

Spatial Data Production In Ruby

Keyhole markup language and geocoding

Keyhole Markup Language (KML)

- KML is a structured textual markup language for spatial data.
- An open standard promoted by Google.
- Will look instantly familiar to anyone who's seen HTML.

KML Structure

- KML files require a standard header and footer, just like HTML.
- The header simply references the KML specification, as is customary for XML text.
- There's plenty of room for complexity and customization, but you can make a valid KML file using just these lines.

```
<?xml version="1.0" encoding="UTF-8"?>  
<kml xmlns="http://www.opengis.net/kml/2.2">  
<Document>
```

body goes here

```
</Document>  
</kml>
```

Placemarks

- **The code to the right is a complete, valid KML document that can be displayed by Google Earth and other software.**
- A placemark is the simplest possible KML feature, consisting of a name and a pair of coordinates.
- It represents a point with an attached popup balloon.
- More complex features such as polygons are equally simple to write, and consist of an array of coordinates.
- Descriptions can contain HTML for extra pizzazz, features can be animated, they can have special styles, they can stream from a network, they can only show up at different zoom levels, and so on and so on. But none of that is required.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>

  <Placemark>

    <name>
      Simple placemark
    </name>

    <description>
      A simple placemark description.
    </description>

    <Point>
      <coordinates>
        -77.03394,38.90485,0
      </coordinates>
    </Point>

  </Placemark>

</Document>
</kml>
```

Placemarks

- If we had a list of items with coordinates, such as this list of WDI locations, it would be very simple to write some Ruby to generate the KML.
- You just need to split the text and insert the various attributes where they belong in the text template.
- Keep appending the text to itself, then add the KML header and footer to the result when you're finished.

GA Wash. DC	38.90485,-77.03394
GA New York	40.73930,-73.98942
GA Chicago	41.89061,-87.62688

Placemarks- Make KML with Ruby

This snippet will take the 3 General Assembly location coordinates and print out a complete KML document:

```
def attribs2placemark(name, lat, lng)
  #Insert attributes into placemark template
  placemark = "<Placemark>\n<name>#{name}</name>\n<Point>\n<coordinates>\n#{lng},#{lat},0\n</coordinates>\n</Point>\n</Placemark>\n"
end

#Start with a block of text that has coordinates
raw_data = "GA Washington DC\t38.90485\t-77.03394\nGA New York\t40.73930\t-73.98942\nGA Chicago\t41.89061\t-87.62688"

#Split the text into lines and the lines into attributes, and append a copy of each filled-out placemark template to a string.
kml_body = ""
raw_data.split("\n").each do |line|
  name, lat, lng = line.split("\t")
  kml_body += attribs2placemark(name, lat, lng)
end

#Add kml header and footer to the result
head = "<?xml version='1.0' encoding='UTF-8'?>\n<kml xmlns='http://www.opengis.net/kml/2.2'>\n<Document>\n"
finished_kml = head + kml_body + "</Document>\n</kml>"
puts finished_kml
```

But but I don't have coordinate data

All I have is this list of addresses!

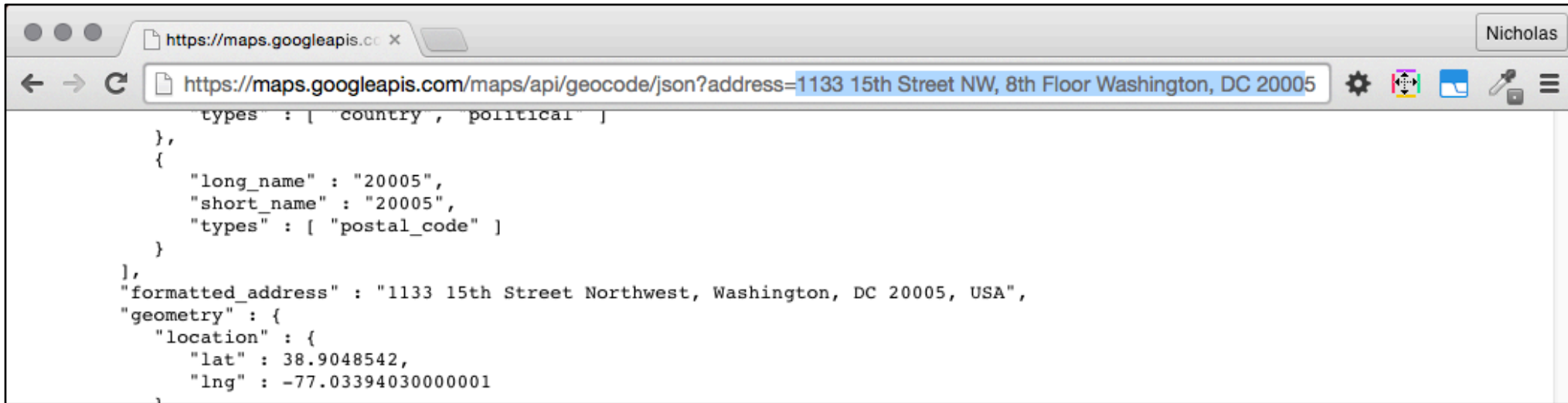
GA Wash. DC	1133 15th Street NW, 8th Floor Washington, DC 20005
GA New York	902 Broadway, 4th Floor New York, NY 10010
GA Chicago	444 N Wabash Ave, 5th Floor Chicago, IL 60611

Geocoding, my friend!

- We can search for text strings in an engine called a geocoder, in order to find their associated coordinates.
- Google provides a web API for this.

Geocoding with Google's API

If we add an address to the end of
`https://maps.googleapis.com/maps/api/geocode/json?address=`

A screenshot of a web browser window. The address bar shows the URL `https://maps.googleapis.com/maps/api/geocode/json?address=1133 15th Street NW, 8th Floor Washington, DC 20005`. The page content displays a JSON response from the API. The JSON is formatted with indentation. The visible portion of the JSON includes:

```
{
  "types": [ "country", "political" ],
},
{
  "long_name": "20005",
  "short_name": "20005",
  "types": [ "postal_code" ]
},
{
  "formatted_address": "1133 15th Street Northwest, Washington, DC 20005, USA",
  "geometry": {
    "location": {
      "lat": 38.9048542,
      "lng": -77.03394030000001
    }
  }
}
```

We get a whole bunch of JSON about the address returned to us.

It looks like the coordinates are in the `["results"] ["geometry"] ["location"]` portion of the return.

Geocoding with Ruby and Google

We can use the snippet below to hit Google's API for each of our addresses, and print out the coordinates we want:

```
require "HTTParty"

def geocode(loc)
  results = HTTParty.get("https://maps.googleapis.com/maps/api/geocode/json?address="+loc)["results"]
  location = results.first["geometry"]["location"]
end

#Start with a block of text
raw_data = "GA Washington DC\t1133 15th Street NW, 8th Floor Washington, DC 20005\nGA New York\t902 Broadway,
4th Floor New York, NY 10010\nGA Chicago\t444 N Wabash Ave, 5th Floor Chicago, IL 60611\n"

new_data = ""
#Split the text into lines and the lines into attributes, and send the address attribute to the geocoder.
Then append the result back into the original text, replacing the address with coordinates.
raw_data.split("\n").each do |line|
  name, addr = line.split("\t")
  #Send the address to the geocoder, and incorporate the result back into a string of text
  coords = geocode(addr)
  new_data += "#{name}\t#{coords['lng']},#{coords['lat']}\n"
end
puts new_data
```

Putting it together

Say, that snippet looked kind of like the first one... let's just combine the geocoder with the KML maker:

```
require "HTTParty"

def attribs2placemark(name, lat, lng)
  placemark = "<Placemark>\n<name>#{name}</name>\n<Point>\n<coordinates>\n#{lng},#{lat},0\n</coordinates>\n</Point>\n</Placemark>\n"
end

def geocode(loc)
  results = HTTParty.get("https://maps.googleapis.com/maps/api/geocode/json?address="+loc)["results"]
  location = results.first["geometry"]["location"]
end

#Input text
raw_data = "GA Washington DC\t1133 15th Street NW, 8th Floor Washington, DC 2005\nGA New York\t902 Broadway, 4th Floor New York, NY 10010\nGA Chicago\t444 N Wabash Ave, 5th Floor Chicago, IL 60611\n"
kml_body = ""
raw_data.split("\n").each do |line|
  name, addr = line.split("\t")
  coords = geocode(addr)
  kml_body += attribs2placemark(name, coords["lat"], coords["lng"])
end

head = "<?xml version='1.0' encoding='UTF-8'?>\n<kml xmlns='http://www.opengis.net/kml/2.2'>\n<Document>\n"
finished_kml = head + kml_body + "</Document>\n</kml>"
puts finished_kml
```

And there you have it: a map

Questions?

