EECS Seminars

Developing Android Location-aware Apps

Victor Matos
Cleveland State University

Agenda

Developing Android Mapping Apps

What is this talk about?
We will discuss current technologies used to create attractive, powerful and effective location-aware solutions for Android powered devices.

Sections
1. Overview of the Google Maps API resources
2. Review various Android strategies to accommodate mapping operations.
Section 1 - Google Maps API

Introduction

Section 1 - Google Maps API is a web-service developed by Google to assist desktop and mobile devices on their geo-mapping needs.

Includes

- satellite imagery,
- street maps,
- 360° panoramic views of streets (3D Street Views),
- real-time traffic conditions,
- route planning for traveling by foot, car, bicycle, or public transportation,
- Direct and Reverse Geo-coding,
- Elevation and time-zone information,
- Area demarcation using polygons and polylines,
- KML location marking, etc.

Target Platforms

Section 1 - Google Maps APIs are available for a variety of platforms including:

- Android
- iOS
- Web
- Web services

Mapping is free for Google Maps Android API and Google Maps SDK for iOS (although there is a limit of total executions per/day).
Section 1 - Google Maps API

How to Become a Google Map's Developer?

1. Obtain a Developer's Key
Visit the Google Developers Console to obtain a KEY as well as permission to use the resources https://console.developers.google.com/home/dashboard

40 chars key looks like: AIzaSyD0dNC21Acp0TXNKCMIp4-CU

2. Identify & Activate Services

Sixteen services
### Section 1 - Google Maps API

#### 1. Google Maps Android API & Google Maps iOS API

This API interacts with location-aware Android apps to expose maps created with data extracted from the Google Maps Service.

The API automatically handles access to Google servers, map rendering and response to user gestures such as clicks and drags.

Examples will be shown later

#### 2. Google Maps JavaScript API

- The JavaScript API allows you to create customizable *(dynamic)* maps on your **website**.
- A map is shown inside of the designated HTML container (typically a **DIV** element).
- The API includes services such as geocoding, directions, Street View.
- The rendering of the map and its features, as well as its interaction with the user is supported through JavaScript methods.

References: [https://developers.google.com/maps/documentation/javascript/examples/directions-panel](https://developers.google.com/maps/documentation/javascript/examples/directions-panel)
Section 1 - Google Maps API

2. Google Maps JavaScript API

```html
<!DOCTYPE html>
<html>
<head>
    <title>Simple Map</title>
    <style>
        html, body {
            height: 100%; margin: 0; padding: 0;
            #map {
                height: 100%;
            }
        }
    </style>
</head>
<body>
    <h1>Welcome to Cleveland State University</h1>
    <div id="map"></div>
    <script>
        var map;
        function initMap()
        {
            map = new google.maps.Map(document.getElementById('map'),
            {
                center: {lat: 41.502610, lng: -81.675464},
                zoom: 15
            });
        }
    </script>
    <script src="https://maps.googleapis.com/maps/api/js?callback=initMap &key=YOUR_API_KEY" async defer></script>
</body>
```

Section 1 - Google Maps API

2. Google Maps JavaScript API

![Welcome to Cleveland State University](image)
Lesson 24

Section 1 - Google Maps API

2. Google Maps JavaScript API

You could place a lightweight static map on your website without any JavaScript coding. The map request is based on URL parameters sent through a standard HTTP request and returns the map as an image.

http://maps.googleapis.com/maps/api/staticmap?size=640x400&center=Cleveland+State+University,+Cleveland+Ohio&key=YOUR-API-KEY

3. Google Static Maps API
Lesson 24

Section 1 - Google Maps API

4. Google Maps Roads API

Identifies the best-fit road geometry for a given set of location points. Points are generally taken along a path on which you have been traveling.

