:= 'Mozilla/5.001(windows; U; NT4.0; en-US;) Gecko/25250101';
  fProgressUpdateInterval:= 100;
  fCachingEnabled:= True;
  fStop:= False;
  fActive:= False;
end;
```

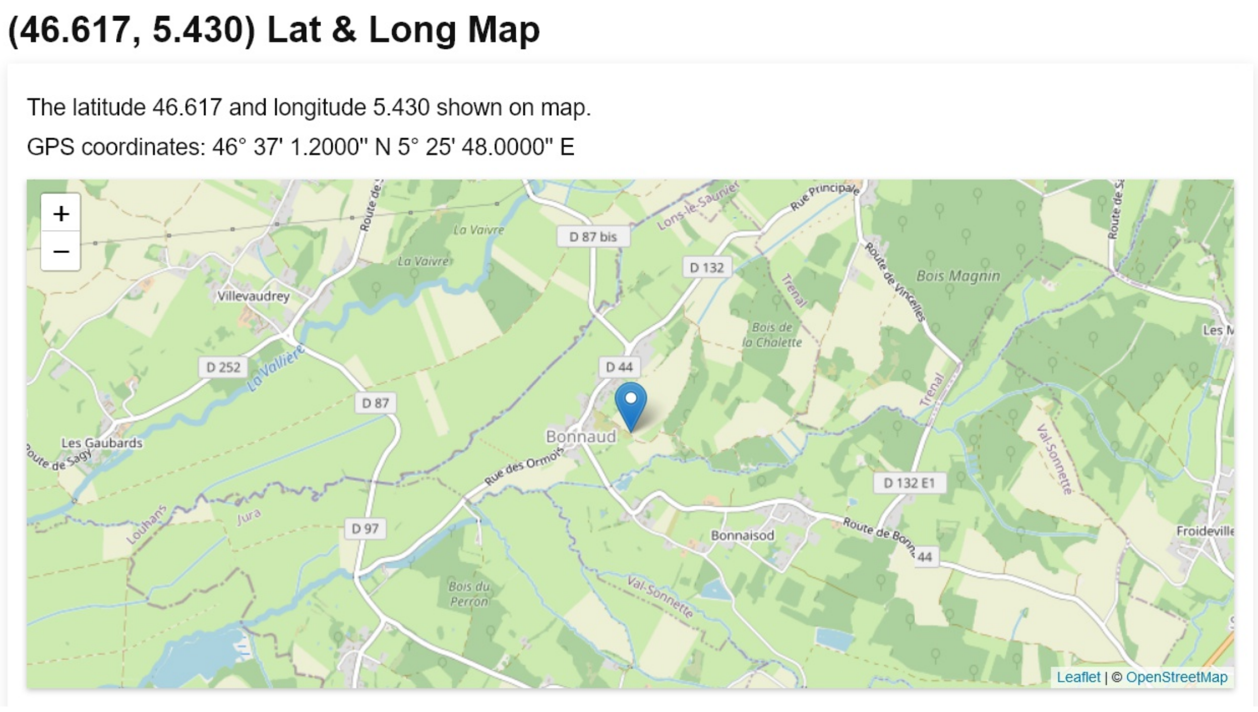
So with OpenStreetMap the call is easy just to get an https:// and user agent to provide. OpenStreetMap (OSM) is a collaborative project to create a free editable map of the world. It is built by a community of mappers that contribute and maintain OSM data. This REST endpoint is also used to reverse geocode, this is, generate an address from a latitude and a longitude. This is the format of the request:

```
   with TWinApiDownload.create do begin
     Useragent:= 'Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1';
     Url:= 'https://nominatim.openstreetmap.org/reverse?lat='
              +flots(flat)+'&lon='+flots(flong)+'&zoom=10&format=json';
     Download1(rest);
     //writeln(rest)
     with TJSONObject.create4(rest) do begin
       writeln('display_name: '+getstring('display_name'));
       free;
     end;
     free;  //ApiDown
   end;
```

The double **with** statement is functional programming, the rest variable is a stringstream in *Download1(rest);* and we free the 2 objects immediately.
A geocoding service (also called an address locator) is a program that allows for a user to input a batch of data contained in a table, search for matches as compared to a reference table, and output the result in a map or GIS layer format. Now we can turn our geocoding coordinates in a map:

https://www.latlong.net/c/?lat=46.617&long=5.430

Two attributes in OSM are of interest, the zoom and the importance. Renderers can't display on a map in each zoom level, so they have to make a selection. The more important an object is, the earlier (in zooming in) the object should be rendered. Also the this key "importance" with a value which refers to the area this object is important for, e.g. regional or urban:
res back_: Coords: lat 46.62084 lng 5.43424 :Bonnaud, Val-Sonnette, Lons-le-Saunier, Jura, Bourgogne-Franche-Comté, France métropolitaine, 39190, France importance: 0.7350


(46.617, 5.430) Lat & Long Map

The latitude 46.617 and longitude 5.430 shown on map.

GPS coordinates: 46° 37' 1.2000" N 5° 25' 48.0000" E

Pic: 93_mX4mapbox_osm_bonnaud_referencemap.jpg

Pic: 93_mX4mapbox_weatherapp_bonnaud_sat_reference.jpg

By the way you can simply open the map in maXbox with the command:

```
OpenWeb('https://www.latlong.net/c/?lat=46.617&long=5.430');
```

Currently there are many options to choose from when geocoding batches of address data. Lets turn our interest to commercial Services. We all know Google or the Google Maps Platform: This Geocoding API can be used to geocode worldwide addresses after obtaining an API key. You will need to enter a credit card to set up a billing account with Google, and they will give you $200/month credit for the first 12 months (view their pricing structure). They will not charge you until you give them permission to. You may also apply for additional educational credit up to $250/month. In case you don't have a valid billing:

```
Exception:
TJSONObject["status--\u0020You\u0020must\u0020enable\u0020Billing\u0020on\u0020t
he\u0020Google\u0020Cloud\u0020Project\u0020at\u0020https\u003A//console.cloud.g
oogle.com/project/_/billing/enable\u0020Learn\u0020more\u0020at\u0020https\u003A
//developers.google.com/maps/gmp-get-started"] not found.
```

Geocodio

Geocoding (also known as forward geocoding) allows you to convert one or more addresses into geographic coordinates (i.e. latitude and longitude). Geocoding will also parse the address and append additional information (e.g. if you specify a zip code, **Geocodio** will return the city and state corresponding zip code as well).

Geocodio supports geocoding of addresses, cities and zip codes in various formats. Geocodio provides bulk geocoding and reverse lookup services through a REST API. The API is able to process a single address, as well as handle bulk requests of up to 10,000 addresses. The results are returned with an accuracy score indicating the confidence Geocodio has in the accuracy of the result. It is also able to parse addresses into individual components.

So I started with Geocodio https://www.geocod.io/docs/#geocoding and registered to get an API Key. Geocodio’s RESTful API allows you to perform forward and reverse geocoding lookups. They support both batch requests as well as individual lookups, but and this is the disadvantage, for the moment only for USA and Canada.

The base API url is https://api.geocod.io/v1.7/.

All HTTP responses (including errors) are returned with JSON-formatted output. For the code I tested with Python and maXbox with an Wininet and JSON Object:

```
function TAddressGeoCoding4(faddr, fcountry: string): String;
var
  Url,API_KEY, source: string;
  jo, locate: TJSONObject;
  urlid: TIdURI;
  fLat,fLong: double;
```

