Lab 1 - A First Web Map
Due: Monday, January 12, 5:00PM PST

Lab purpose:

At this point in the quarter, we're still putting together the fundamentals of how web mapping emerged and what technologies web maps tend to use. By the end of the course, you'll be able to design and deploy interactive maps that you customize from the basemap to the highest levels of interaction. However, today, we're going to create a 'first' web map using QGIS.

QGIS is an open-source GIS software suite that provides many of the same features as ESRI's ArcGIS. For the purposes of this lab, we'll be making use of the **qgis2web** plug-in. This is a powerful feature that lets you create a ***limited*** web map without having to write any code or much web design.

In this lab, you'll learn how to create simple web maps using QGIS. You'll also (re)familiarize yourself with downloading data and using github to host simple websites. Also, think through what QGIS' GUI can and can't do in terms of creating a web map, what features might you want that aren't offered?

This lab draws heavily from the following tutorial:
https://www.qgistutorials.com/en/docs/3/web_mapping_with_qgis2web.html

Feel free to consult it if you are looking for more guidance.

Lab Deliverables and Grading Scale:

This lab is due **Monday, January 12th, by 5:00pm**.
You turn it in by submitting the http link to your hosted map on canvas (the map is not submitted on canvas, only the link).

For this lab only, there is no grading scale. If you submit a **working** web map that contains point data on top of a base layer as described in the directions, you will receive full credit.

In other words, this is a great opportunity to start this class off on the right foot!

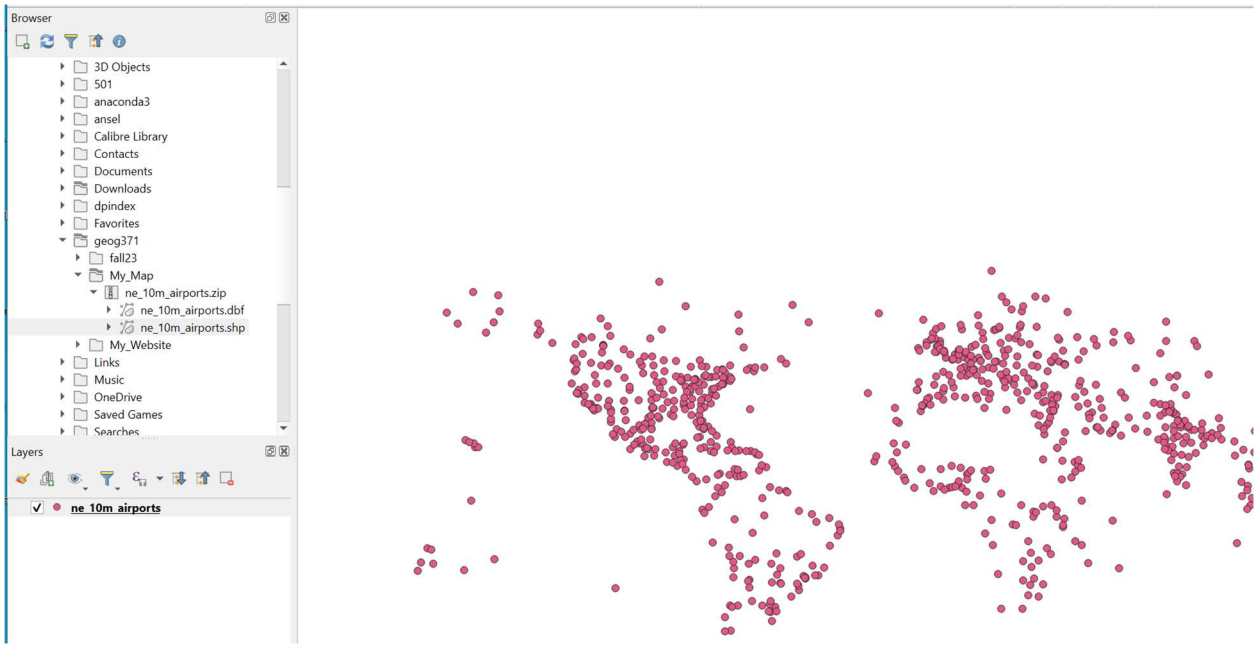
Lab Instructions

A note: *In general, these labs will give you wide leeway to explore. These labs are* ***not*** *step-by-step guides on how to complete each task. If you learn nothing else in this class, gaining the ability to read documentation and trouble-shoot problems will take you further than you might imagine.*

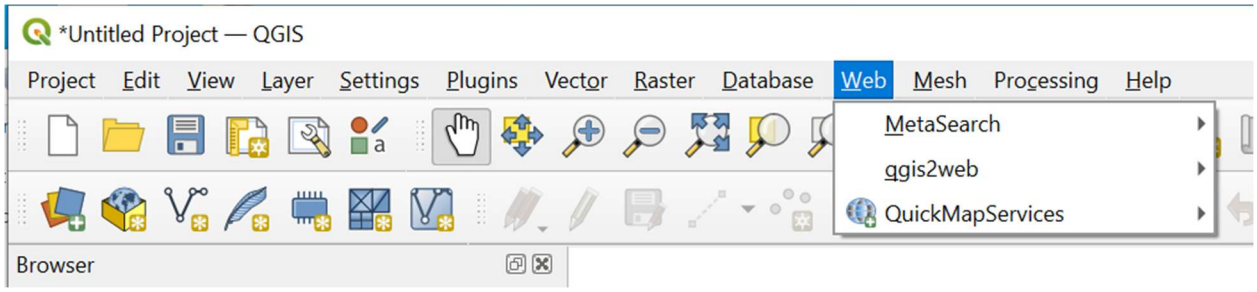
There are certain requirements for each lab, but along the way there are also many opportunities to investigate different directions and to push your own knowledge in new, creative ways. Be creative, trust yourself, and don't be afraid to ***break something*** *- that's why we use version control after all!*

1. First, let's get our **data**. What you'll need is a shapefile with point data. Feel encouraged to make use of **any** appropriate data you wish to use; however, I'll be using the Airports data taken from the Natural Earth vector dataset.
    a. It can be found here: https://www.naturalearthdata.com/downloads/10m-cultural-vectors/airports/
    b. Download it or something else to somewhere that you have write access.