EXAMPLE: In this example the user supplies four locations along her path from school to a baseball game at Progressive Field. In return the SnapPoints service provides a set of coordinates closely matching known roads better describing her trajectory.

https://roads.googleapis.com/v1/snaptoroads?path=41.503224,-81.672490|41.499273,-81.675420|41.497044,-81.679618|41.495133,-81.684059&interpolate=true&key=YOUR-API-KEY

<table>
<thead>
<tr>
<th>Location</th>
<th>Coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSU - College of Engineering</td>
<td>41.503224,-81.672490</td>
</tr>
<tr>
<td>CSU – Convocation Center</td>
<td>41.499273,-81.675420</td>
</tr>
<tr>
<td>CSU Overflow Parking</td>
<td>41.497044,-81.679618</td>
</tr>
<tr>
<td>Progressive Field</td>
<td>41.495133,-81.684059</td>
</tr>
</tbody>
</table>

The answer is presented in JSON notation (only a few lines are shown)

```json
{
  "snappedPoints": [
    {
      "location": {
        "latitude": 41.5026151,
        "longitude": -81.6724211
      },
      "originalIndex": 0,
      "placeId": "ChIJnYO8GGH6MIgRy6k0LbALluM"
    },
    {
      "location": {
        "latitude": 41.5021887,
        "longitude": -81.6723270999999
      },
      "placeId": "ChIJnYO8GGH6MIgRy6k0LbALluM"
    },
    {
      "location": {
        "latitude": 41.502054099999988,
        "longitude": -81.6722935
      },
      "placeId": "ChIJnYO8GGH6MIgRy6k0LbALluM"
    },
    // ...
  ]
}
```
4. Google Maps Roads API

After parsing the JSON response and plotting the returned interpolated fit to the original sequence of points.

SNAPPED LOCATIONS
41.501585, -81.675271
41.502054, -81.672293
41.502190, -81.672251
41.501896, -81.672251
41.502054, -81.672166
41.502054, -81.672032
41.502615, -81.671860
41.502190, -81.671859
41.502615, -81.673164
41.502190, -81.673164
41.502615, -81.675441
41.502190, -81.675441
41.502615, -81.675512
41.502190, -81.675512
41.502615, -81.675529
41.502190, -81.675529
41.502615, -81.675543
41.502190, -81.675543
41.502615, -81.675551
41.502190, -81.675551
41.502615, -81.675539
41.502190, -81.675539
41.502615, -81.675518
41.502190, -81.675518
41.502615, -81.675473
41.502190, -81.675473
41.502615, -81.675415
41.502190, -81.675415
41.502615, -81.675340
41.502190, -81.675340
41.502615, -81.675736
41.502190, -81.675736
41.502615, -81.676757
41.502190, -81.676757
41.502615, -81.677315
41.502190, -81.677315

This API allows you to include a static Street View panorama (or thumbnail) into your webpage. No JavaScript programming is needed. The viewpoint (direction of the scene and angle of the camera) is specified in the sender’s URL string which is transmitted using plain HTTP.

https://maps.googleapis.com/maps/api/streetview?size=600x400&location=41.501585,-81.675271&heading=-40&pitch=5&fov=90&key=YOUR-API-KEY

CSU Location: 41.501585° -81.675271°
This API allows you to fill an HTML iFrame element with a dynamic Google map which could include:

- base maps,
- indoor plans,
- satellite images,
- street views, and
- turn-by-turn directions.

The map is defined in the src attribute of the <iframe>.

The webpage holding the iFrame element uses plain HTML transport.
Lesson 24

Section 1 - Google Maps API

6. Google Maps Embed API

Example: Draw an embedded map showing directions from Cleveland to Indianapolis

```html
<body>
  <h1>Section 1 - Google Maps API Embed - Directions</h1>
  <iframe
    width="600"
    height="450"
    frameborder="0"
    style="border:0"
    src="https://www.google.com/maps/embed/v1/directions
      ?origin=Cleveland+Ohio
      &destination=Indianapolis+Indiana
      &key=YOUR-API-KEY"
    allowfullscreen>
  </iframe>
</body>
</html>
```
Lesson 24

