

Geog 461W - Lab 01

Fall 2016

Instructor: Dr. Alan M. MacEachren

TA: Morteza Karimzadeh

Based on lab instruction prepared by Yanan Xin

Instructions and Deliverables

In this lab, you will:

- Create a tiled webmap and change its styling
- Add features to the map encoded in GeoJSON
- Add popups and style them

Try starting early so that you can ask the TA for help in time. Similar to the bootcamp, you should submit the following in Canvas:

1. The answer to the two questions in red (5 points each).
2. Zipped folder containing completed HTML file and GeoJSON (5 points for finishing until the end of task 3, 10 for finishing task 4, 15 for finishing task 5 and 20 points for finishing until the very end, which is task 6).

Prelude

Make sure you familiarize yourself with a debugger, such as Chrome's Web Developer Tools. The debugger helps you detect mistakes in your code and fine-tune it. I did a quick search on YouTube and found the following videos. Take a look at home, not in class...Use up lab's time to interact with your TA instead.

<https://www.youtube.com/watch?v=KbEx0s06VLs>

<https://www.youtube.com/watch?v=htZAU7FM7GI>

<https://www.youtube.com/watch?v=wcFnnxfA70g>

Lab 01: A Simple Leaflet Map

Task 01

- 1) Download the [Lab01Data.zip](#) and unzip the folder. Rename the FOLDER to your LastNameLab01Folder. You will later zip and upload this folder again.
- 2) Find the Map00.html file in the folder. Rename it to YourLastNameLab1.html.
- 3) Get the latitude and longitude of a place you like from Google Maps. Copy the lat, lon number. e.g. walker building, 40.7932, -77.866
- 4) Open the html file you downloaded using your text editor (any pure text editor would work, such as Sublime Text) and change the **center of the map view** in this file. Hint: line 22.
- 5) Double click the html file and here you go! Well, you've done this in the bootcamp. You created a "tiled webmap" using a tile service (Open Street Maps) and a mapping library (Leaflet). Read the [WikiPedia entry](#) and [MapBox's blog](#) to learn more about (raster) tiled webmaps and how they work.

Question 1

You read about the advantages of raster tiled maps. List at least two disadvantages of a “raster” tiled web map. Is there an alternative that addresses these shortcomings?

Type in your answer in Canvas.

Task 02

Change the background map (tiles). More OpenStreetMap options are [here](#).

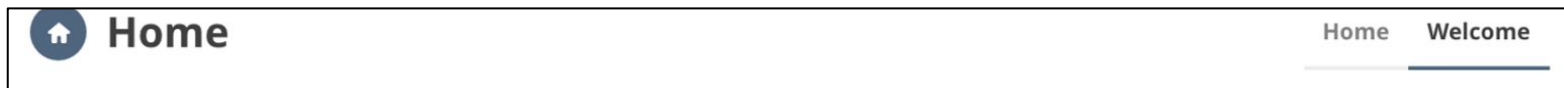
Notice in the text box the change in **Plain JavaScript** when you choose different map styles? Now open the html file you finished from the last task and try to revise the `tileLayer` attributes to display a new map.



Code Explanation: The first parameter of the `L.tileLayer()` function is a URL template for the tile image. The value of `attribution` parameter appears at the bottom-right corner of your map. Always remember to give credit to the map contributors! `maxZoom` attribute indicates the maximum zoom level. Some tile sets only cover certain zoom levels. Setting `maxZoom` (and sometimes `minZoom`) can prevent users from zooming out of the map to an extent for which there is no map data available or from zooming in farther than data allows.

Task 03

We can also design our own map style using Mapbox!

- 1) Sign up for a [mapbox account](#). After you sign up, you will be directed to the quick start part. (If you already have a mapbox account, in your home page, click the welcome tag at the upper-right corner. Like the picture below.)



- 2) Choose the **Design a map** tab. Follow the steps (Pick a template. Make some configurations if you like.) Now go back to your  **Home** page, on the right of your new styled map, click on  next to Edit. Choose `</>` **share, develop & use**. On the right panel, develop with this style, switch to Leaflet and **copy the Leaflet URL**.

Task 03 (continued)

- 3) Now, return to the html file you finished in task 02. Replace the first parameter (the URL template for the tile layer) in the **L.tileLayer()** function with the Leaflet URL you just copied.
- 4) Don't forget to change attribution. Mapbox requires you to use the following

attribution format:

```
print : © Mapbox, © OpenStreetMap
```

```
Code snippet: &copy; <a href="https://www.mapbox.com/about/maps/">Mapbox</a> &copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>
```

- 5) Double click the html file again and see your customized tiled map appear instead!

