Final Finishing Touches

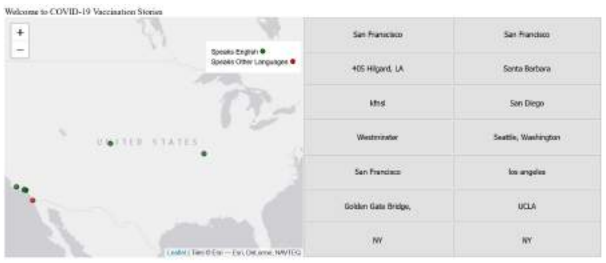
Group Peer Review Exercise

- Group Peer Review Exercise

The goal of this make-up lab is to get more familiar with coding for your final group projects.

Goals

- Touch up our map with a legend
- Revisiting legendary CSS Grid
- Add another new Leaflet plugin


Welcome to COVID-19 Vaccination Stories

San Francisco	San Francisco
405 Hilgard, LA	Santa Barbara
kfnsl	San Diego
Westminster	Seattle, Washington
San Francisco	los angeles
Golden Gate Bridge,	UCLA
NY	NY

Lab outline

1. Legendary Additions!
2. Revisiting CSS Grid
3. Adding another Leaflet plugin
4. Final Lab Code

Starting template code for final lab (lab #9)

index.html

```html
<!DOCTYPE html>
<html>
```

```
    <head>
        <title>Hello World</title>
        <!-- hint: remember to change your page title! -->
        <meta charset="utf-8" />
        <link rel="shortcut icon" href="#">
        <link rel="stylesheet" href="styles/style.css">

        <!-- Leaflet's css-->
        <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />

        <!-- Leaflet's JavaScript-->
        <script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js">
</script>

        <!-- Papa Parse -->
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/PapaParse/5.3.0/papaparse.min.js">
</script>
    </head>

    <body>
        <header>
            Covid Vaccination Stories
        </header>

        <div class="main">
            <div id="contents">
                <div id="placeForButtons"></div>
                <!-- Be sure to use your own survey here!!!!!!! -->
                <div id="theSurvey">
                    <div id="surveyButton">
                        <a
href="https://docs.google.com/forms/d/e/1FAIpQLSfcElv5dlXInR7XHQz27_OcYJlWcIUr-
GBbc-ocefWlGd1uXg/viewform">📝Take the survey</a>
                    </div>
                </div>

            </div>
            <div id="the_map"></div>
        </div>
        <div id="footer">
            Copyright(2023)
        </div>
        <script src="js/init.js"></script>
    </body>
</html>
```

styles/style.css

```
body{
    display: grid;
```

```css
    grid-template-rows: 50px auto auto;
    grid-template-areas: "header" "main_content" "footer";
    background-color: aqua;
    gap: 10px;
}

header{
    grid-area: header;
    justify-self: center;
    align-self: center;
}

#footer{
    grid-area: footer;
}

.main{
    grid-area: main_content;
    grid-template-columns: 1fr 1fr;
    grid-template-areas: "main_map content";
    display: grid;
}

#contents{
    grid-area: content;
    display: grid;
    grid-template-rows: 3fr 1fr;
    grid-template-areas: "buttonHome" "survey"
}

#the_map{
    height:80vh;
    grid-area: main_map;
}

#theSurvey{
    grid-area: survey;
    justify-self: center; /* added this to center the button in the div */
}

/* css for the button */
#surveyButton{
    padding: 15px 32px;
    margin: 10px;
    background-color: #4CAF50;
    cursor: pointer;
}

/* css for button to get rid of the underline */
#surveyButton a{
    text-decoration: none;
}

```

```
#placeForButtons{
    grid-area: buttonHome;
    display:grid;
    grid-template-columns: repeat(2, 1fr);
}
```