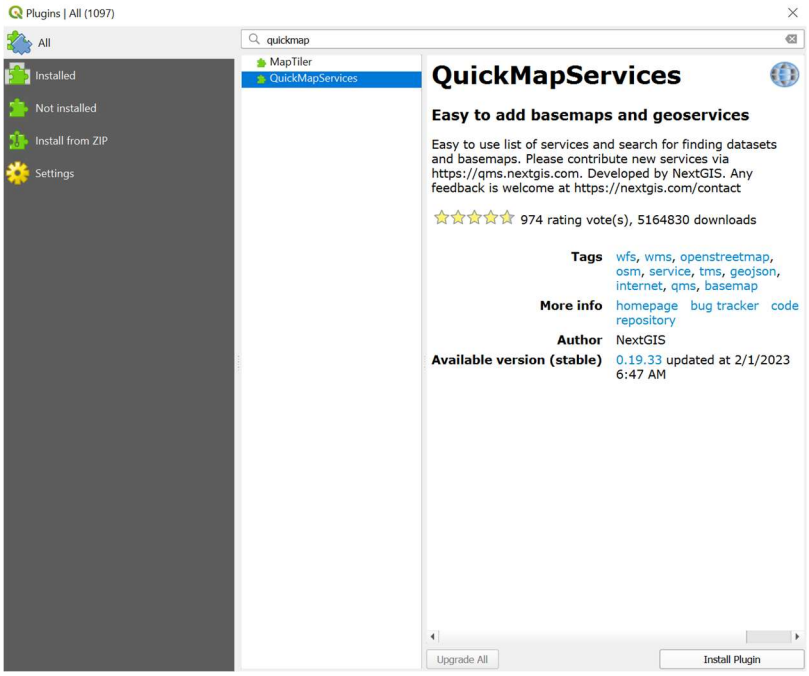
2. Now, let's load that data into QGIS.
    a. In the Browser pane, navigate to where the data is stored.
    b. Expand it, and **drag the .shp file into the display pane**.
    c. You'll end up with something like this:



3. Now, let's add a web-tile background. We'll use a plug-in called QuickMapServices here.
    a. Check to see if it's installed, click the web tab like in the picture below:



    b. If it's there, great, skip to **step 4**. If not, let's install a plug-in!
        i. Click the Plugins tab. And select "Manage and Install Plugins…"
        ii. The following screen should pop up. Fill the search with 'quick map services', select it, and click "Install"



c. As an aside, MapTIler will also come up. It provides **vector** tiles. Feel free to experiment with it if you want.

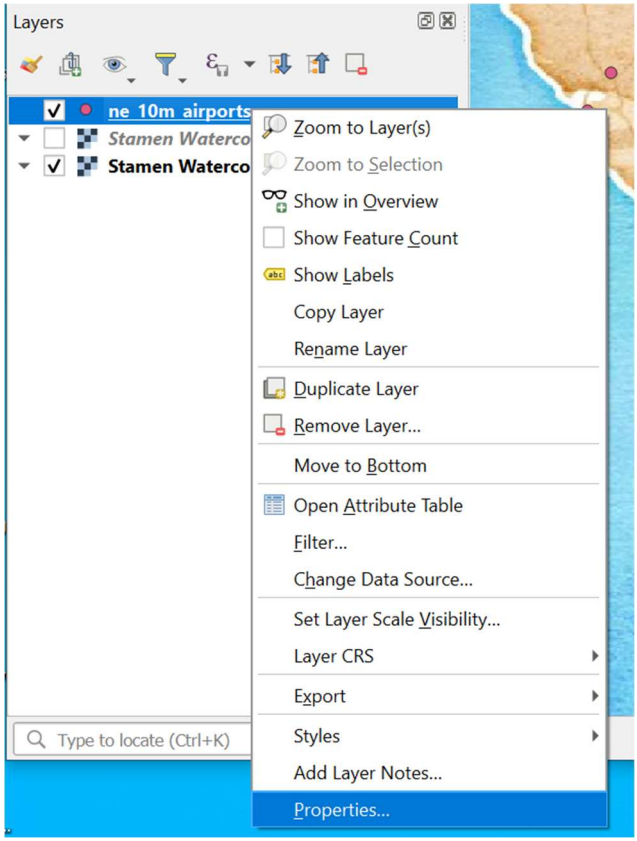
4. QuickMapServices provides basemaps from a number of internet providers. You'll see the default OSM is already there, but why not experiment a bit. Select search and enter 'stamen.' Pick a result and add it to your map. You can always remove it. Here, I've added the Stamen Watercolor (note there are some terms of service on this one that can make it not load. Feel free to use the default OSM if you prefer).

a. You can find out more about QuickMapServices here: https://qms.nextgis.com/#

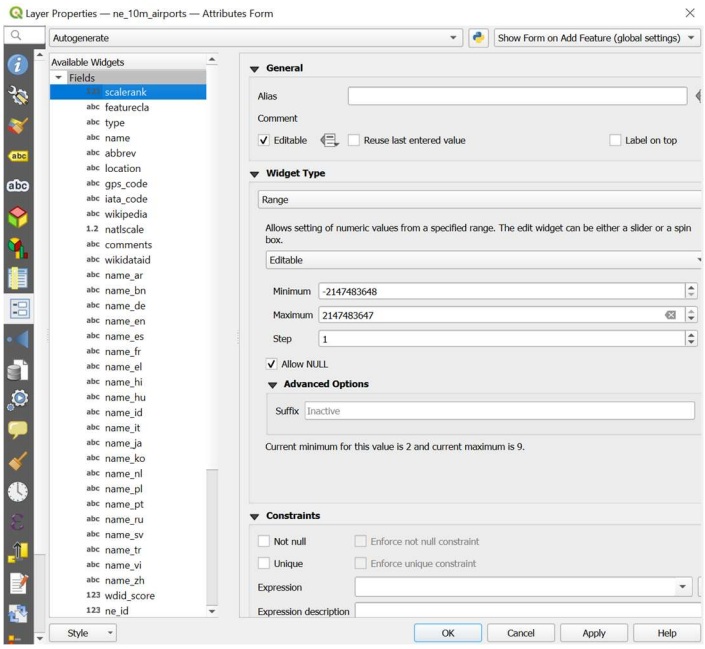
5. Now, we're going to style our map a bit before we use another plugin (qgis2web) to create our web map.
   a. Qgis2web *should* be installed on most of the lab computers, if it is not, **repeat step 3b** but install 'qgis2web' (Plugins are a powerful way to expand qgis and leverage the open-source community that supports it!

