13.5.3 Map Multiple GeoJSON Points

Now that you have a handle on how to map GeoJSON point type and add data to a popup marker, Basil and Sadhana want you to fetch GeoJSON data from a URL. After all, this is how GeoJSON data is usually accessed, and this is how you will access the earthquake data.

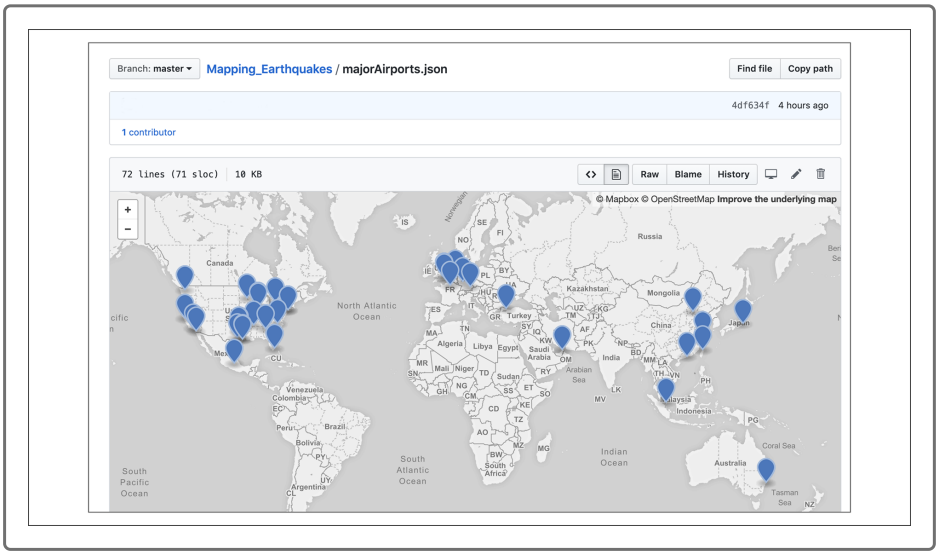
When mapping points, lines, and polygons, the data we use is accessed from a URL because this data is usually inaccessible for download or maybe too large to store on your computer and add as an external file.

Download the `majorAirports.json` file and put it on the Mapping_Earthquakes repository.

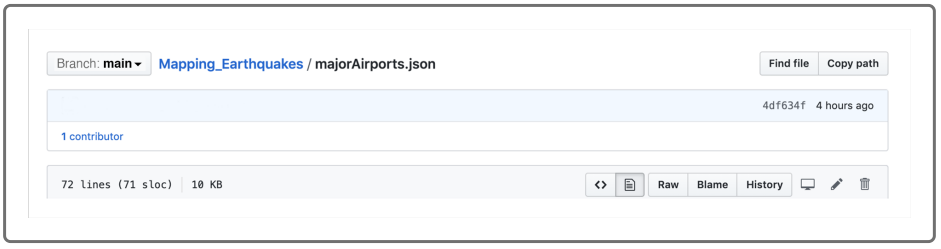
Download majorAirports.json (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_13/majorAirports.json)

Using the URL for the `majorAirports.json` file in your GitHub repository, we'll add multiple points onto a map.

When you click on the `majorAirports.json` file on GitHub, you should see an OpenStreetMap populated with major airports. Our map will look similar to this after we are done.



Click the Raw button and the GeoJSON data will be loaded in the browser.



If the file size is large, it could take awhile to load on the page. Once it loads, it should look like the following:

To begin adding the data to the map, first we need to read the external `majorAirports.json` file.

REWIND

To read an external `.json` file, we need to use the `d3.json()` method. To use the `d3.json()` method, we need to have the `<script src="https://d3js.org/d3.v5.min.js"></script>` file in the `index.html` page.

Open the `index.html` file, and in the `<head>` section above the CSS link, add the following D3.js library file script:

```
<!-- d3 JavaScript -->
<script src="https://d3js.org/d3.v5.min.js"></script>
```

The `<head>` section of your `index.html` file should look like the following:

Next, we'll edit the `logic.js` file.

1. Change the geographical center of the map to the geographical center of the Earth and set the zoom level as follows:

```
// Create the map object with center and zoom level.
let map = L.map('mapid').setView([30, 30], 2);
```

2. Next, we'll access the `majorAirports.json` file on GitHub with the following `airportData` variable. Your URL may be different, but it

should begin with `https://raw.githubusercontent.com`.

3. Add the following code after your `tileLayer()` method:

```
// Accessing the airport GeoJSON URL
let airportData = "https://raw.githubusercontent.com/<GitHub_name>/Mappi
```

NOTE

> Having the `tileLayer()` method before accessing large datasets ensures that the map gets loaded before the data is added to it.

4. Next, we'll add the `d3.json()` method, which returns a promise with the `then()` method and the anonymous `function()`.

    - Inside the `d3.json()` method we'll add the `airportData` variable.
    - Inside the anonymous `function()` we'll add the `data` parameter, which references the `airportData`.
    - We'll pass this `data` to the `L.geoJSON()` layer and then it'll be added to the map with `addTo(map)`.