Section 1 - Google Maps API

6. Google Maps Embed API

Example: Draw an embedded map showing a satellite view of Cleveland St.

```html
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
  <H1>Section 1 - Google Maps API Embed - View</H1>

  <iframe width="600" height="450"
    frameborder="0" style="border:0"
    src="https://www.google.com/maps/embed/v1/view
?center=41.502610,-81.675464
&zoom=18&maptype=satellite
&key=YOUR-API-KEY"
  allowfullscreen>
  </iframe>

</body>
</html>
```
**Section 1 - Google Maps API**

6. Google Maps Embed API

Example: Draw an embedded map showing a street view image of CSU's Student Center

```html
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>

<H1>Section 1 - Google Maps API Embedd - Street View</H1>

<iframe width="600" height="450" frameborder="0" style="border:0"
src="https://www.google.com/maps/embed/v1/streetview
?location=41.501645,-81.675179
&heading=-45&pitch=10&fov=40
&key=YOUR-API-KEY"
allowfullscreen>
</iframe>

</body>
</html>
```
Section 1 - Google Maps API

6. Google Maps Embed API

Example: Draw an embedded map showing a search for restaurants in Cleveland's downtown.

```html
<html xmlns="http://www.w3.org/1999/xhtml">
<head></head>
<body>

<H1>Section 1 - Google Maps API Embedd - Search</H1>

<iframe width="600" height="450" frameborder="0" style="border:0"
    src="https://www.google.com/maps/embed/v1/search?q=restaurants+in+downtown+Cleveland&key=YOUR-API-KEY"
    allowfullscreen>
</iframe>

</body>
</html>
```
6. Google Maps Embed API

This API is useful when the user is interested in locating points-of-interest (POI) on or around a particular place. The POI database currently includes about 100 million entries (it is frequently updated – owner verified strategy). POIs cover a wide range of categories. Other key features of the API include autocomplete, nearby-search, place picker, photos and add place.

https://maps.googleapis.com/maps/api/place/autocomplete/json?components=country:us&input=Rock+and+Roll&key=YOUR-API-KEY

Rock and Roll Hall of Fame
Rock and Roll Hall of Fame and Museum - Library and Archives
Rock and Roll Boulevard
Rock & Roll City Studios
Lesson 24

Section 1 - Google Maps API

7. Google Places API Web Service - NearbySearch

Example: Search for FOOD places in the vicinity (500m) of CSU’s location.

https://maps.googleapis.com/maps/api/place/nearbysearch/json
?location=41.502610, -81.675464 &radius=500
&types=food
&key=YOUR-_API-KEY

Becky’s
Cafe Ah-Roma
Artefino Art Gallery Café
Moko Café
Otto Moser’s Restaurant
Elements Bistro
Subway
China Sea Express

Section 1 - Google Maps API

8. Google Places API Web Service - TextSearch

Example: Use text query to search for restaurants in Cleveland’s downtown.

https://maps.googleapis.com/maps/api/place/textsearch/json
?q=restaurants+in+Cleveland+downtown
&key=YOUR-_API-KEY

Lola
The Greenhouse Tavern
Blue Point Grille
Slyman’s Restaurant
Superior Pho Restaurant
Mallorca
Johnny’s Downtown
Chinato
Lesson 24

### Section 1 - Google Maps API

#### 9. Google Maps Geocoding API

Straight and reverse Geocoding allow you to respectively

1. supply a postal address and obtain its corresponding geographic coordinates, and
2. enter a coordinate and obtain in return the postal address (or its Google Database Object ID) for the supplied coordinates.