Task 04

Let's add some “features” to the map. These will be overlayed on top of the tiled (raster) webmap that serves as a background. Map users will be able to interact with those features.

One of the most popular vector data encodings for the web is GeoJSON, which is a special case of JSON. Read on JSON and GeoJSON to familiarize yourself. Let's see how Leaflet handles GeoJSON data.

*To learn how to convert other data formats to GeoJSON, read Supplement 1.

(Converting data will cost another 10-30 minutes depends on the data you choose.)

- 1) The data we're going to use in this task is a subset of reported bike thefts in Philadelphia in 2011. Look in your working folder to see the data (with JSON and JS extensions). (Original data is downloaded from OpenDataPhilly. The data have been cleaned by removing records with 0 lat or lon values.)

Task 04 (Continued)

2) Open the file you downloaded using your text editor. You can see the JSON structure. You can use an online JSON parse such as [this](#) or [this](#) to view the data clearly by copying the contents into the left column of the web pages. As you can see, all “spatial” (coordinates) and aspatial (attributes) data is “encoded” as text in GeoJSON. Can you tell if you are looking at a point, line or polygon dataset?

Let's turn our GeoJSON data into a javascript object (JSON) by declaring a variable name for the object. Open the *Bike_Thefts_sample.geojson* file that you downloaded, add **var bikeThefts =** at the very beginning of the file and add a semicolon **;** at the very end. Save the file with **js extension** with the name *Bike_Thefts_sample.js* and put the file **in the same folder** as your html file (I've already done this for you. Open the files in your text editor and inspect them).

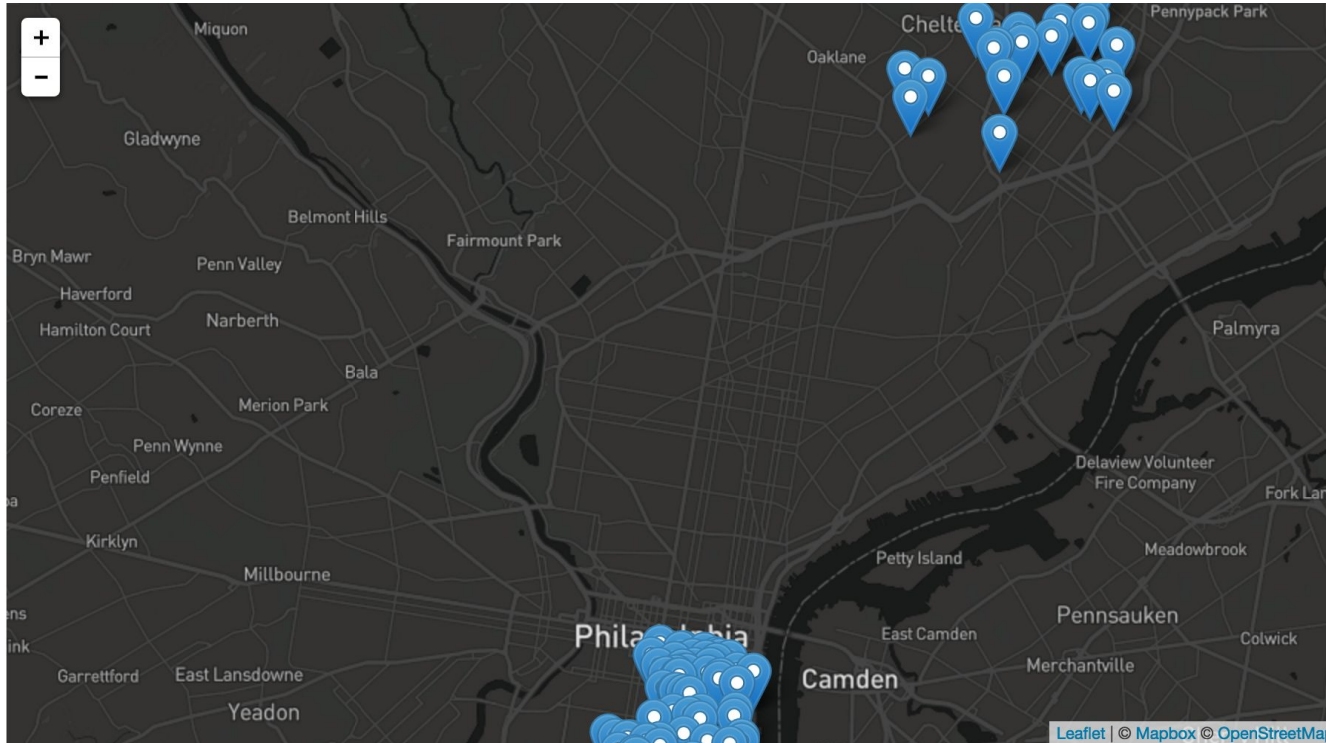
Task 04 (Continued)