```
begin
  urlid:= TIdURI.create('');
  API_KEY:='785b4141b...................'; //get your own one please
  if fcountry <> '' then
    Url:= urlid.URLEncode('https://api.geocod.io/v1.7/geocode?q='+
                          fAddr+'&country='+fcountry+'&api_key='+API_KEY) else
    Url:= urlid.URLEncode('https://api.geocod.io/v1.7/geocode?q='+
                          fAddr+'&api_key='+API_KEY);

  jo:= TJSONObject.Create4(wdc_WinInet_HttpGet2(Url,Nil));
  try
    jo.getString('input')
    locate:=
jo.getJSONArray('results').getJSONObject(0).getJSONObject('location');
    source:= jo.getJSONArray('results').getJSONObject(0).getString('source');
    //geometry.getJSONObject('coordinates');
    fLat:= locate.getDouble('lat')
    fLong:= locate.getDouble('lng');
    result:=Format('Coordinates: lat %2.3f lng %2.3f :%s',[flat,flong,source]);
  except
    Xraise(Exception.Create(jo.getString('error')));
  finally
    jo.Free;
    urlid.free;
  end;
end;
```

You can test your REST-Call direct in the browser with:

https://api.geocod.io/v1.7/geocode?q=cologne&api_key=785b4141b&#8230...;

Or you choose a map one tester online:
https://www.geoapify.com/geocoding-api

One note to the code function above. I pass and address optional the country (to prevent name conflicts) to get the coordinates from the location:

```
 writeln(TAddressGeoCoding4('Ontario','Canada'));
```

In most cases, the standard output format would be used. In certain situations, it can however be beneficial to work with a JSON structure that is specifically designed for your use case.

The interesting part is the line

```
  jo:= TJSONObject.Create4(wdc_WinInet_HttpGet2(Url,Nil));
```

with the constructor to pass the httpget2 which returns the json result in one line to the *TJSONObject*.

If a JSON object is posted, you can specify a custom key for each element of your choice. This can be useful to match queries up with your existing data after the request is complete. If using a JSON array, results are **guaranteed** to be returned in the same order as they are requested.

The Geocodio API implements the concept of “warnings”. This is meant to assist and guide developers when implementing our API. But I checked also the error as Json field in the try except catch. To prevent misleading, inconsistent or duplicated locations like Cologne, Paris or Bern all over the world we should also check the **source** as identity authentication too.

If a city is provided without a state, Geocodio will automatically guess and add the state based on what it is most likely to be. Geocodio also understands shorthand's for both streets and cities, for example *NYC*, *SF*, etc., are acceptable city names. Each geocoded result is returned with an **accuracy** score (like the importance in OpenStreetMap), which is a decimal number ranging from 0.00 to 1.00. This score is generated by an internal Geocodio engine based on how accurate the result is believed to be. The higher the score, the better the result. Results are always returned ordered by accuracy score.

Conclusion

Geocoded locations expressed in latitude, longitude coordinates can be obtained one at a time in web maps such as OSM, Geocodio, Bing Maps or Google Maps. **Reverse geocoding** is the process of converting latitude and longitude into a street address. The relative ease of geocoding and resulting accuracy can vary widely depending on a number of factors. What is the nature or the source of the data? How accurate is it and what format is it in? What geocoding technique will be used like REST or batch or scripts? Determining a geocoding strategy that best suits a particular need is not always clear.

The key to confidently geocoding data lies in understanding the reference table of the GIS which the data is being matched to, how a match is found, and the resulting spatial accuracy.

Scriptname: 1108_CAI_GeocodingTWininetAPI_Maponly.pas

Geocoding topics and Script

-
- https://www.geoapify.com/geocoding-api
- https://www.latlong.net/
- https://www.openstreetmap.org/
- https://sourceforge.net/projects/maxbox/files/Examples/EKON/EKON26/1108_CAI_GeocodingTWininetAPI_Maponly.pas/download
- https://www.countrycoordinate.com/search/?keywords=bern%2C+switzerland

Author: Max Kleiner, April 2022