Make sure you're comfortable with the process!).

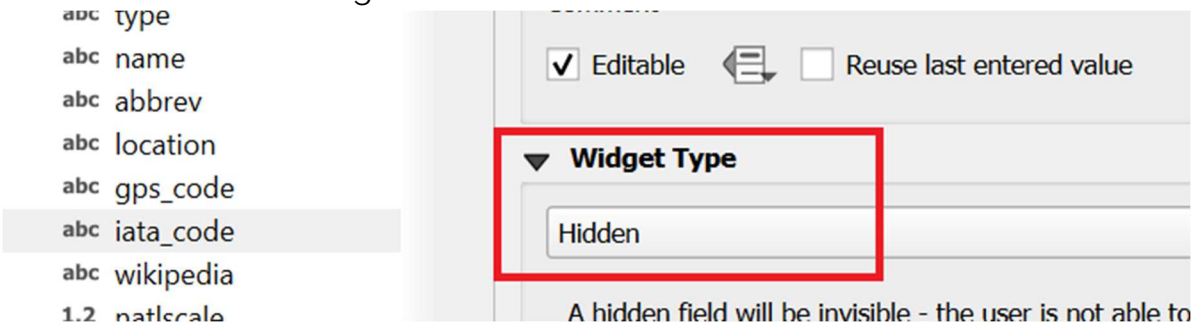
6. With the plugin installed, right click on the ne_10m_airports layer (or whatever data layer you are using) and select properties.



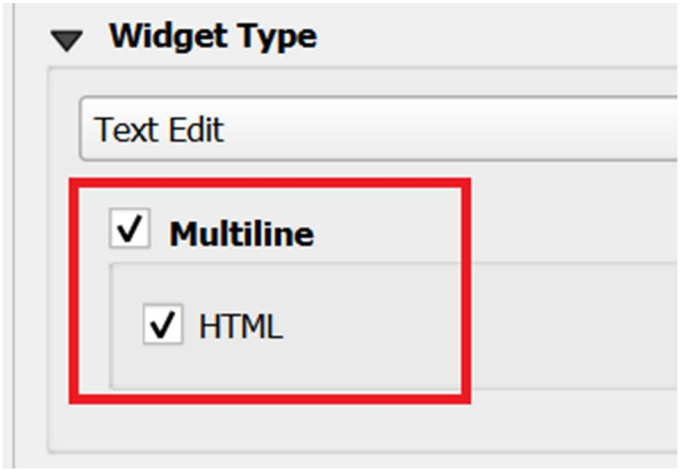
   a. Select the "Attribute Form" tab. You'll get something like this:



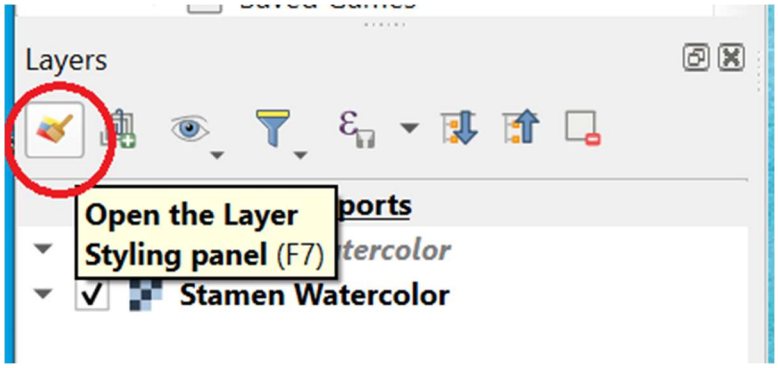
b. Right now, we're setting up what data will display to the user when they click on a point. So, let's turn off everything we don't want.
    i. Set the "Widget Type" to **Hidden** for all fields that you don't wish to display on your web map.
    ii. You must keep: *name, type, and wikipedia*. You *may* keep anything else.
    iii. Yes, this takes awhile. Yes, it's boring. Also, yes, there are other ways to do this; but, this is the least programmatic and most straightforward at the moment.



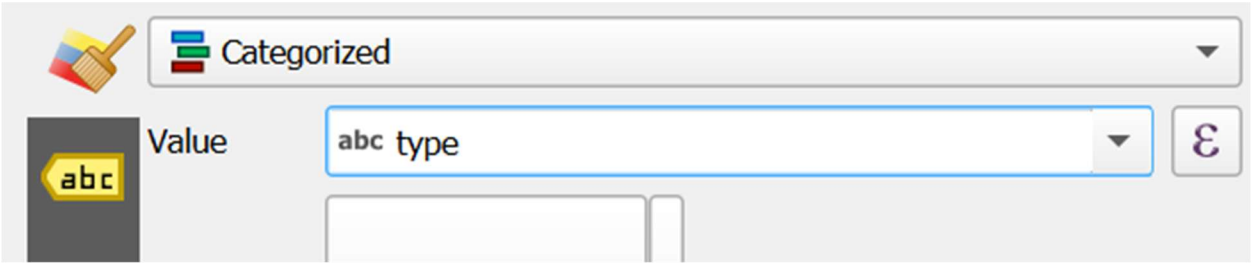
c. For the wikipedia field, make sure you check "multiline" and "html" - this will cause the field to display as a link.