3) Edit your HTML file. “Import” the js file as an external js script in the head like below:

```
<script src="http://cdn.leafletjs.com/leaflet/v0.7.7/leaflet.js"></script>  
<script src = "Bike_Thefts_sample.js"></script>
```

4) Then add **L.geoJson(bikeThefts).addTo(map);** after the L.tileLayer function. Double click the html file to see your map.

Task 04 Map



* The sample data are the first 100 bike thefts in 2011, Philadelphia. By accident, the points selected fall into two clusters.

Notes for Task 04

- 1) In relation to methods presented in the bootcamp material, you may have noticed that we used the third way of using js script in an html -- Putting it in a separate document and import it to your HTML document.
- 2) The [API for Leaflet's L.geoJson function](#) is like this: `L.geoJson(<Object> geojson?, <GeoJSON options> options?)`. In the html, we used it like this: `L.geoJson(bikeThefts)`. As you may have noticed we didn't configure any options here for this geojson layer. You can add style for markers to make the map more attractive and interpretable. For more details about using geojson data with Leaflet, check the online tutorial [here](#).

Task 05: Add Popups

Popups are usually used when you want to attach some information to a particular feature on the map that the user can interact with (in this case, click on). We need to use the **onEachFeature** option of the **L.geoJson** function. Check the onEachFeature API [here](#). The function passed to this option will be applied to each feature layer, in our case, to each bike theft data point.

- 1) Start editing your html file.
- 2) First define a function to be passed to onEachFeature. Let's take some time to look at this function. See next slide.

Add this function to the very beginning of your script (right after the `<script>` opening tag).

```
function addPopups(feature, layer) {  
  // does this feature have a property named Location?  
  if (feature.properties && feature.properties.Location) {  
    layer.bindPopup(feature.properties.Location);  
  }  
}
```

```
var bikeThefts = {  
  "type": "FeatureCollection",  
  "features": [{  
    "type": "Feature",  
    "properties": {  
      "DC Number": "12161",  
      "Location": "1600 BLOCK SNYDER AV",  
      "Theft Date": "3/24/11",  
      "Theft Year": "2011",  
      "District": "1",  
      "Stolen Value": "150",  
      "Theft Hour": "18",  
      "UCR": "625",  
      "Latitude": "39.9249462",  
      "Longitude": "-75.17362857"  
    },  
    "geometry": {  
      "type": "Point",  
      "coordinates": [-75.17362857, 39.9249462]  
    }  
  }, {  
    "type": "Feature",
```

The function's name is `addPopups`. The if statement(`if(...){...};`) is to check whether our geojson data includes `properties` as a field and within `properties`, a property named `Location`. See the geojson data attached on the left. We're going to use the `Location` value as popup content. To avoid errors, we need to make sure every point has this field and value (note that this point is named `location`, but really is the address.)

Question 2

Question 2 : Why are we adding the onEachFeature function first thing in the script (instead of in the middle or at the end of script)?

If you do not know the answer, you can try placing the function at the end of your script and see what happens in the debugger when you reload the page.