js/init.js

```
// declare variables
let mapOptions = {'center': [34.0709,-118.444],'zoom':5}

let vaccinated = L.featureGroup();
let nonVaccinated = L.featureGroup();

let layers = {
    "Vaccinated Respondent": vaccinated,
    "Unvaccinated Respondent": nonVaccinated
}

let circleOptions = {
    radius: 4,
    fillColor: "#ff7800",
    color: "#000",
    weight: 1,
    opacity: 1,
    fillOpacity: 0.8
}

const dataUrl = "https://docs.google.com/spreadsheets/d/e/2PACX-
1vSNq8_prhrSwK3CnY2pPptqMyGvc23Ckc5MCuGMMKljW-dDy6yq6j7XAT4m6GG69CISbD6kfBF0-
ypS/pub?output=csv"

// define the leaflet map
const map = L.map('the_map').setView(mapOptions.center, mapOptions.zoom);

// add layer control box
L.control.layers(null,layers).addTo(map)

let Esri_WorldGrayCanvas =
L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/Canvas/World_
{
    attribution: 'Tiles &copy; Esri &mdash; Esri, DeLorme, NAVTEQ',
    maxZoom: 16
});

Esri_WorldGrayCanvas.addTo(map);

function addMarker(data){
    if(data['Have you been vaccinated?'] == "Yes"){
        circleOptions.fillColor = "red"

vaccinated.addLayer(L.circleMarker([data.lat,data.lng],circleOptions).bindPopup
        createButtons(data.lat,data.lng,data['What zip code do you live in?'])
        }
    else{
        circleOptions.fillColor = "blue"

nonVaccinated.addLayer(L.circleMarker([data.lat,data.lng],circleOptions).bindPo
Vaccinated</h2>`))
        createButtons(data.lat,data.lng,data['What zip code do you live in?'])
    }
    return data
}

function createButtons(lat,lng,title){
    const newButton = document.createElement("button"); // adds a new button
    newButton.id = "button"+title; // gives the button a unique id
    newButton.innerHTML = title; // gives the button a title
    newButton.setAttribute("lat",lat); // sets the latitude
    newButton.setAttribute("lng",lng); // sets the longitude
    newButton.addEventListener('click', function(){
        map.flyTo([lat,lng]); //this is the flyTo from Leaflet
    })
    const spaceForButtons = document.getElementById('placeForButtons')
    spaceForButtons.appendChild(newButton);//this adds the button to our page.
}

function loadData(url){
    Papa.parse(url, {
        header: true,
        download: true,
        complete: results => processData(results)
    })
}

function processData(results){
    console.log(results)
    results.data.forEach(data => {
        console.log(data)
        addMarker(data)
    })
    vaccinated.addTo(map) // add our layers after markers have been made
    nonVaccinated.addTo(map) // add our layers after markers have been made
    let allLayers = L.featureGroup([vaccinated,nonVaccinated]);
    map.fitBounds(allLayers.getBounds());
}

loadData(dataUrl)
```

Last update: 2023-06-01

Legendary Additions!

Let’s start by making our legend not collapsable:

Change the `L.control.layers(null,layers).addTo(map)` on **line 33** to:

```
33  L.control.layers(null,layers,{collapsed:false}).addTo(map)
```

We can do better though and add an actual legend. Notice the `let layers ={}` object right above the `L.control` that we changed. The properties in there `Vaccinated` and `Non-Vaccinated` are actually HTML content that controls how the layers are displayed.

We will add a `<svg>` which is an `svg` element. Our `layers` object should look like the following:

```
let layers = {
    "Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4' 
stroke='black' stroke-width='1' fill='red' /></svg>": vaccinated,
    "Non-Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4' 
stroke='black' stroke-width='1' fill='blue' /></svg>": nonVaccinated
}
```

A much more useful legend should appear. As mentioned before, I’m not a big fan of the Leaflet legend, as there are many more user friendly ways to display a legend, like having the a different `<div>` on the page.

Why are legends important?

They help to provide context into what is represented on the map. This makes sure as we construct our narrative that people know what is represented on the map and do not have to guess what is being shown.

Since legends are so important, many people have implemented different versions of a legend on Leaflet.

Last update: 2023-06-01

Revisiting layouts with CSS Grid

Before we get into adding new plugins, CSS Grid is a powerful way to control how functionality relates to each other. Remember that the flexibility of grid helps to make sure that specific `rows` or `columns` are able to be targeted no matter what plugins we use.

Anytime you see a `display: grid` like the following:

```
body{
    display: grid;
}
```

This means a `CSS grid` is in use there.

Anytime you see `grid-area` that means this HTML Element is within a grid container, for example:

```
#the_map{
    height:80vh;
    grid-area: main_map;
    }
```

At this point, it may help to think of CSS grid as adding grid-lines to a piece of paper, where the piece of paper is our webpage, and grid-area is the content we wish to add to the grid.

In order to put content that fits in our grid-lines we need to make sure the `html content` and `div`s that we want to be aligned are within the areas that the grid covers!

Creating a CSS Grid Legend

Let's revisit our `index.html` and create a new `div` element on the map layer called with an id of `legend`:

index.html

```
        <div id="the_map">
            <div id="legend">
            </div>
        </div>
```

Adding our legendary HTML

Instead of using Leaflet to populate the legend HTML we can directly create the same legend in our HTML while using `div`s to make sure the content doesn't overlap:

index.html

```
    <div>
        Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4'
stroke='black' stroke-width='1' fill='red' /></svg>
    </div>
    <div>
        Non-Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4'
stroke='black' stroke-width='1' fill='blue' /></svg>
    </div>
```

This should all go into the `div` for the `legend` as follows:

index.html

```
        <div id="the_map"></div>
            <div id="legend">
                <div>
                    Vaccinated <svg height='10' width='10'><circle cx='5'
cy='5' r='4' stroke='black' stroke-width='1' fill='red' /></svg>
                </div>
                <div>
                    Non-Vaccinated <svg height='10' width='10'><circle cx='5'
cy='5' r='4' stroke='black' stroke-width='1' fill='blue' /></svg>
                </div>
            </div>
        </div>
```

What Z-heck?