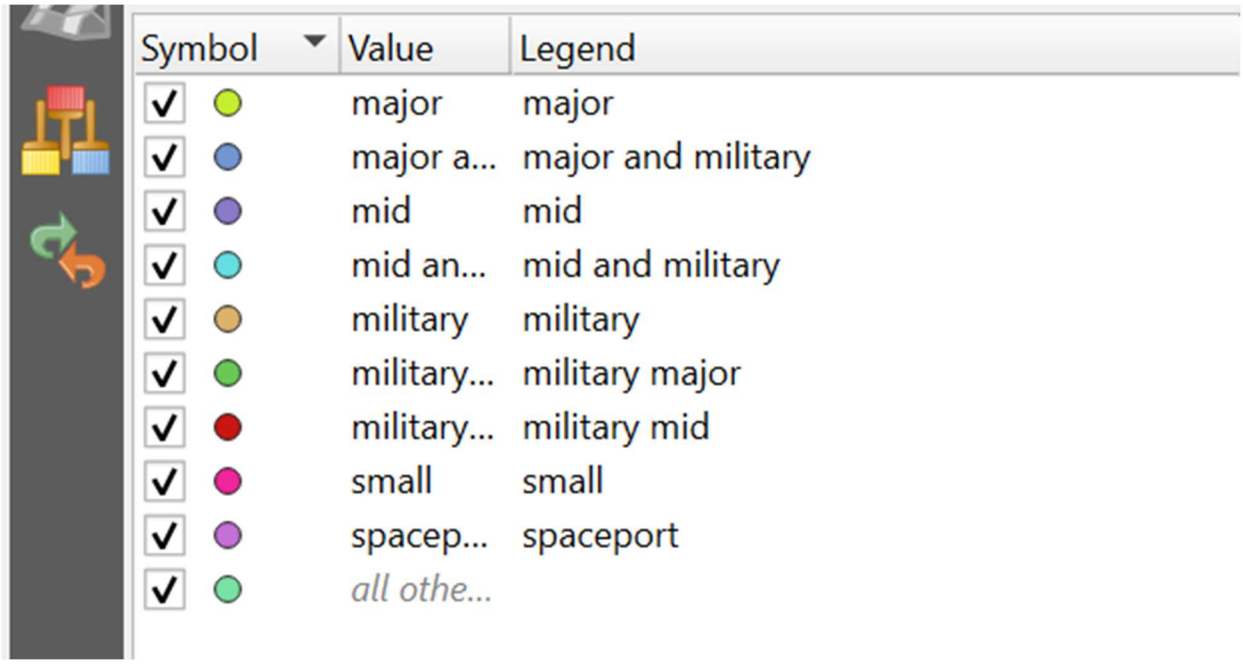
7. Now, let's stylize our map a bit. If you look at the airports layer's properties (right click on it and select properties), you'll see that each airport has been given a 'type'. Let's do some basic thematic mapping and make the different types display differently.
    a. Click the "Layer Styling Panel"



b. Change the type of symbol to "Categorized". For the value, set it as "type." Then click Classify (at the bottom of the pane).



c. You should end up with something like this:

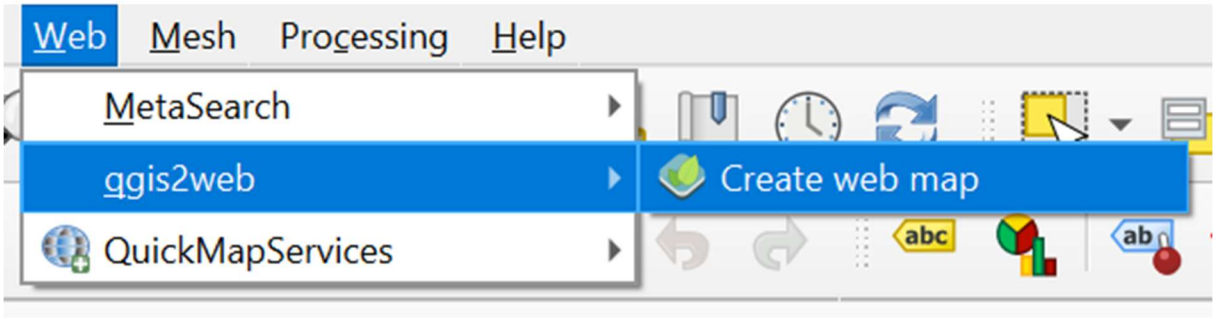


d. Here, you have a bit of leeway. Why not experiment with how the different types of airports are displayed? Do you want to display all the types, or would you prefer to really focus in on a few (for example, only military airports or only civilian). You could, for example, display only