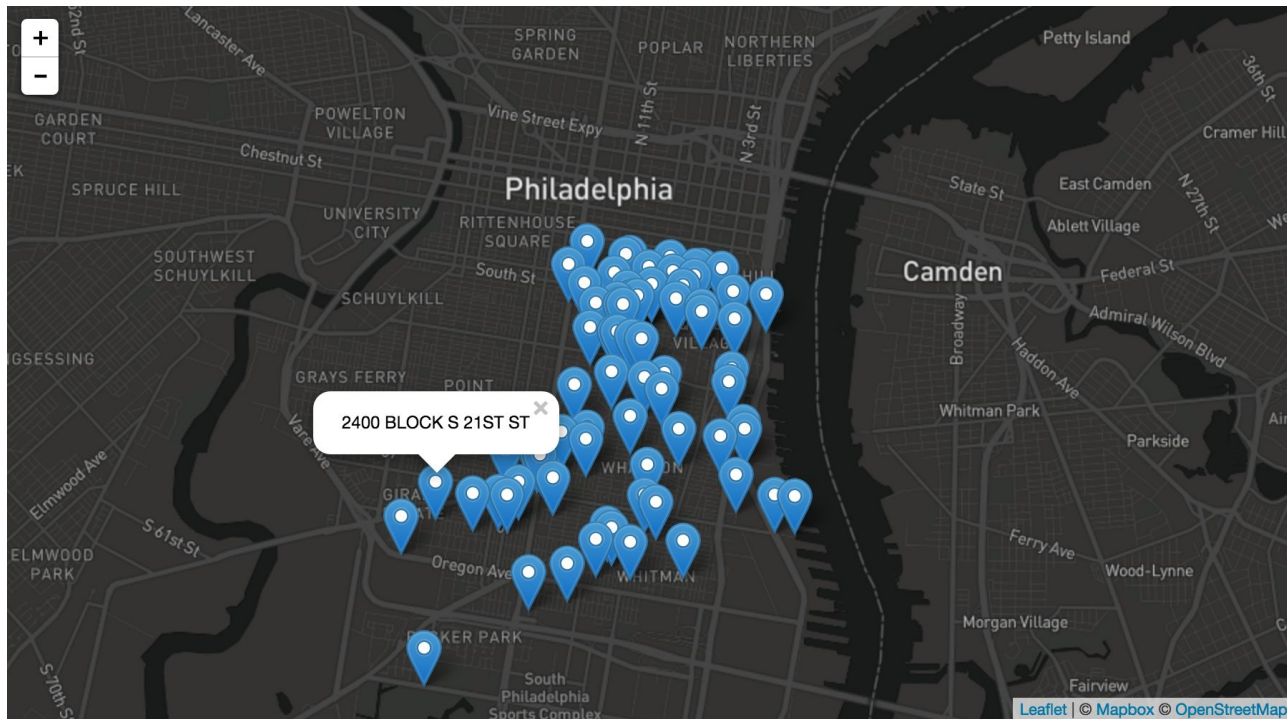
Type in your answer in Canvas.

3) Replace the your L.geoJson function with the new one below:

```
L.geoJson(bikeThefts, {  
  onEachFeature: addPopups  
}).addTo(map);
```

Besides the data `bikeThefts`, we added an option `onEachFeature`, which takes the function `addPopups` as input.

Now you can open the html file in your browser and click on markers to see the popups!



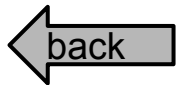
Task 06: Style your marker and popups

Based on the map you finished in Lab01, finish the tasks below:

For reference, check Leaflet's documentation and examples [here](#) and [here](#).

- 1) Change the color and shape of your marker based on an attribute, such as the Theft year. Hint: Add a style option.
- 2) Change the content of the popup to include one more properties of the bike thefts.

Now, zip your LastNameLab01Folder and upload it to Canvas. Don't forget to type your answer the two lab question in the text entry section of Canvas.



Supplement 1

Convert data to GeoJSON:

- 1) [Ogr2Ogr Web Client](#)
- 2) QGIS
- 3) [Shape Escape](#)
- 4) [CSV to GeoJSON](#)
- 5) [Mapshaper](#)
- 6) [Use ogr2ogr via Node](#) (shapefile to GeoJSON and TopoJSON)

*One thing you need to pay attention. **GeoJSON format put longitude first, latitude second for coordinate values.** See a wrongly formatted dataset in next slide.

Bicycle Thefts 9/15/2014 – 2/8/2015 Geojson

Download

URL: <https://data.phila.gov/resource/pbdj-svpx.geojson>

From the dataset abstract

Reported bicycle thefts to the Philadelphia Police Department from the 2010-01-01 to 2013-09-16, and a second set of data for thefts between 2013-09-15 and 2015-02-08. Trouble...

Source: [Bicycle Thefts](#)



Bicycle Thefts Data in Philadelphia

Source:

<https://www.opendataphilly.org/dataset/bicycle-thefts/resource/8bf27c53-0abc-4181-ac17-a40859296599>

Supplement 2

Some useful materials to learn web mapping:

- A great Leaflet tutorial. <http://maptimeboston.github.io/leaflet-intro/>
- A discussion of best online mapping tools.
<https://www.toptal.com/web/the-roadmap-to-roadmaps-a-survey-of-the-best-online-mapping-tools>
- D3 gallery. <https://github.com/d3/d3/wiki/Gallery>