<table>
<thead>
<tr>
<th>Postal Address</th>
<th>Geo-Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>2121 Euclid Avenue, Cleveland, Ohio 44114</td>
<td>Lat: 41.502610, Lon: -81.675464</td>
</tr>
</tbody>
</table>

Example: (Straight Geocoding) Convert postal address into location

[Example Code]

```json
https://maps.googleapis.com/maps/api/geocode/json
  ?address=2121+Euclid+Ave+Cleveland+Ohio+44115
  &key= YOUR-API-KEY
```

Edited JSON response

```json
{
  "results": [
    {
      "address_components": [
        
      ],
      "formatted_address": "2121 Euclid Ave, Cleveland, OH 44115, USA",
      "geometry": {
        "location": {
          "lat": 41.5024529,
          "lng": -81.6752705
        }
      }
    }
  ]
}
```
Lesson 24

Section 1 - Google Maps API

9. Google Maps Geocoding API

Example: (Reverse Geocoding) Convert location to postal addresses

https://maps.googleapis.com/maps/api/geocode/json
?latlng=41.5024529,-81.6752705
&key=YOUR-API-KEY

Parsed/Edited JSON response (places on the vicinity of the location)

Rhodes West, Cleveland, OH 44115, USA
2121 Euclid Ave, Cleveland, OH 44115, USA
E 21ST St & Euclid Av, Cleveland, OH 44115, USA
Downtown, Cleveland, OH, USA
Cleveland, OH, USA
Cleveland, OH 44115, USA
Cuyahoga County, OH, USA
Ohio, USA
United States

10. Google Maps Directions API

1. The Directions API service provides the capability to select and display on a map a recommended path between two locations.

2. The selected path could be customized to a particular mode of transportation including: driving, cycling, walking and public transportation.

3. Your request is written as a parameterized HTTP URL.

4. The system responds with a dynamic map that offers detailed visual guidance, as well as textual turn-by-turn details.

5. You may specify origins, destinations and waypoints either as text strings or as latitude/longitude coordinates.

6. No developer KEY is needed in the request.
**Section 1 - Google Maps API**

**10. Google Maps Directions API**

**Example:** Obtain directions to go from CSU to the Rock & Roll Hall of Fame

https://www.google.com/maps/dir/Cleveland+State+University,+Cleveland+Ohio/Rock+and+Roll+Hall+of+Fame+Cleveland

**11. Google Maps Distance Matrix API**

1. The Google Maps Distance Matrix API is a service applicable to mapping problems involving a set of starting and ending points.

2. The service provides **travel distance and time for each pair of points** taken from **origins** and **destinations**.

3. The information returned is based on the recommended route between start and end points, as calculated by the Section 1 - Google Maps API, and consists of rows containing duration and distance values for each pair.

4. This service does not return detailed route information.

5. Route information can be obtained by passing the desired single origin and destination to the Google Maps Directions API.
11. Google Maps Distance Matrix API

https://maps.googleapis.com/maps/api/distancematrix/json
?origins=Columbus+OH|41.5024529,-81.6752705
&destinations=1600+Pennsylvania+Ave+Washington+DC|New+York+NY
&key=YOUR- API -KEY

<table>
<thead>
<tr>
<th>DESTINATIONS</th>
<th>ORIGINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1600 Pennsylvania Ave Washington DC</td>
<td>New York, NY</td>
</tr>
<tr>
<td>Columbus, OH, USA</td>
<td>655 Km 6 hours 25 mins</td>
</tr>
<tr>
<td>1899 E 22nd St, Cleveland, OH 44114, USA</td>
<td>610 Km 5 hours 54 mins</td>
</tr>
</tbody>
</table>

12. Google Maps Geolocation API

• The Google Maps Geolocation API is useful in situations in which the use of GPS is not possible.

• It returns a location and accuracy radius based on information about cell towers and WiFi nodes that the mobile client can detect.

• Communication is done over HTTPS using POST.

• Both request and response are formatted as JSON.

