PROG 2700 ASSIGNMENT 3 (PARTS A, B, & C) – JS OBJECTS/ARRAYS, & APIS
Client-Side Programming

Part A: FreeCodeCamp

https://learn.freecodecamp.org/

Sign up with your Github account.

Complete the Tutorial Steps in the 'Basic Algorithm Scripting' and 'Functional Programming' Portions under 'JavaScript Algorithms and Data Structures Certification'.

Part B: Map, Filter, and Reduce (Friends TV Show)

Using the starter files (PROG2700_Assignment3B_StarterFiles.zip) complete the main.js script by creating the required functions to make the script work.

You're strongly encouraged to reference the Online Resources and Materials for articles and insight into the Map, Reduce, and Filter functions.

Part C: 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. (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 certain interval of time.

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 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

API

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

This API endpoint will return flight data for thousands of 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.

- **Requirement: Filter the resulting 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.

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 URL:** https://hrmbusapi.herokuapp.com/
- **Backup API URL:** https://hrmbuses.azurewebsites.net

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.

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

GeoJSON

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 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 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 perform the transformation as necessary.
- Refresh the map by re-rendering the markers in their new positions.

Note: adding JavaScript to cause the entire browser page to reload is not an acceptable solution for this requirement.

Note: be careful with this requirement. You should never trigger a re-fetch of your data until the previous fetch has been completed and processed. Otherwise, you may cause unintended results in your application.

Other requirements

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

1. **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)
2. **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 aircraft marker to indicate the direction it is travelling.
3. **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.

Code Requirements

The following requirements are required for this assignment to meet the learning outcomes:

- **Your code must not contain 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.

Requirements (50 points)

REQ-001 DEMONSTRATE RETRIEVAL OF THE REQUIRED RAW TRANSIT DATA (20%)

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

REQ-002 CONVERT RAW DATA INTO GEOJSON (20%)

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

REQ-003 PLOT MARKERS ON MAP TO SHOW EACH VEHICLE (20%)

- You can use the default marker for this requirement.

REQ-004 ADD AUTO REFRESH FUNCTIONALITY TO THE PAGE (20%)

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

REQ-005 ADDITIONAL FUNCTIONALITY (20%)

- Display the provided Vehicle Icon (Plane or Bus)
- Icon Rotation
- Marker Popup containing information about the vehicle.

Instructions

1. **Don't forget that a Code Review demonstration of your code is a necessary part of this assignment. You MUST complete the code explanation/code review part of the video submission checklist to get credit for the assignment. 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. You do not need audio or to speak during the rest of the video, but it is required for the code review section as indicated in the checklist.**

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

PROG2700: Assignment Three - Video Submission Checklist

Part A: FreeCodeCamp	Show the following, <u>if not already given credit for Part A</u>: ☐ in the Browser log into Code Academy and show completion of tasks
Part B: Map, Filter, and Reduce	Show the following: ☐ the program running and displaying desired results in the console ☐ show the code to retrieve data from your chosen API ☐ show that the code is not in global scope (e.g. IIFE usage) ☐ show that the code uses no loop structures of any kind
Part C: Transit Tracker	Show the following: ☐ the program running and showing bus/flight positions on a map on the webpage ☐ that only the desired buses (i.e. Routes 1-10) or flights (i.e. Canadian origin) are shown ☐ the bus/flight indicators use custom images (i.e. of a plane or bus) ☐ that the bus/flight indicators are rotated based on current direction ☐ the bus/flight positions updating repeatedly on the map ☐ that clicking on a particular bus/flight shows a pop-up with information on that particular bus/flight ☐ show that the code is not in global scope (e.g. IIFE usage) ☐ show the code to retrieve data from the chosen API ☐ show that the code uses no loop structures of any kind ☐ citations for any code samples used
Code Review: **<u>Mandatory</u>**	Show the following: **☐explain the code in detail for <u>the entire application for 3C</u> (i.e. querying the API, filtering and preparing the data, plotting it on the map, & updating the map repeatedly) (needs audio)**