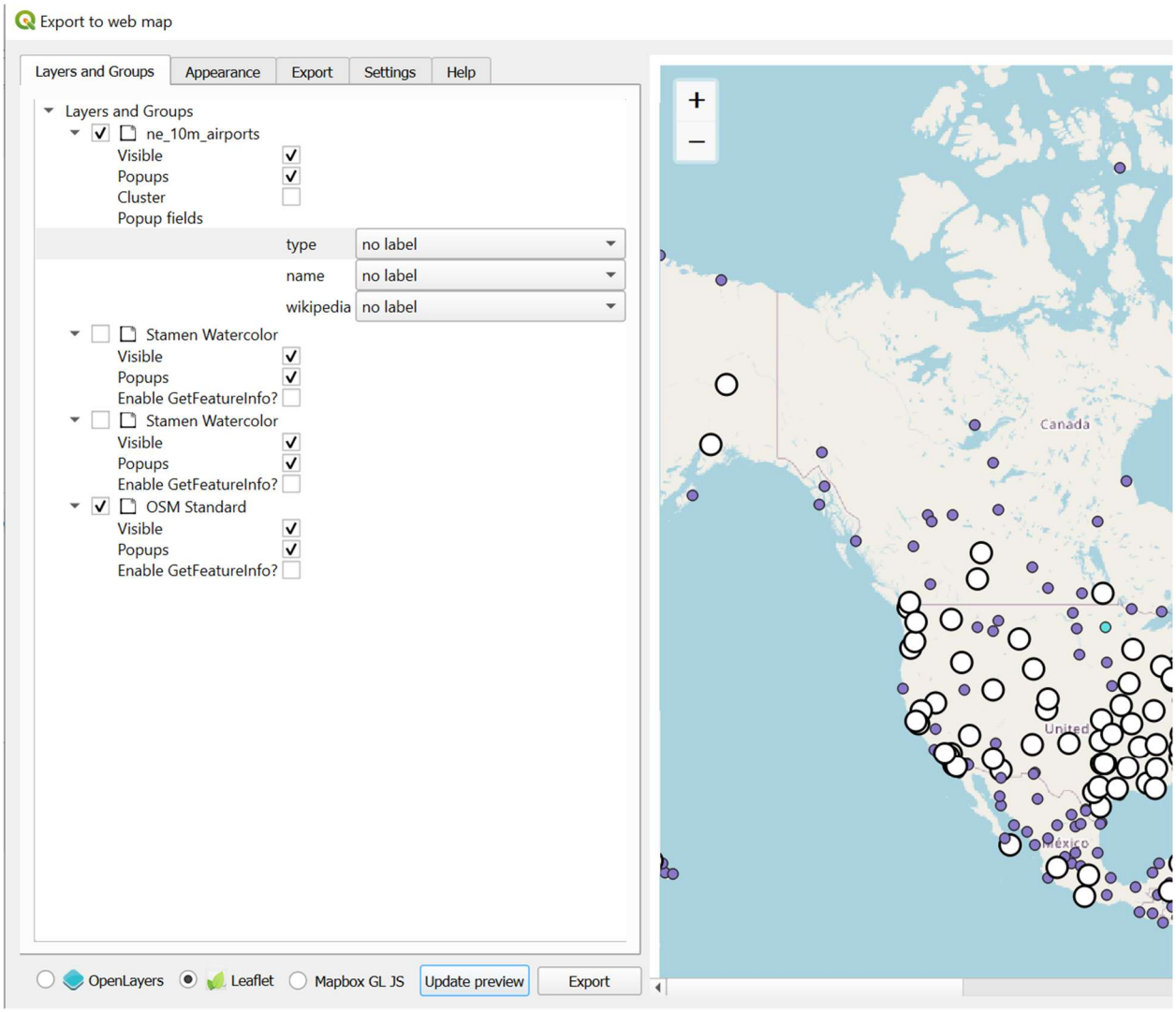
civilian airports and change the size and color of each to reflect their hierarchy of size (major larger than mid which is larger than small…).

e. Turn symbols on and off (the check marks) and click on them to change the individual symbol and its size.
f. Experiment with this until the map contains the data you want it to and it looks the way you want it to.
g. Remember: for this lab, you won't be graded on composition, but you will receive feedback on it. This makes now a great time to practice your cartographic skills.

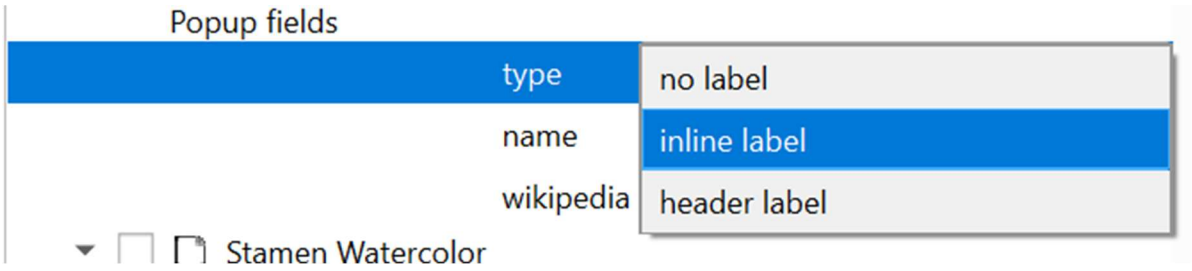
8. Ok, you now have a map that displays *something* (probably airports) on a basemap and the data has been stylized in a way that you find useful/meaningful/interesting/pretty. Let's get this to the web!
    a. Select the web pane and this time choose "qgis2web" - if it's not there, install the plugin just like you did QuickMapServices.



    b. The Export to web dialog box will appear. This is **the** means by which you customize and edit the web layout of your map. On the left are a number of settings you can customize; on the right, is a preview of your map as configured. Use the "update preview" button regularly to see your changes.