• Use CURL to invoke the service and supply JSON encoded location data.
Lesson 24

Section 1 - Google Maps API

12. Google Maps Geolocation API

Example: Ask for your coordinates. Supply your router’s WIFI address. In Windows use: “ipconfig /all” look for your MAC physical address. See below JSON encoded location based on WIFI id. Use CURL to invoke service.

```
curl -d "@c:/temp/myfileBU342Office.json" 
    -H "Content-Type: application/json" 
    -i "https://www.googleapis.com/geolocation/v1/geolocate 
      ?key=YOUR-API-KEY"
```

myFileBU342Office.json

```
{
  "homeMobileCountryCode": 310,
  "wifiAccessPoints": [
    {
      "macAddress": "80:56:F2:CE:E6:25"
    }
  ]
}
```

Response

```
{
  "location": {
    "lat": 41.5039782,
    "lng": -81.6747187
  },
  "accuracy": 441.0
}
```

CURL is available from: https://curl.haxx.se/

13. Google Maps Elevation API

You may request the altitude of any place on earth including underwater locations on the ocean floor (expressed as negative values). The API also calculates elevation changes along a given path.

Example: Find the elevation of Cleveland State U. respect to the sea level (203.63 m)

```
https://maps.googleapis.com/maps/api/elevation/json
?locations=41.5024529,-81.6752705
&key=YOUR-API-KEY
```

Response

```
{
  "results": [
    {
      "elevation": 203.6396942138672,
      "location": {
        "lat": 41.5024529,
        "lng": -81.6752705
      },
      "resolution": 4.771975994110107
    }
  ],
  "status": "OK"
}
```
Lesson 24

Section 1 - Google Maps API

14. Google Maps Time Zone API

- The Google Maps Time Zone API provides the time zone and UTC offset for any location on the planet. You also receive the name of the time-zone and its daylight-saving offset (dstOffset).
- The timestamp input parameter represents the local time using Unix-TimeStamp format (seconds since Jan-1-1970 @ midnight – Epoch Time).
- The local time of a given location is the sum of the timestamp parameter, and the dstOffset and rawOffset fields from the result.

Example: Assume we are at Cleveland State University (41.5024529, -81.6752705) on February 5, 2016 and local time is 7:30PM. The Unix timestamp is 1454718600.

https://maps.googleapis.com/maps/api/timezone/json?location=41.5024529,-81.6752705&timestamp=1454718600&key=YOUR-API-KEY

Response

```json
{
    "dstOffset": 0,
    "rawOffset": -18000,
    "status": "OK",
    "timeZoneId": "America/New_York",
    "timeZoneName": "Eastern Standard Time"
}
```

Section 2 - Android OS

Architecture

Android is an open-source operating system used for smartphones and tablet computers.

It allows access to Google’s own services like Search, YouTube, Maps, Gmail and more.
**Activity’s Life Cycle**

1. **Option 1 (Simple but limited)**
   - Trigger an *Intent* requesting ACTION.VIEW services. Add to the Intent a URI or URL – similar to those discussed in Section1 - invoking the Google Map API capabilities. Map activity is *disconnected* from the rest of the application.

2. **Option 2 (Portability across devices – Complex - Look & Feel discrepancies)**
   - Applications are made of HTML+CSS+JavaScript pages displayed on WebViews. Device’s local data and hardware resources can be shared with the WebViews using additional interfaces. Platform-independent development may not rely on SDK.

3. **Option 3 (Device Optimized Experience)**
   - Use the **Google Maps Android API** & Android **MapFragment** to receive maps (and data)
     - access to Google Maps servers,
     - data downloading,
     - map display,
     - response to map gestures,
     - add markers, polygons, and overlays to a basic map.
     - user interaction with the map.
Option 1. Android Intents

- An **intent** allows you to start an activity in another app by describing a simple action you’d like to perform (such as “view a map” or “take a picture”) in an **Intent** object.

- **Implicit intents** do not specify the app component to start, but instead specifies an **action** and provides some **data** to perform the action.

