nscc

PROG 5010 - Front-End Programming - Assignment 2

Transit Tracker

Summary

You will create a solution that will do the following

- Display a map in the browser. (You'll be given starter files for this)
- Fetch real-time transit data information data from a publicly available API. (either Flight or Bus data)
- Filter the raw data to a subset with specific criteria.
- Convert the filtered API data into GeoJSON format.
- Plot markers on the map to display the current position of vehicles.
- Add functionality that will cause the map to auto refresh after a specified interval of time.

General Requirements

Display a Geographical Map

For this assignment you will be working with the Leaflet.js mapping library. Leaflet is a leading open-source JavaScript library for mobile-friendly interactive maps. It provides an easy-to-use programming API for customizing and building various types of maps.

Valuable Leaflet Resources: (it is highly recommended that you check out these resources)

- https://leafletjs.com/reference-1.7.1.html
- https://maptimeboston.github.io/leaflet-intro/ (Read this to get an intro to how Leaflet works!)
- https://www.youtube.com/watch?v=6QFkgOeQc0c&list=PLDmvslp_VR0xjh7wGMNd_1kk0zUox6Sue (The first three videos are particularly relevant)

Fetch real-time transit data

For this assignment, you can choose one of two available data options:

Option 1: Real-time flight data

The first option for real-time transit data that you can leverage for this assignment can be accessed at OpenSky Network (https://opensky-network.org). The OpenSky Network is a community-based receiver network which continuously collects air traffic surveillance data. Unlike other networks, OpenSky keeps the collected raw data forever and makes it accessible to researchers.

- API Endpoint URL: https://opensky-network.org/api/states/all

This API endpoint will return flight data for thousands for aircraft. Your application will need to fetch this data in its raw form and be able to filter the results according to the following criteria.

- Filter requirement: Filter the fetched flight data so that you keep only those aircraft whose country of origin is Canada.

Refer to the OpenSky Network documentation to get an explanation of what data is returned and what format it presents itself.

If you decide to map flight data, work within the provided `flights` folder.

Option 2: Real-time bus data

The second option for real-time transit data that you can leverage for this assignment can be accessed via Halifax Transit open data. Halifax Transit has launched the General Transit Feed Specification (GTFS) open data feed to developers as a beta release. This data is used by Google and other third-party developers to create apps for use by the public.

- API Endpoint URL: https://hrmbusapi.herokuapp.com/

This API endpoint will return real-time data for all buses currently in service throughout HRM. Your application will need to fetch this data in its raw form and be able to filter the results according to the following criteria.

- Filter requirement: Filter the resulting data so that you keep buses on routes 1-10 only.

If you decide to map bus data, work within the provided `buses` folder.

Convert raw API data into GeoJSON format

Leaflet supports and works well with the GeoJSON data format. It is a format for encoding a variety of geographic data structures and is widely used in the digital cartography industry.

You can learn more about GeoJSON from these resources:

- https://macwright.org/2015/03/23/geojson-second-bite.html
- http://geojson.io
- https://www.youtube.com/watch?v=8RPfrhzRw2s

You are required to transform the raw data from your chosen API data (either flights or buses) into GeoJSON format so that they can be applied to the map for point marking. Focus on `Feature arrays` or `Feature Collections` (either will work) when building out your formatted data.

Plot Markers on the map using the converted GeoJSON data

Once you have your newly transformed data in GeoJSON format. Apply this data to the provided map using the programming API for GeoJSON in Leaflet.

Valuable Resources: https://leafletjs.com/examples/geojson/

Apply code to auto-refresh the map

Apply the following functionality to your app which will resemble how real-time transit tracking software behaves:

- After a certain period of time, re-fetch the updated API data and re-perform the GeoJSON transformation as necessary.
- Refresh the map by re-rendering the markers in their new positions.

Note: Simply adding code that forces the entire web page to reload is NOT an acceptable solution for this requirement.

Note: Be careful with this requirement. You should never trigger a re-fetching of your data until you can confirm that any previous fetching of data has been completed and processed. Otherwise, you may cause unintended results within your application.

Other Requirements

The following requirements are considered less critical to your application but will add to your overall mark for the assignment:

- **Custom Vehicle Icon** – Your starter map shows an example of the default marker icon for Leaflet. Update your map to use one of the provided vehicle icons as markers or choose one of your own. (Plane icons if you chose flight data and bus icons if you choose bus data)
- **Rotate Vehicle Icon** – Your API data will include data relevant to the current direction the vehicle is moving relative to True North (0 degrees). Using the provided Leaflet Plugin (leaflet-rotatedmarker.js) Resource: https://github.com/bbecquet/Leaflet.RotatedMarker, rotate each vehicle marker to visually indicate the direction that it is currently travelling.
- **Marker popups** – Leaflet provides the ability to load in data about each marker by leveraging a click event. You could fill this popup with some of the additional data provided by the API and stored as a Property in your GeoJSON feature objects.

Coding Requirements

You must adhere to the following coding requirments for this assignment to meet the learning outcomes:

- Your code must NOT contain for loop structures of any kind. Select from the available array functions that we've been exploring in order to accomplish the goals of the assignment.

Marking Breakdown

Total Marks: 15%

REQ-1: Demonstrate Retrieval of the Required Raw Transit Data (3%)

- Data must be filtered as specified earlier in this document
- You can demonstrate this by console.logging the raw data

REQ-2: Convert Raw Data into GeoJSON format (3%)

- Demonstrate transformation of filtered data into GeoJSON format
- You can demonstrate this by console.logging the GeoJSON data

REQ-3: Plot Markers on Map to Show Position of each Vehicle (3%)

- You can use the default marker for this requirement

REQ-4: Add Auto-Refresh Functionality to the Page (3%)

- All APIs refresh pretty reliably every seven(7) seconds. You may need to tweak this interval a bit.

REQ-5: Additional Functionality (3%)

- Replace the default display icon with a provided vehicle icon (Bus or Plane)
- Correct Icon Rotation to demonstrate the direction that the vehicle is currently traveling.
- Marker popup containing information about the vehicle.

Assessment

1. A student-led demonstration of your code is part of this assignment. This demonstration will be submitted as a video walkthough. Part of the assessment will include your ability to speak about the code you wrote, even if it doesn’t completely work or do what you expect.
2. Late submissions will be subject to the late penalties laid out in the course outline.

Academic Integrity and Plagiarism

Code sharing by any means is considered plagiarism and is strictly forbidden under the NSCC Academic Integrity policy.

NSCC ACADEMIC INTEGRITY GUIDELINES
NSCC ACADEMIC INTEGRITY REPORTING POLICY