Unfortunately, the Leaflet map has a higher `Z-index` than our **`legend`** div so we need to use CSS to make our `Z-index` higher for the legend. If you think of a webpage as height being y-value, width being x-values, then the stacking of content is controlled by the Z-index.

Basically, a `**Z-index**` is value that controls which layers are on-top of other layers in a webpage.

We will change this in our style.css and make some other nice tweaks in the process:

styles/style.css

```css
#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    position: relative;
}
```

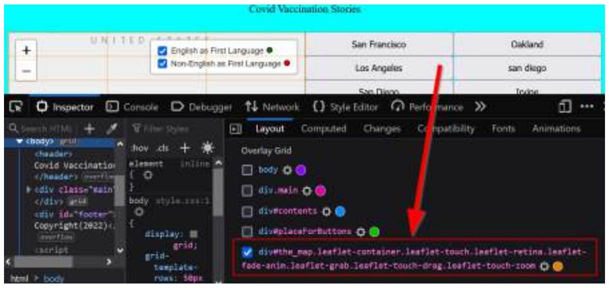
In our `style.css` we will turn the map into a **3 by 3 grid** as follows:

styles/style.css

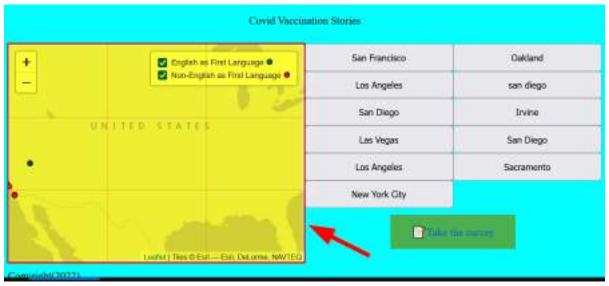
```css
#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
}
```

Reminder: `Inspecting` our grid

Remember, in **FireFox** we can right click, choose `Iinspect element` and in `Inspector` go to `Layout` and check our grid to see it:


Covid Vaccination Stories
English as First Language
Non-English as First Language
San Francisco
Oakland
Los Angeles
san diego
Inspector
Console
Debugger
Network
Style Editor
Performance
Layout
Computed
Changes
Compatibility
Fonts
Animations
Overlay Grid
body
div.main
div#contents
div#placeForButtons
div#the_map.leaflet-container.leaflet-touch.leaflet-retina.leaflet-fade-anim.leaflet-grab.leaflet-touch-drag.leaflet-touch-zoom

When the checkmark is checked, it should look like the following:



Why the `minmax(0,1fr)` instead of just `1fr`?

If you do `1fr` then you run the risk of images and other content overflowing and not respecting the grid when it is empty. You can read more about it here:

- https://css-tricks.com/preventing-a-grid-blowout/

Reminder: Justifying and aligning our content

While in the `#map` lets make sure our content fits in the middle of our grid by using `align-items: center` and `justify-items: center`. Align is used for horizontal alignment and justify is used for vertical adjustments. You can read more about align and justify here:

- https://www.digitalocean.com/community/tutorials/css-align-justify

styles/style.css

```
#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
    align-items: center;
    justify-items: center;
}
```

Alternative positions for align/justify!

Sometimes you'd want to change your vertical alignment and horizontal justification of an item in CSS grid, so here is a table describing some of the common possibilities:

Style	Property	Target	Description
`align-items:`	`start;`	vertical	**top** align an item in the row
`align-items:`	`center;`	vertical	**middle** align an item in the row
`align-items:`	`end;`	vertical	**bottom** align an item in the row
`justify-items:`	`start;`	horizontal	**left** justify an item in the column
`justify-items:`	`left;`	horizontal	**left** justify an item in the column

Style	Property	Target	Description
`justify-items:`	`center;`	horizontal	**middle** justify an item in the column
`justify-items:`	`end;`	horizontal	**right** justify an item in the column
`justify-items:`	`right;`	horizontal	**right** justify an item in the column

For example, if you want the legend in our `css-grid` to be left aligned, you would need to use the following:

styles/style.css

```
#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
    align-items: center;
    justify-items: start;  /* "left" can also work here */
}
```

Positioning our grid HTML elements without `grid-areas`

Instead of naming contents like we have done in the past with `grid-areas` for the `header`, `main_map`, etc., we can generically specify where HTML elements should go using the **`grid-column`** **and** **`grid-row`** CSS attributes.