- Common Mapping Intents invoke the ACTION.VIEW with a URI similar to the following examples:

**Data URI Scheme**

```java
String urlStr = "geo:0,0?q=2121+Euclid+Ave+Cleveland+Ohio+44115";
String urlStr = "geo:41.501645,-81.675179?z=11";
String urlStr = "geo:0,0?q=41.501645,-81.675179(CSU+Student's+Center)";
String urlStr = "https://www.google.com/maps/place/Cleveland+State+University";
String urlStr = "https://www.google.com/maps/dir/Cleveland+State+University/The+Rock+and+Roll+Hall+of+Fame+and+Museum"
```

Android Mapping Strategies

### Option 1. Android Intents

Show a map centered around given coordinates

```java
String urlStr = "geo:0,0?q=41.501645,-81.675179(CSU Student's Center)Z=15";
Intent myMapIntent = new Intent(
    android.content.Intent.ACTION_VIEW,
    Uri.parse(urlStr));
startActivity(intent);
```
Option2. WebViews

- You may design an Android App in the same way you create a website.
- The app consists of a set **HTML+JavaScript** pages which are displayed on Android containers called **WebViews**.
- The device’s browser shows the WebViews and processes user interactions.

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <WebView
        android:id="@+id/webView1"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

Option2. WebViews

- You need to permit JavaScript code to be executed.
- Finally load in the WebView a local HTML+JS webpage.

```java
webView = (WebView) findViewById(R.id.webView1);
webView.getSettings().setJavaScriptEnabled(true);
webView.setWebViewClient(new WebViewClient());
webView.loadUrl("file:///android_asset/form1_send.html");
```
Section2 - Android OS

Android Mapping Strategies

Option2. WebViews  The following HTML page shows a map centered around Cleveland's downtown. The HTTP call is based on the Google Maps API (See Section1)

```html
<html xmlns="http://www.w3.org/1999/xhtml">
<head> </head>
<body>
<H1> Section1 - Google Maps API Embedd – Search </H1>

<iframe width="600" height="450" frameborder="0" style="border:0"
src= "https://www.google.com/maps/embed/v1/search
?q=restaurants+in+downtown+Cleveland
&key=YOUR-API-KEY"
allowfullscreen>
</iframe>

</body>
</html>
```

Section2 - Android OS

Android Mapping Strategies

Option2. WebViews  The following HTML page shows a map centered around Cleveland's downtown. The HTTP call is based on the Google Maps API (See Section1)
Section 2 - Android OS

Android Mapping Strategies

Option 2. WebViews  The following HTML page shows a map centered on Cleveland.

```html
<html>
<head>
    <title>Simple Map</title>
    <meta name="viewport" content="initial-scale=1.0">
    <style>
        html, body { height: 100%; margin: 0; padding: 0; }
        #map { height: 100%; }
    </style>
</head>
<body>
<h1>Welcome to Cleveland State University</h1>
<div id="map"></div>
<script>
    var map;
    function initMap() {
        map = new google.maps.Map(document.getElementById('map'), {
            center: {lat: 41.502610, lng: -81.675464},
            zoom: 15
        });
    }
</script>
<script src="https://maps.googleapis.com/maps/api/js?callback=initMap &key=YOUR_API_KEY" async defer></script>
</body>
</html>
```

The image below shows the map displayed by the previous HTML page.
Option 3. Native Android Mapping Classes (MapFragment)

Android Studio creates a skeleton native mapping activity. You need to supply your developer’s key. The maps are drawn inside a MapFragment container.

```java
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Sydney and move the camera
    LatLng sydney = new LatLng(-34, 151);
    mMap.addMarker(new MarkerOptions()
        .position(sydney)
        .title("Marker in Sydney"));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
```
Lesson 24

Section 2 - Android OS

Android Mapping Strategies

Option 3: Native Android Mapping Classes (MapFragment)

Base map produced by the Android Studio skeleton activity.