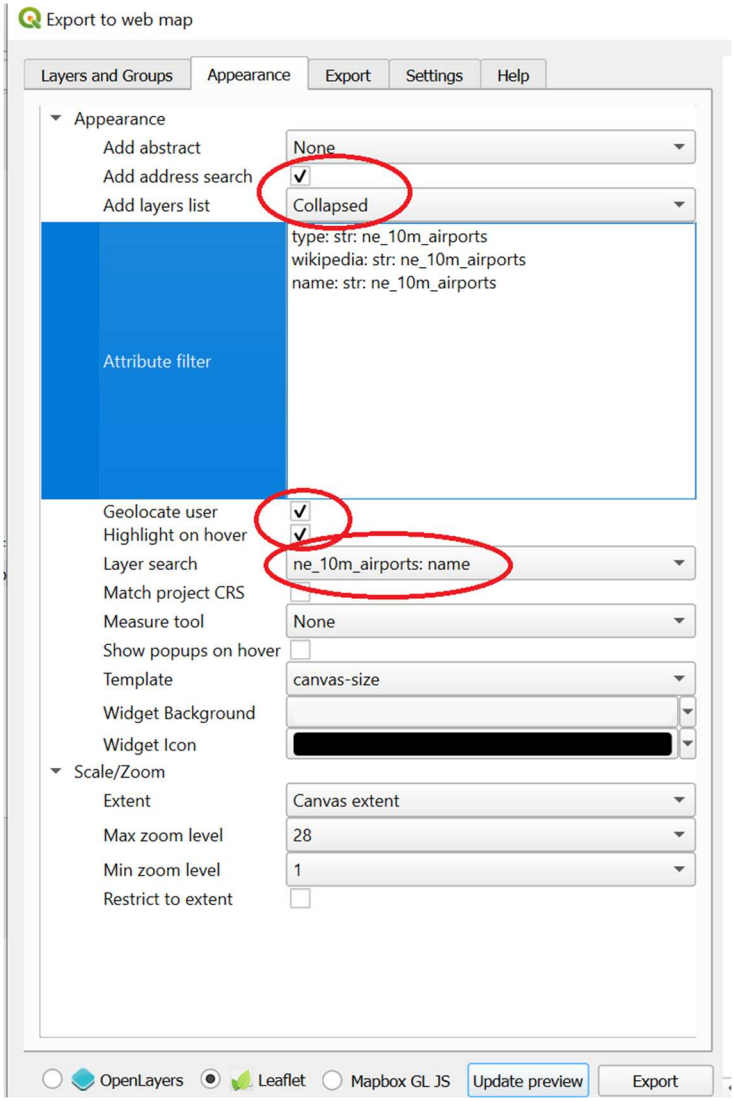


c. Along the bottom you'll see different web mapping libraries that the plugin can use when it exports your map. For this lab, please make sure **leaflet** is selected (though feel free to experiment with the others).
d. Set the popup fields to have inline labels.



e. Now, click on the **Appearance** tab. Check the boxes for address search, geolocate user, and highlight on hover. If you want, set the layer search to the name field (you could also turn iata_code back on in

the layer properties and use that if you prefer - or another property, it's up to you). See the image below for what I did.

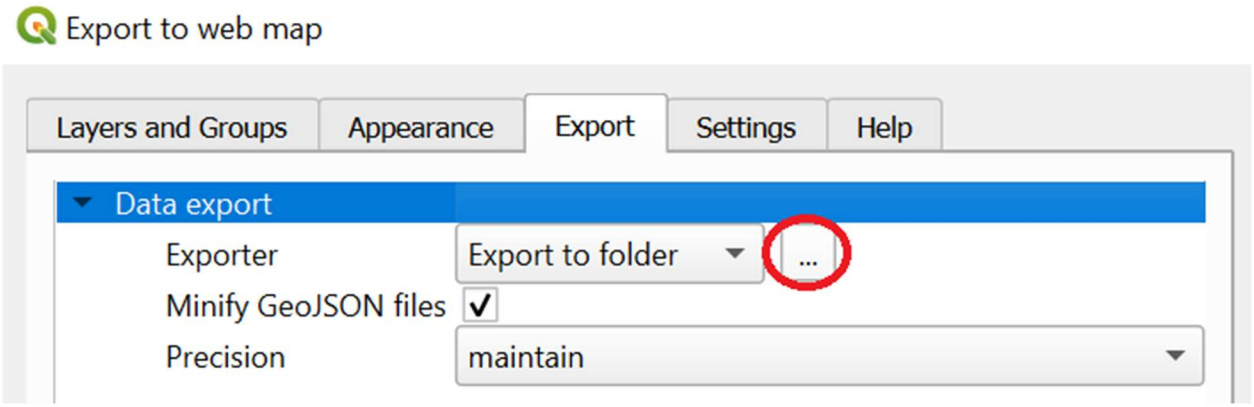


f. **Experiment**. Try different things - change the zoom, turn on pop ups on hover… Add a measure tool and see what that does… There are ***a lot*** of options and you should tweak your map until it behaves (as close to) the way you want it to (as possible).

g. Remember, use **Update Preview** to check what affects your changes are having.

9. When you're satisfied with your map, click on the **Export** tab. Make sure 'Minify geoJSON files' is checked and click the … next to Exporter. This will let you choose where the files are exported. Make sure it's somewhere you have

write access.

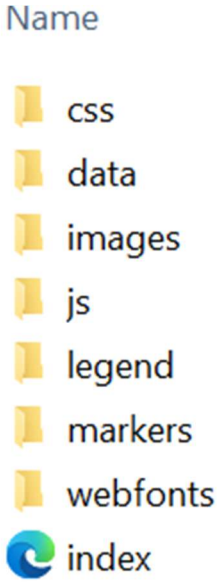


a. Once everything is set, go ahead and click "Export."
b. If the export is successful, your map should open up in a new page in your default web browser.
c. This is an interactive 'web map,' but *right now*, it's being hosted locally on the machine you are using. Let's change that.