Let's practice this by adding the legend to the lower right corner of our map 3x3 grid, using the following CSS selector for `#legend`:

styles/style.css

```
#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1; /*! 1 */
    grid-row: 3; /*! 2 */
    position: relative;
}
```

1. `grid-column: 1;` says put this content into the first column!
2. `grid-row: 3;` says put this content into the 3rd row.

Notice how `grid-column: 1` specifies the first column and `grid-row: 3` specifies the last row in our 3x3 grid layout.

Our legendary style should now look like this:

```
#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1;
    grid-row: 3;
    position: relative;
}
```

To span multiple rows or columns ontop of specifying a number you can do `grid-column: 1 / span 2;` which will make the content span 2 columns from the left to right!

styles/style.css

```
#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1 / span 2;
    grid-row: 2;
    position: relative;
}
```

⚽ In-class Exercise #1 - Getting CSS-Griddy with it!

Practice using CSS Grid to change the location of the legend to the top right corner where the current Leaflet legend is. Try to make it span more than 1 column or row.

Tasks

1. Move our custom legend to the top right corner of our `#map` div.
2. Remove the Leaflet Legend in the top right corner
3. **Bonus:** See if you can right align the legend and make it span 3 columns

Answer

styles/style.css

```css
#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
    align-items: center;
    justify-items: center;
}

#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1;
    grid-row: 3;
    position: relative;
}
```

js/init.js

```js
// add layer control box
// L.control.layers(null,layers,{collapsed:false}).addTo(map)
```

Bonus answer:

styles/style.css

```css
#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
    align-items: center;
    justify-items: start;
}

#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1 / span 3;
    grid-row: 1;
    position: relative;
}
```

Adding an `event listener`

If you want the legend to have the same functionality of turning on and off layers, we will need to add an event listener to the legend div with JavaScript.

Remember: an event listener is a function that gets attached to an element when a particular action is done, usually a mouse “click”:

sample event listener

```
function aFunFunction(){
    console.log("i did something fun!")
}

const element = document.getElementById("the_map");
element.addEventListener("click", aFunFunction);
```

This event listener will trigger `aFunFunction` each time the map is clicked!

Let’s add a useful event listener to each of our legend divs, but first we have to give a unique ID to each legend element:

index.html

```
            <div id="legend">
                <div id="vaccinatedLegend">
                    Vaccinated <svg height='10' width='10'><circle cx='5' cy='5'
r='4' stroke='black' stroke-width='1' fill='red' /></svg>
                </div>
                <div id="nonvaccinatedLegend">
                    Non-Vaccinated <svg height='10' width='10'><circle cx='5'
cy='5' r='4' stroke='black' stroke-width='1' fill='blue' /></svg>
                </div>
            </div>
```

Optional: Checkbox!

If you want to completely copy the Leaflet legend style with a check box, you can add the following code in front of the div for the legend:

```
<input type="checkbox" id="uniqueCheckboxID" checked>
```

Checked?

We add the `checked` attribute in `<input type="checkbox" id="uniqueCheckboxID" checked>` to ensure that our check box is checked at the beginning of page load.

The code should look as follows:

index.html

```
            <div id="legend">
                <div id="vaccinatedLegend">
                    <input type="checkbox" id="vaccinatedCheckbox" checked>
                    Vaccinated <svg height='10' width='10'><circle cx='5' cy='5'
r='4' stroke='black' stroke-width='1' fill='red' /></svg>
                </div>
                <div id="nonvaccinatedLegend">
                    <input type="checkbox" id="nonVaccinatedCheckbox">
                    Non-Vaccinated <svg height='10' width='10'><circle cx='5'
cy='5' r='4' stroke='black' stroke-width='1' fill='blue' /></svg>
                </div>
            </div>
```

To make the whole text toggle on and off the checkbox you have to wrap our text and svg in a `label` tag and tell it which checkbox it is `for` using the **`for`-attribute**:

```
    <div id="vaccinatedLegend">
        <input type="checkbox" id="vaccinatedCheckbox">
        <label for="vaccinatedCheckbox">
            Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4'
stroke='black' stroke-width='1' fill='red' /></svg>
        </label>
    </div>
    <div id="nonvaccinatedLegend">
        <input type="checkbox" id="nonVaccinatedCheckbox">
        <label for="nonVaccinatedCheckbox">
            Non-Vaccinated <svg height='10' width='10'><circle cx='5' cy='5'
r='4' stroke='black' stroke-width='1' fill='blue' /></svg>
        </label>
    </div>
```

