17.1 Lesson Summary - Data Visualization with Leaflet

Map based visualizations allow you to demonstrate the relationship your data has with the world around us. There are a number of tools for visualizing map-based data on a webpage including Leaflet and mapbox.

Concept: **Leaflet** is a JavaScript library that allows you to incorporate **map-based** visualizations into your webpages. To incorporate Leaflet into your webpage you should reference a Leaflet CSS file in your webpage's *<head>* and you should reference the leaflet.js file after the Leaflet CSS reference.

* Activity: 01-Ins\_Basic\_Map
* Suppl link: <https://leafletjs.com/>, <https://www.tutorialspoint.com/leafletjs/index.htm>

Concept: After you have added the Leaflet references to your webpage's HTML to **add a map** to your webpage you should declare a *<div>* element with a "map" id. Use Leaftlet to create a map in that *<div>* element and then add a *tileLayer* to add the background map image. Within the tile layer you must reference a mapping service such as mapbox. For example:

*var myMap = L.map("map", {*

*center: [40.7178, -74.0431],*

*zoom: 13*

*});*

*L.tileLayer("https://api.mapbox.com/styles/v1/{id}/tiles/{z}/{x}/{y}?access\_token={accessToken}", {*

*attribution: "© <a href='https://www.mapbox.com/about/maps/'>Mapbox</a> © <a href='http://www.openstreetmap.org/copyright'>OpenStreetMap</a> <strong><a href='https://www.mapbox.com/map-feedback/' target='\_blank'>Improve this map</a></strong>",*

*tileSize: 512,*

*maxZoom: 18,*

*zoomOffset: -1,*

*id: "mapbox/streets-v11",*

*accessToken: API\_KEY*

*}).addTo(myMap);*

* Activity: 01-Ins\_Basic\_Map
* Suppl link: <https://www.mapbox.com/>

Concept: You can add **markers** to your map to provide extra data about a location. For example:

*var marker = L.marker([45.52, -122.67], {*

*draggable: true,*

*title: "My First Marker"*

*}).addTo(myMap);*

* Activity: 02-Ins\_Markers, 03-Stu\_City\_Markers

Concept: You are also able to add various **shapes** to your maps. To create shapes, you must pass coordinates into the corresponding shape functions. To add a circle and a polygon to your map you could use the following code:

*L.circle([45.52, -122.69], {*

*color: "green",*

*fillColor: "green",*

*fillOpacity: 0.75,*

*radius: 500*

*}).addTo(myMap);*

*// Create a Polygon and pass in some initial options*

*L.polygon([*

*[45.54, -122.68],*

*[45.55, -122.68],*

*[45.55, -122.66]*

*], {*

*color: "yellow",*

*fillColor: "yellow",*

*fillOpacity: 0.75*

*}).addTo(myMap);*

* Activity: 04-Ins\_Other\_Markers, 05-Stu\_Other\_Markers, 06-Ins\_City\_Population, 07-Stu\_Country\_World\_Cup

Concept: Adding data to your maps in **layers** can allow a user to toggle between different views and show or hide different data. Leaflet provides 2 different types of layers, **Base Layers** and **Overlays**. Only one Base Layer is visible at the same time making it ideal for core map visualizations like toggling between Google Maps' satellite and street view. Overlays are ideal for elements you may wish to coexist or hide/display individually such as markers. To add a base map layer and an overlay layer to your map you can use the following code:

*L.control.layers(baseLayer, overlayLayer).addTo(myMap);*

* Activity: 08-Ins\_Layers, 09-Stu\_City\_Population\_Layers

Concept: **GeoJSON** can be used to describe geographic data by defining coordinates, regions, shapes, and additional data associated with locations. You can get the features of the GeoJSON data with Leaflet's geoJSON function. For example:

*var features = L.geoJSON(geoJsonData, {*

*onEachFeature: onEachFeature*

*});*

Once you have your features you can add these to your map layers.

* Activity: 10-Stu\_Geo-Json
* Suppl link: <https://geojson.org/>