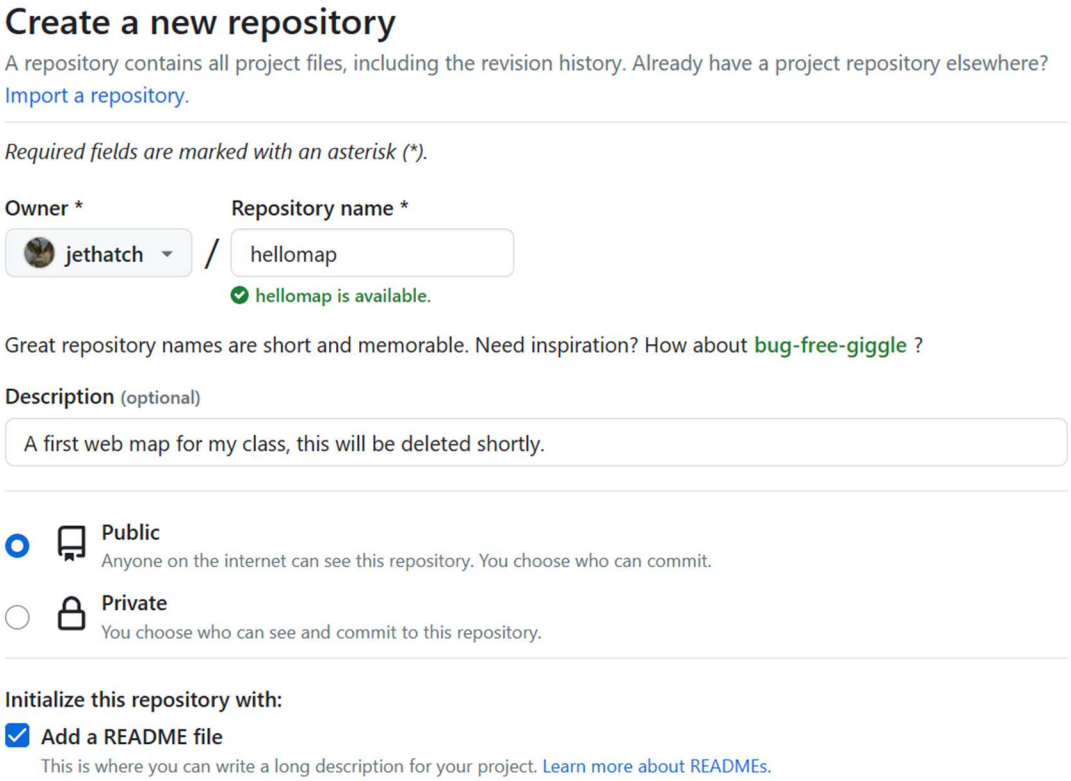
You should be at this point by the end of lab on Monday. *Some* of the material that follows will be explained in class on Wednesday. However, you are welcome to continue working if you feel comfortable (or simply want to experiment! Remember, a core tenet of this class is that it can be very educational to *break things*!). The assigned readings in the workbook also cover much of this material.

What follows requires some familiarity with folder structure and a github account. We will work through both on Wednesday of week 1.

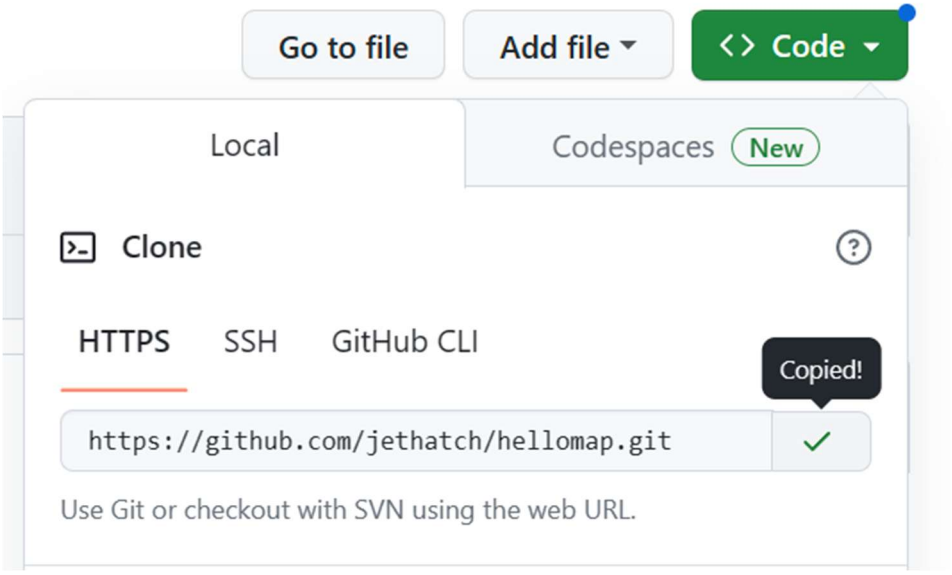
10. Using the file explorer on your machine, navigate to the directory where you exported the map. You'll find something that looks like this:



    a. That looks an **awful** lot like the default web folder example we (will have) looked at in class (on Wednesday) and can be found in your workbook. Yes, there are a few more items (legend, markers, webfonts), but we can see that there's an index.html file and sub-directories that hold various elements of the web page.
    b. We're going to be using the ***same process*** we used to make our first web site in class to upload and host this web map. The directions here will be more terse than the first example; if you are confused, **check the class exercise from Wednesday (or wait for it).**.
    c. The basic process is going to be:
        i. Create a repository on github
        ii. Clone the repository to my workspace.
        iii. Move the web map files to the repository.
        iv. Commit and push the repository back to github.
        v. Turn pages on for the repository.
        vi. Turn the lab in and call it a day.
    d. Go to Github, log in if you aren't, and create a new repository. It should look like this (I recommend you call the repo 'lab1' or something similar).



e. Grab the .git address for the repo



f. Head over to my command prompt (or terminal or…) and "git clone" that repository (you can also do this outside the command prompt if you prefer, but practice makes perfect):

