Web Mapping Exercise 1
GISC 27105/37105
GitHub and Text-Based Web Mapping Files

In this lab, you will begin to experiment with the most basic aspects of web cartography. We will be using Leaflet.JS functions to make basic adjustments to a web map that I have already created. The primary goals of this exercise are:

1) to set up a simple framework for creating and saving files that will be publicly accessible;
2) to gain experience reading, understanding, and manipulating HTML documents

This assignment page does not walk you through ALL of the steps needed to complete the assignment, however I have added extensive **comments** in the code to guide you through confusing aspects. Partner up as needed and consult web references if you get lost. NOTE that you can edit the file from your desktop and THEN upload it, but in either case, you will want to use "Developer Tools" or Developer panels of your web browser to help diagnose any problems. For example, in Chrome, right-click on inspect to bring up the page elements, and then select "Console" in the top right panel to see what errors come up.

Getting Started & Organizing Your Directory

- Create a GitHub Account https://github.com/. You will receive a message to confirm your email address once you have signed up.

- While logged into GitHub, you should be prompted to create a **repository**. Create a *public* repository that matches your GitHub user name + ".github.io"
(e.g. if your username is "GISC27105" pick GISC27105.github.io as the name of the repository)

- This new repository will function like a web server. If you put an HTML file here, it will be publicly accessible like any other web page.

A note about organizing your directories and simplifying **Uniform Resource Locators** (URLs)

You can use subfolders to further differentiate among files for various assignments or projects. Also note that any files named "**index.html**" will come up as the default web page when a user views that folder in a browser, with no need to specify an exact file name. For example, if you make separate subfolders for "assignment1" and "assignment2," it is possible to create and save separate, unique files named index.html at both locations, and you could share the URLs as (to continue the above example) GISC27105.github.io/assignment1 and GISC27105.github.io/assignment2, rather than GISC27105.github.io/assignment1/index.html and GISC27105.github.io/assignment2/index.html

- You may add a basic index.html file of any kind to style your "landing page," but we'll be doing work in a specific subfolder…

- Add a subfolder to your repository called "Lab1." At the end of the assignment, you will save your file to this location.

Editing the Map

- Download the file I provided for Exercise 1. You can edit this file directly in a text editor from your desktop OR you can upload it directly to your GitHub "Lab1" folder and make edits there directly. (In any case, make sure to upload the completed Lab 1 file to your GitHub account, under your Lab 1 folder when finished).

Editing files in these assignments

It is sometimes easier to develop an HTML or JS file "locally" – that is, on your computer – *before* publishing it to GitHub (or any web server). You can view files using your web browser even though they're not actually on the Internet! If you need to make an edit, you can simply make and save that change, and then refresh your browser to see it implemented immediately, without waiting for it to register elsewhere. Throughout this course, you may notice a delay of up to several minutes before a file uploaded to GitHub properly appears in its first or newest iteration.

Once you have downloaded and opened the file with a text editor, you should looking at a very simple HTML file that draws upon the Leaflet library to create an interactive map experience for the user. In the head section, two files from Leaflet are referenced, a JavaScript (JS) file and a Cascading Style Sheets (CSS) file. The former gives a predefined list of functionality, like a dictionary of mini programs and functions, so that you can use shorthand code to modify the range of user activities with the map. The latter is a style sheet that accompanies many application programming interfaces (APIs), to help with managing other visual elements on the page like the appearance of things like pop ups or the relative position of layers (for example, why you see a zoom button "in front of" the slippy map on the page). Notice that the full functionality coded into these files is available in your HTML document, and visible through your web browser, simply because these other pages are referenced. You can reference other libraries or APIs in a similar fashion. You can also download these files from their source to modify them further if you need to develop advanced, custom functionality.

Begin by opening your file with a text editor. These are the changes you will make:

- Change the file so that the circle is located over Mansueto Library
- Change the TITLE of the web page to say your name
- Make the mapped area larger by modifying the characteristics of the "div" that contains the map (see Rubric below)
- ADD at least one vertex to the polygon shape
- Add Walker 303 as a point to your map (hint: it's already in the code)
- Change the pop-up text for Mansueto to say "this is also a library!"

- Move the marker from the Midway Plaisance to Crerar Library

- Change out the base map! Here is a list of possible options (note that some are paywalled): https://leaflet-extras.github.io/leaflet-providers/preview/

Additional Notes

Comments are indicated by //. If you begin a line of code with // in an HTML document, the browser will ignore that line of code. This also allows you to quickly "comment out" code you want to remove, without deleting it entirely from your document.

A <**div**> tag in an HTML document defines a "division" or a section of the document that serves as a kind of container within which you can apply styles or fire functions. You will usually apply most of your changes to the div that contains the map.

A <**script**> tag contains scripting elements – JavaScript code – that you can write directly into the HTML document. Most of the functionality changes and programming updates you make will take place within script tags. More simply, in this class, the reference to the API or JS library in the head section normally will preload a library of actions, and then in your own page within the script tags, you will specify HOW those features/actions are fired in the page.

The **base map** is a tiled image service layer (we will cover this in future lectures/assignments), and in this assignment, it is the only tile layer (in Leaflet, this is L.tileLayer). You can change it by swapping out the link AND accompanying map tile details. If you're working from the Leaflet Providers site, it's often easiest to copy out everything between parentheses. Just make sure that you remember to ADD it to the map. In Leaflet, this is done through the .addTo() function. For a div called "mymap" you would write .addTo(mymap).

Remember: after publishing to GitHub, you can still make changes by editing the file directly or by uploading a new file to overwrite an existing one.

Not seeing a change you've made? Try using SHIFT+CTRL+R to do a "hard refresh" after changes

Rubric

☐ A different base map is used (10 points)
☐ Circle feature has been moved to Mansueto (5 points)
☐ Mansueto pop up has been customized (5 points)
☐ Triangle has been modified into a different shape (5 points)
☐ Crerar Library has a marker (5 points)
☐ Title of webpage has been correctly changed (5 points)
☐ Mapped area has been enlarged to at least 1000px width, 800px height (5 points)