Polyline Reverse Engineering

Start:

Mapbox route-main HTML file.

Things that we noticed when looking through this document (open it in a text editor):

* This file has that huge 20,000 lines of coordinates
* These are stored in a geoJSON structure (the thing with brackets {} everywhere)
* The data type is LineString

So, what do we want to do? Get our own route lines to display.

What we need: [Google Routes API](https://developers.google.com/maps/documentation/routes/overview) … Why? Keep reading.

First stop: [Routes API Demo](https://developers.google.com/maps/documentation/routes/demo)

* This caught my attention since the demo stated PolyLine and geoJSON (what we need!)
* Trying out the demo (putting in addresses) gave a line, but we can’t get the coordinates ☹

Next stop: [Get a Route](https://developers.google.com/maps/documentation/routes/compute_route_directions)

* This gives an example of a POST request (same as we did in ArcGIS I for Google Places)
* After asking chatGPT to make it Python, we created a script to make the API call (polyline.ipynb)
* This errored on first attempt. I learned it was all the extra information starting with “TRAVEL MODE” and stuff that messes up something in the API call.
  + According to [this documentation](https://developers.google.com/maps/documentation/routes/reference/rest/v2/TopLevel/computeRoutes) (very in depth and complicated), all those settings are OPTIONAL :) so I delete them.
* We get a successful call! BUT it returns data in an ‘encoded Polyline’ format (it looks like gibberish)

Next stop: [Polyline decoder](https://developers.google.com/maps/documentation/routes/polylinedecoder)

* If we paste the string of the encoded polyline that was in quotes ‘ ‘, we should get a map route (similar to the API demo results)
* Going back to our ipynb file, let’s change the coordinates to two locations we care about.

Next stop: [Polyline to GeoJSON](https://jsfiddle.net/welesley/tw7qLvh4/2/)

* I couldn’t figure out how to output the data to geoJSON straight from the Google API, so this utility decodes it for us.
* Paste the string of encoded polyline, and we finally have our coordinates in GeoJSON format!

Return to MapBox

* Open file in text editor again
* Delete the old coordinates
  + Start deletion after “geometry”: -----
  + There is going to be some wonky brackets to delete at the bottom
* Paste in our new GeoJSON lineString data.
  + Fix brackets that don’t line up if required
* Run the file, and hopefully it will work!