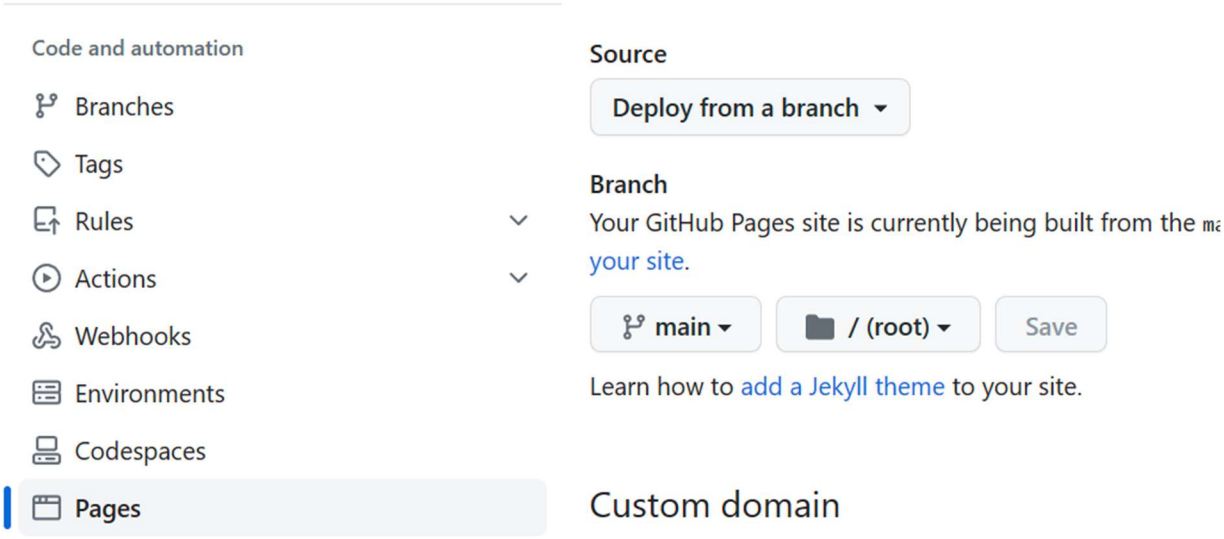
```
C:\Users\jetha\geog371>git clone https://github.com/jethatch/hellomap.git
Cloning into 'hellomap'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

g. Copy my files over into that directory.

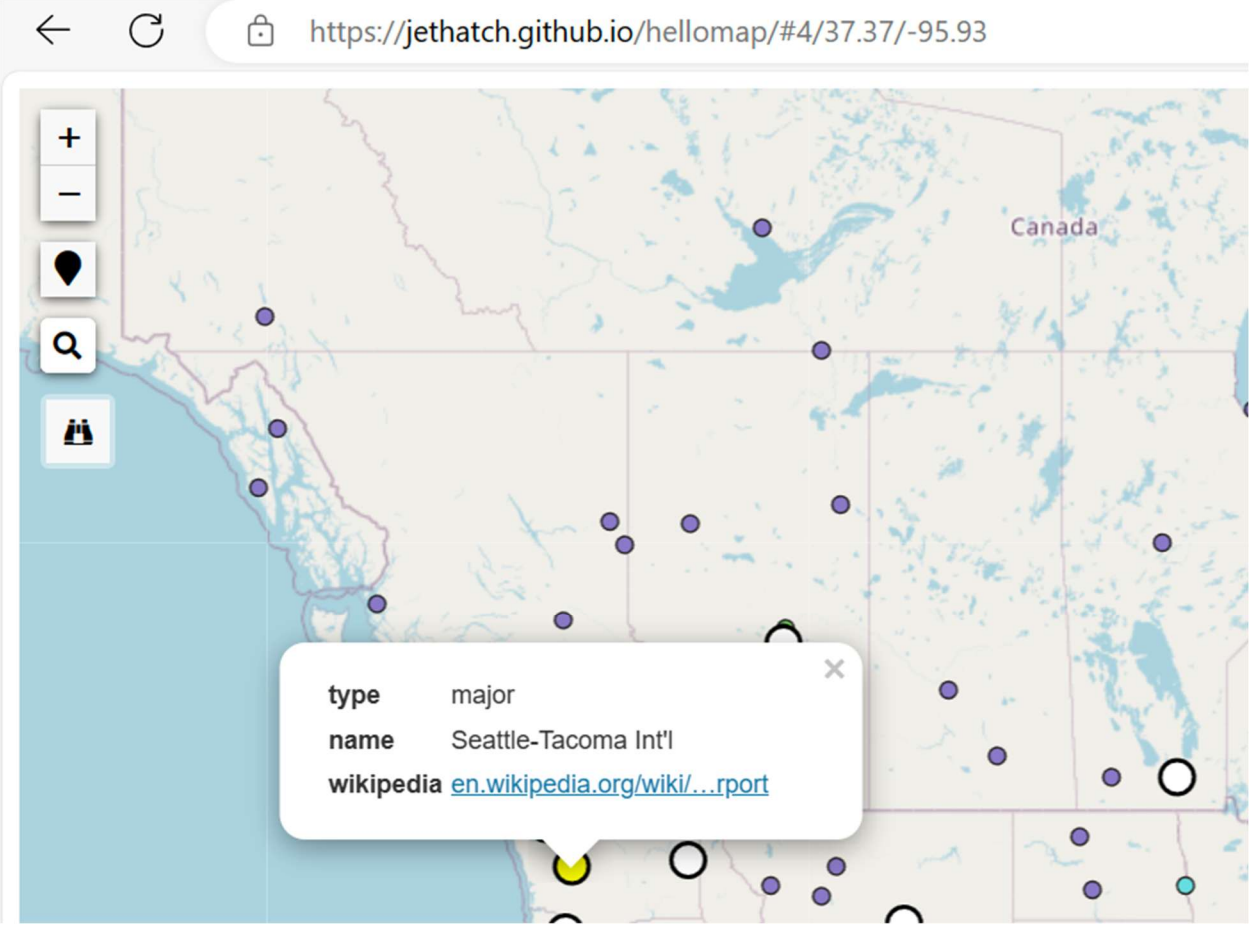
h. **Add** the files ("git add ."), **commit** the files ("git commit -a -m"My first map!"), and then **push** the files. These are separate commands:

```
C:\Users\jetha\geog371\hellomap>git add .
C:\Users\jetha\geog371\hellomap>git commit -a -m"My first map!"
C:\Users\jetha\geog371\hellomap>git push
```

I. Go and turn github pages on for this repository and set the Branch to "main" (check the in class tutorial if you've forgotten how to do this)



Finally, **wait a few minutes** (the most exciting part…) and then go see if the page is hosted. The address should be:
http://[github_username].github.io/[repository_name]
In my case, if I go to http://jethatch.github.io/hellomap, I get this:



SUBMIT A LINK TO YOUR WORKING WEB MAP ON CANVAS BY THE MondayOF WEEK 2 AT 5:00PM PST.