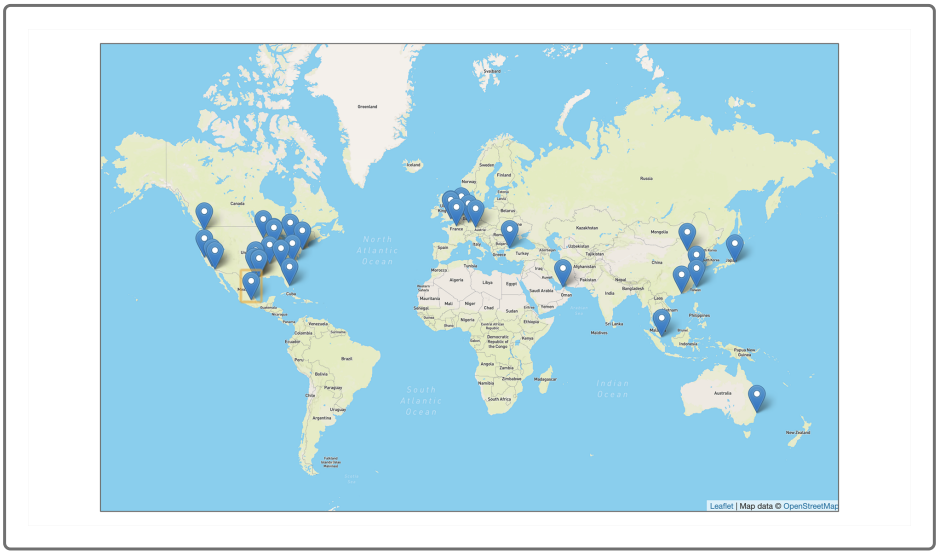
```
// Grabbing our GeoJSON data.
d3.json(airportData).then(function(data) {
    console.log(data);
  // Creating a GeoJSON layer with the retrieved data.
  L.geoJSON(data).addTo(map);
});
```

Your `logic.js` file should look like the following:

```
// Create the map object with center and zoom level.
let map = L.map('mapid').setView([30, 30], 2);

// We create the tile layer that will be the background of our map.
let streets = L.tileLayer('https://api.mapbox.com/styles/v1/mapbox/streets-v11/tiles/{z}/{x}/{y}?access_token={accessToken}', {
    attribution: 'Map data &copy; <a href="https://www.openstreetmap.org/">OpenStreetMap</a> contributors, <a href="https://creativecom
    maxZoom: 18,
    accessToken: API_KEY
});
// Then we add our 'streets' tile layer to the map.
streets.addTo(map);

// Accessing the airport GeoJSON URL
let airportData = "https://raw.githubusercontent.com/         /Mapping_Earthquakes/master/majorAirports.json";

// Grabbing our GeoJSON data.
d3.json(airportData).then(function(data) {
    console.log(data);
  // Creating a GeoJSON layer with the retrieved data.
  L.geoJson(data).addTo(map);
});
```

Let's see how our map looks now. Open your `index.html` file in your browser using the command `python -m http.server`—just to be sure that the data is accessible through the Python server.

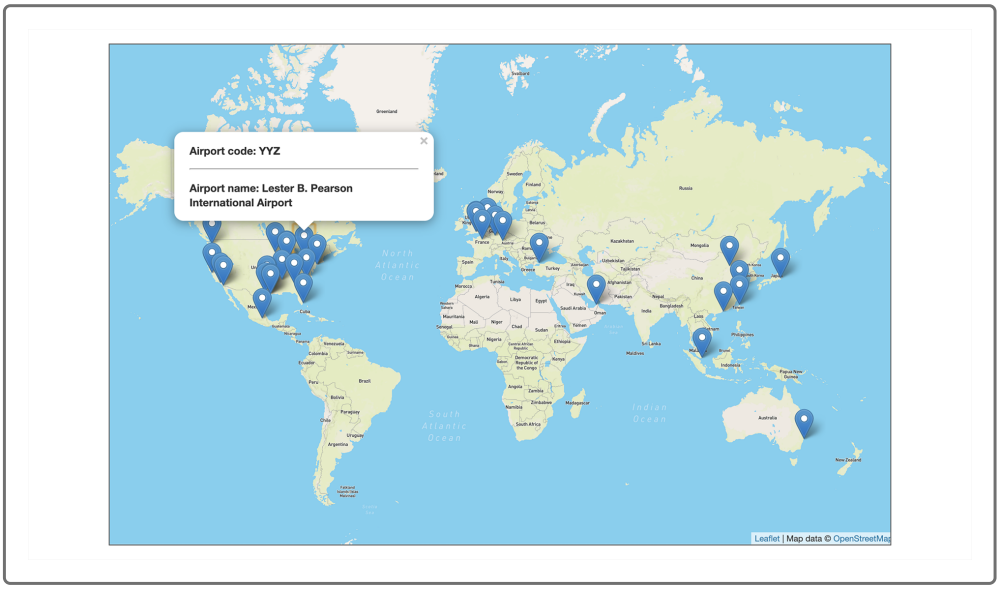
Your map should look like the following:



SKILL DRILL

Edit your `L.geoJson()` layer to add a popup marker that displays all airports' codes and names.

Your map should look like the following:



Great job on adding multiple-point type GeoJSON data to your map. Next, Sadhana is going to show you how to add another map to the `index.html`

file so you can toggle between two different maps.