Lastly, we need to change our event listener in our JavaScript to target the check box ID instead of the legend ID:

js/init.js

```
const vaccinatedLegendHTML = document.getElementById("vaccinatedCheckbox");
const nonvaccinatedLegendHtml = document.getElementById("nonVaccinatedCheckbox");
```

Add Layers/Remove Layers

Since we have feature groups, we can use Leaflet to add or remove them from the map using the following:

`map.removeLayer('Layer I want to remove')`

We will add 2 event listeners, one for each layer and the JavaScript should be as follows:

js/init.js

```
// get the legend HTML checkbox 'vaccinatedCheckbox` to target
const vaccinatedLegendHTML = document.getElementById("vaccinatedCheckbox");

// add the event listener for the click
vaccinatedLegendHTML.addEventListener("click",toggleVaccinatedLayer)

// our function to toggle on/off for english legend's group layer
function toggleVaccinatedLayer(){
    if(map.hasLayer(vaccinated)){
        map.removeLayer(vaccinated)
    }
    else{
        map.addLayer(vaccinated)
    }
}

// target the nonVaccinatedCheckbox div
const nonvaccinatedLegendHtml = document.getElementById("nonVaccinatedCheckbox");

// add the event listener for the click
nonvaccinatedLegendHtml.addEventListener("click",toggleNonVaccinatedLayer)

// toggle the legend for nonvaccinatedLegend grouplayer
function toggleNonVaccinatedLayer(){
    if(map.hasLayer(nonvaccinated)){
        map.removeLayer(nonvaccinated)
    }
    else{
        map.addLayer(nonvaccinated)
    }
}
```

Last update: 2023-06-01

Final Lab Code

Up to this point, your lab code should look like the following:

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Hello World</title>
        <!-- hint: remember to change your page title! -->
        <meta charset="utf-8" />
        <link rel="shortcut icon" href="#">
        <link rel="stylesheet" href="styles/style.css">

        <!-- Leaflet's css-->
        <link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />

        <!-- Leaflet's JavaScript-->
        <script src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script>

        <!-- Papa Parse -->
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/PapaParse/5.3.0/papaparse.min.js">
</script>
    </head>

    <body>
        <header>
            Covid Vaccination Stories
        </header>

        <div class="main">
            <div id="contents">
                <div id="placeForButtons"></div>
                <!-- Be sure to use your own survey here!!!!!!! -->
                <div id="theSurvey">
                    <div id="surveyButton">
                        <a
href="https://docs.google.com/forms/d/e/1FAIpQLSfcElv5dlXInR7XHQz27_OcYJlWcIUr-
GBbc-ocefWlGd1uXg/viewform">📝Take the survey</a>
                    </div>
                </div>
            </div>
            <div id="the_map">
                <div id="legend">
```

```
                    <div id="legend">
                        <div id="vaccinatedLegend">
                            <input type="checkbox" id="vaccinatedCheckbox">
                            <label for="vaccinatedCheckbox">
                                Vaccinated <svg height='10' width='10'><circle
cx='5' cy='5' r='4' stroke='black' stroke-width='1' fill='red' /></svg>
                            </label>
                        </div>
                        <div id="nonvaccinatedLegend">
                            <input type="checkbox" id="nonVaccinatedCheckbox">
                            <label for="nonVaccinatedCheckbox">
                                Non-Vaccinated <svg height='10' width='10'>
<circle cx='5' cy='5' r='4' stroke='black' stroke-width='1' fill='blue' /></svg>
                            </label>
                        </div>
                    </div>
                </div>
            </div>
            <div id="footer">
                Copyright(2023)
            </div>
            <script src="js/init.js"></script>
        </body>
</html>
```

js/init.js

```
// declare variables
let mapOptions = {'center': [34.0709,-118.444],'zoom':5};

let vaccinated = L.featureGroup();
let nonVaccinated = L.featureGroup();

let layers = {
    "Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4'
stroke='black' stroke-width='1' fill='red' /></svg>": vaccinated,
    "Non-Vaccinated <svg height='10' width='10'><circle cx='5' cy='5' r='4'
stroke='black' stroke-width='1' fill='blue' /></svg>": nonVaccinated
}

let circleOptions = {
    radius: 4,
    fillColor: "#ff7800",
    color: "#000",
    weight: 1,
    opacity: 1,
    fillOpacity: 0.8
};

const dataUrl = "https://docs.google.com/spreadsheets/d/e/2PACX-
1vSNq8_prhrSwK3CnY2pPptqMyGvc23Ckc5MCuGMMKljW-dDy6yq6j7XAT4m6GG69CISbD6kfBF0-
ypS/pub?output=csv";
```