The following are typical actions that you may apply to the map:

- `addCircle(circleOptions options)
- addGroundOverlay(groundOverlayOptions options)
- addMarker(markerOptions options)
- addPolyLine(polyLineOptions options)
- addTileOverlay(tileOverlayOptions options)
- animateCamera(camerUpdate, cancelableCallback callable)
- animateCamera(camerUpdate, int durationMs, cancelableCallback callable)

Section 3 – Vernacular Mapping

Problem Definition

Describe your current location under the following conditions:

- No latitude/longitude coordinates
- No street names
- Tell the user directions in terms of turns and distances.
- At each turn, allow the user to search for landmarks to aid them on their route

- Example 1:
  “The fifth house on the second block west of the Farmer’s Market on your way to the old train station”

- Example 2
  “Starting at the Church of San Pedro drive 500m south (passing the National Bank Building on your right), then 200m west”

NOTE: Vernacular/Colloquial geography – geo-location in ordinary people’s language
Section 3 – Vernacular Mapping

Example: Subject points to her location using CSU as starting point.

I'm here

Use this place as starting reference

Go 0.3m East (you see CSU Garage on your right),

Turn right go South (Midas Shop on the opposite corner),

Drive 0.3m

Turn right (Gas Station on opposite corner), drive 1m West.
**Example:** Subject points to her location using CSU as starting point.

**Directions**
- Go 0.3m East (you see CSU Garage on your right),
- Turn right go South (Midas Shop on the opposite corner), Drive 0.3m
- Turn right (Gas Station on opposite corner), drive 1m West.

**Example B:** Reverse Coloquial Mapping [from vernacular address to map]

Staring at San Pedro’s Church
- go 550 m south (You will see Banco Nacional on your right & Escuela Roosevelt on your left),
- go 200 m West,
- (Destination: Yellow corner house - San Jose, CR)
Thanks for attending this talk.

An edited fragment of a JSON response given to an NEARBYSEARCH request

```
{
  "html_attributions": [],
  "next_page_token": "CoQC-QAAAM_cru-VXipKdKl4jvjZbWm-ypidT1wvouaa4jTM91BbO0e5jMtcjPy-B1r78A",
  "results": [
    {
      "geometry": {
        "location": {
          "lat": 41.50401,
          "lng": -81.67950399999999
        }
      },
      "icon": "https://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png",
      "id": "4b3921ddda6fac643d6f7687f6ea870adec8ead3",
      "name": "Becky's",
      "opening_hours": {
        "open_now": true,
        "weekday_text": []
      },
      "photos": [
        {
          "height": 1230,
          "html_attributions": [],
          "photo_reference": "CmRdAAAAIbJoA5ugSLRI8fFahFmPjnyNYasUBLGmZiw1P09NgAnLUCCOCINNprbha3O0cPEi9fE0Q",
          "width": 2556
        }
      ],
      "place_id": "ChIJWTTYUGP6MIgRZYpqNFgULWY",
      "price_level": 2,
      "rating": 4.6,
      "reference": "CmRaAAAAUZ7rEtvLvvOyuEf8qSx14ILlVgHCfc4XCqxeihOGe70Q_SJzCPrlLNZohyvW2TitCee55vhkCqkgzTTGTwGiD5ANwBd1",
      "scope": "GOOGLE",
      "types": ["bar", "restaurant", "food", "point_of_interest", "establishment"],
      "vicinity": "1762 East 18th Street, Cleveland"
    }
  ]
}
```
Appendix1. Google Places API Web Service - NearbySearch

An edited fragment of a JSON response given to an NEARBYSEARCH request

```json
{
  "html_attributions": [],
  "next_page_token": "CoQC-QAAANe-rvoK20KiclcPdjlwjk6Z_TGDs-poKIFIVccwaSjTBX1b4O9kwj9EIvjFp-Zk1K73A",
  "results": [
    {
      "geometry": {
        "location": {
          "lat": 41.50401,
          "lng": -81.67950399999999
        }
      },
      "icon": "https://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png",
      "id": "4b3921ddda6fac643d6f7687f6ea870adec8ead3",
      "name": "Becky's",
      "opening_hours": {
        "open_now": true,
        "weekday_text": []
      },
      "photos": [
        {
          "height": 1230,
          "html_attributions": [
            "Becky's"
          ],
          "photo_reference": "CmRdAAAAIbJoA5ugSLRI8fFahFmPjnyNYasUBLGmZiw1P09NgAnLUCCOCINNprbha3O0cPEi9fE0Q",
          "width": 2556
        }
      ],
      "place_id": "ChIJWTTYUGP6MIgRZYpqNFgULWY",
      "price_level": 2,
      "rating": 4.6,
      "reference": "CmRaAAAAUZ7rEtvLvvOyuEf8qSx14ILlVgHCfc4XCqxeihOGe70Q_SJzCPrlLNZohyvW2TitCee55vhkCqkgzTTGTwGiD5ANwBd1",
      "scope": "GOOGLE",
      "types": ["bar", "restaurant", "food", "point_of_interest", "establishment"],
      "vicinity": "1762 East 18th Street, Cleveland"
    }
  ]
}
```

Appendix2. Google Places API Web Service - TextSearch

A fragment of the JSON response given to an TEXTSEARCH request

```json
{
  "html_attributions": [],
  "next_page_token": "CvQB7gAAAKP5KlbKChV11YFp_uAuTqiF0OF‐‐y7c7FtV4OM8uj_kI8KbJXlRNs1I4O1lfSAY6HUb‐‐PPBzkTGSzI",
  "results": [
    {
      "formatted_address": "2058 E 4th St, Cleveland, OH 44115, United States",
      "geometry": {
        "location": {
          "lat": 41.498777,
          "lng": -81.69031559999999
        }
      },
      "icon": "https://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png",
      "id": "4f2c84376f1dc3003824c8cf568933adb3afddb2",
      "name": "Lola",
      "opening_hours": {
        "open_now": true,
        "weekday_text": []
      },
      "photos": [
        {
          "height": 1056,
          "html_attributions": [
            "Alicia Salloum"
          ],
          "photo_reference": "CmRdAAAAEPSgpkLo1mfpb9txIQ4jJor1O5z0iq37W3zqeytxIXLBc0plbBDFJxsNyaYbzocWnz2_s0E0_dhdi",
          "width": 1500
        }
      ],
      "place_id": "ChIJXbkJPYD6MIgRbSBLp6BJa18",
      "price_level": 3,
      "rating": 4.4,
      "reference": "CmRXAAAASRsY‐oFaps9o4hPiQCf6ITgbQATClEdWJNpURb_DBNSj04X8jzyxMJC1_bKvLWxkj_TUQG6T1iUAFw",
      "types": ["bar", "restaurant", "food", "point_of_interest", "establishment"],
      "vicinity": "176 East 18th Street, Cleveland"
    }
  ]
}
```
Section 1 - Google Maps API

References

Google Developers Console
https://console.developers.google.com/home/dashboard

Google Maps Javascript API
http://developers.google.com/maps/documentation/javascript/

Google Static Maps API
https://developers.google.com/maps/documentation/static-maps/

Google Maps Roads API
https://developers.google.com/maps/documentation/roads/intro

Google Street View Image API
https://developers.google.com/maps/documentation/streetview/

Google Maps Embed API
https://developers.google.com/maps/documentation/embed/

Google Places API Web Service
http://developers.google.com/places/web-service/

Google Maps Geocoding API
https://developers.google.com/maps/documentation/geocoding/intro

Google Maps Directions API
http://developers.google.com/maps/documentation/directions/

Google Maps Distance Matrix API
https://developers.google.com/maps/documentation/distance-matrix/intro

Google Maps Geolocation API
https://developers.google.com/maps/documentation/geolocation/intro

Google Maps Elevation API
https://developers.google.com/maps/documentation/elevation/intro

Google Maps Time Zone API
https://maps.google.com/maps/api/timezone/outputFormat?parameters

Timestamp Converter
http://www.timestampconvert.com/

CURL Command Line URL
https://curl.haxx.se/

Google Maps
https://support.google.com/maps/answer/146349?hl=en