```
const vaccinatedLegendHTML = document.getElementById("vaccinatedCheckbox");
const nonVaccinatedLegendHtml = document.getElementById("nonVaccinatedCheckbox");

const map = L.map('the_map').setView(mapOptions.center, mapOptions.zoom);


let Esri_WorldGrayCanvas =
L.tileLayer('https://server.arcgisonline.com/ArcGIS/rest/services/Canvas/World_Light
{
    attribution: 'Tiles &copy; Esri &mdash; Esri, DeLorme, NAVTEQ',
    maxZoom: 16
});

Esri_WorldGrayCanvas.addTo(map);

// add layer control box
// L.control.layers(null,layers,{collapsed:false}).addTo(map)

function addMarker(data){
    if(data['Have you been vaccinated?'] == "Yes"){
        circleOptions.fillColor = "red"

vaccinated.addLayer(L.circleMarker([data.lat,data.lng],circleOptions).bindPopup(`<h2
        createButtons(data.lat,data.lng,data['What zip code do you live in?'])
        }
    else{
        circleOptions.fillColor = "blue"

nonVaccinated.addLayer(L.circleMarker([data.lat,data.lng],circleOptions).bindPopup(`
Vaccinated</h2>`))
        createButtons(data.lat,data.lng,data['What zip code do you live in?'])
    }
    return data
}

function createButtons(lat,lng,title){
    const newButton = document.createElement("button"); // adds a new button
    newButton.id = "button"+title; // gives the button a unique id
    newButton.innerHTML = title; // gives the button a title
    newButton.setAttribute("lat",lat); // sets the latitude
    newButton.setAttribute("lng",lng); // sets the longitude
    newButton.addEventListener('click', function(){
        map.flyTo([lat,lng]); //this is the flyTo from Leaflet
    })
    const spaceForButtons = document.getElementById('placeForButtons')
    spaceForButtons.appendChild(newButton);//this adds the button to our page.
}

function loadData(url){
    Papa.parse(url, {
        header: true,
        download: true,
        complete: results => processData(results)
    })
};

function processData(results){
    console.log(results)
    results.data.forEach(data => {
        console.log(data)
        addMarker(data)
    })
    vaccinated.addTo(map) // add our layers after markers have been made
    nonVaccinated.addTo(map) // add our layers after markers have been made
    let allLayers = L.featureGroup([vaccinated,nonVaccinated]);
    map.fitBounds(allLayers.getBounds());
};

loadData(dataUrl)

// toggle the legend for vaccinatedLegend grouplayer
vaccinatedLegendHTML.addEventListener("click",toggleVaccinatedLayer)

function toggleVaccinatedLayer(){
    if(map.hasLayer(vaccinated)){
        map.removeLayer(vaccinated)
    }
    else{
        map.addLayer(vaccinated)
    }
}

// add the event listener for the click
nonvaccinatedLegendHtml.addEventListener("click",toggleNonVaccinatedLayer)

// toggle the legend for nonvaccinatedLegend grouplayer
function toggleNonVaccinatedLayer(){
    if(map.hasLayer(nonvaccinated)){
        map.removeLayer(nonvaccinated)
    }
    else{
        map.addLayer(nonvaccinated)
    }
}
```

styles/style.css

```
body{
    display: grid;
    grid-template-rows: 50px auto auto;
    grid-template-areas: "header" "main_content" "footer";
    background-color: aqua;
    gap: 10px;
}
```

```css
header{
    grid-area: header;
    justify-self: center;
    align-self: center;
}

#footer{
    grid-area: footer;
}

.main{
    grid-area: main_content;
    grid-template-columns: 1fr 1fr;
    grid-template-areas: "main_map content";
    display: grid;
}

#contents{
    grid-area: content;
    display: grid;
    grid-template-rows: 3fr 1fr;
    grid-template-areas: "buttonHome" "survey"
}

#the_map{
    height:80vh;
    grid-area: main_map;
    display: grid;
    grid-template-columns: repeat(3, minmax(0, 1fr));
    grid-template-rows: repeat(3, minmax(0, 1fr));
    align-items: center;
    justify-items: right;
}

#legend{
    z-index: 9999;
    background-color: white;
    padding: 10px;
    grid-column: 1 / span 3;
    grid-row: 1;
}

#theSurvey{
    grid-area: survey;
    justify-self: center; /* added this to center the button in the div */
}

/* css for the button */
#surveyButton{
    padding: 15px 32px;
    margin: 10px;
    background-color: #4CAF50;
    cursor: pointer;
}

/* css for button to get rid of the underline */
#surveyButton a{
    text-decoration: none;
}

#placeForButtons{
    grid-area: buttonHome;
    display:grid;
    grid-template-columns: repeat(2, 1fr);
}
```

Last update: 2023-06-01

Final In-class exercise Prep

For the in-class exercise, we will use the `git practicing repo` located here:

- https://github.com/albertkun/23S-ASIAAM-191A-Git-Practicing/

The `git` link to clone is here:

```
https://github.com/albertkun/23S-ASIAAM-191A-Git-Practicing.git
```

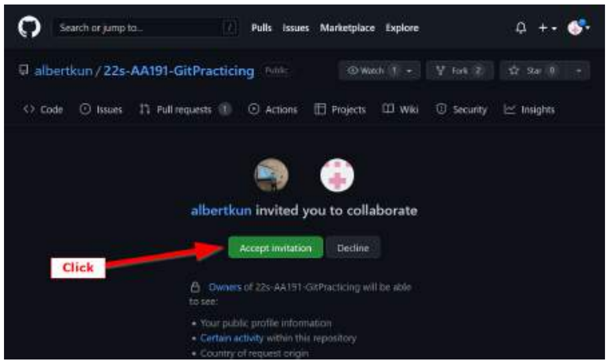
This is similar to what you would do for cloning your group projects if you have not done so already.

Accept the invite to collaborate on the repo

You should have recieved an email to collaborate on the repository already, if not click the link below while logged into your GitHub account:

https://github.com/albertkun/23S-ASIAAM-191A-Git-Practicing/invitations

Then click the accept invite button:



albertkun invited you to collaborate

Owners of 22s-AA191-GitPracticing will be able to see:

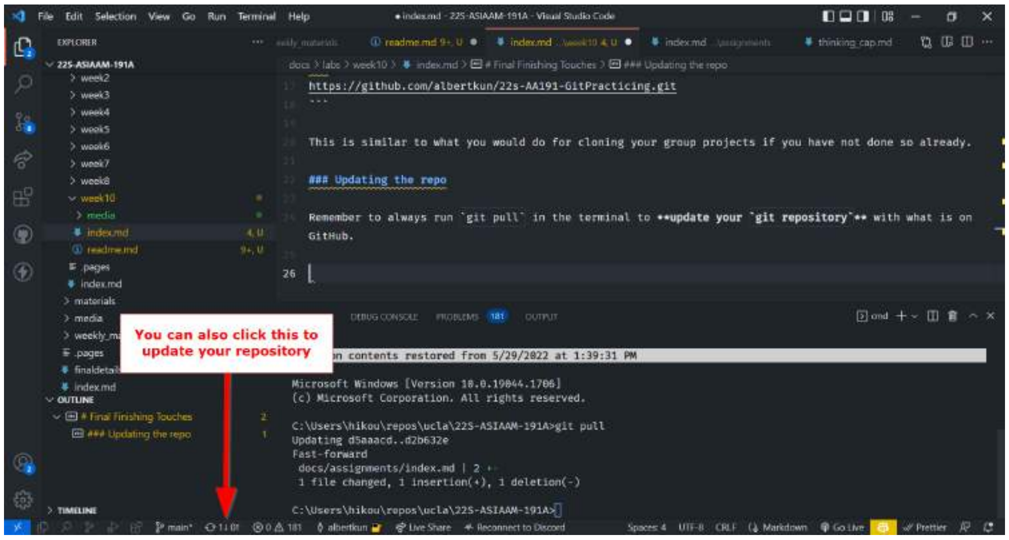
- Your public profile information
- Certain activity within this repository
- Country of request origin

Updating the repo

Remember to always run `git pull` in the terminal to **update your `git repository`** with what is on GitHub.

```
git pull
```

If you don’t want to use the terminal, you can also click the following button on VS Code to `push` **AND** pull updates:



```
https://github.com/albertkun/22s-AA191-GitPracticing.git
```

```
This is similar to what you would do for cloning your group projects if you have not done so already.

### Updating the repo

Remember to always run `git pull` in the terminal to **update your `git repository`** with what is on GitHub.
```

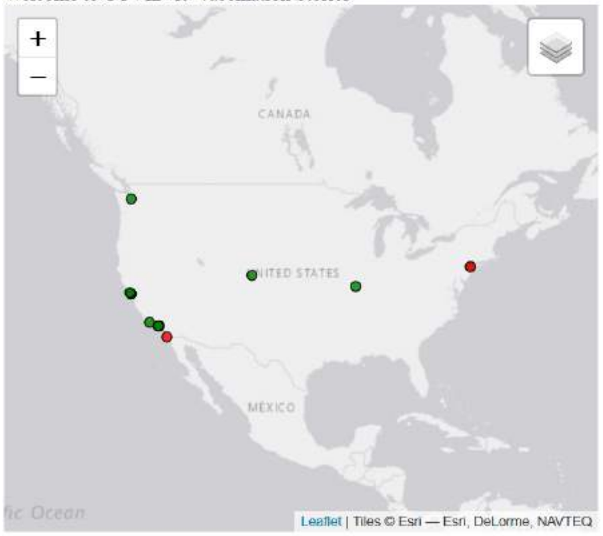
```
n contents restored from 5/29/2022 at 1:39:31 PM

Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hikou\repos\ucla\22S-ASIAAM-191A>git pull
Updating d5aaacd..d2b632e
Fast-forward
 docs/assignments/index.md | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\hikou\repos\ucla\22S-ASIAAM-191A>
```

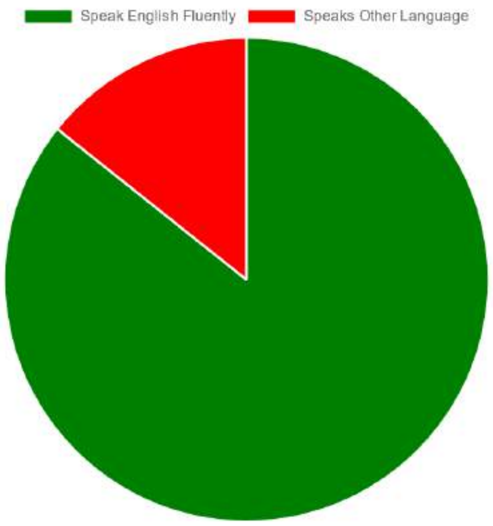
Update branches

If you have other branches on the repo, you can update branches by typing the following command in the terminal:

```
git fetch --all
```

Welcome to COVID-19 Vaccination Stories

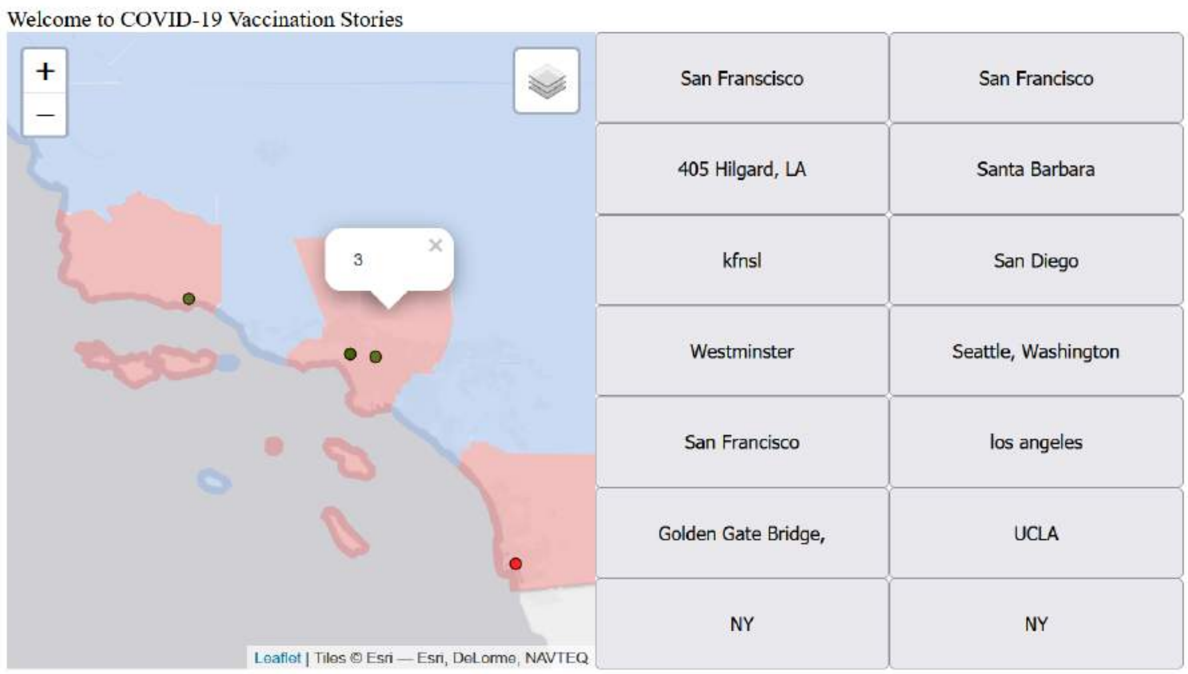




Now that we have a better sense of how to use CSS grid, we can think about how to fit other libraries and tools.

Evaluating the right libraries and tools for the task is an important part of being a web developer that is both ethically minded and able to contribute back to meaningful projects.

Turf.js



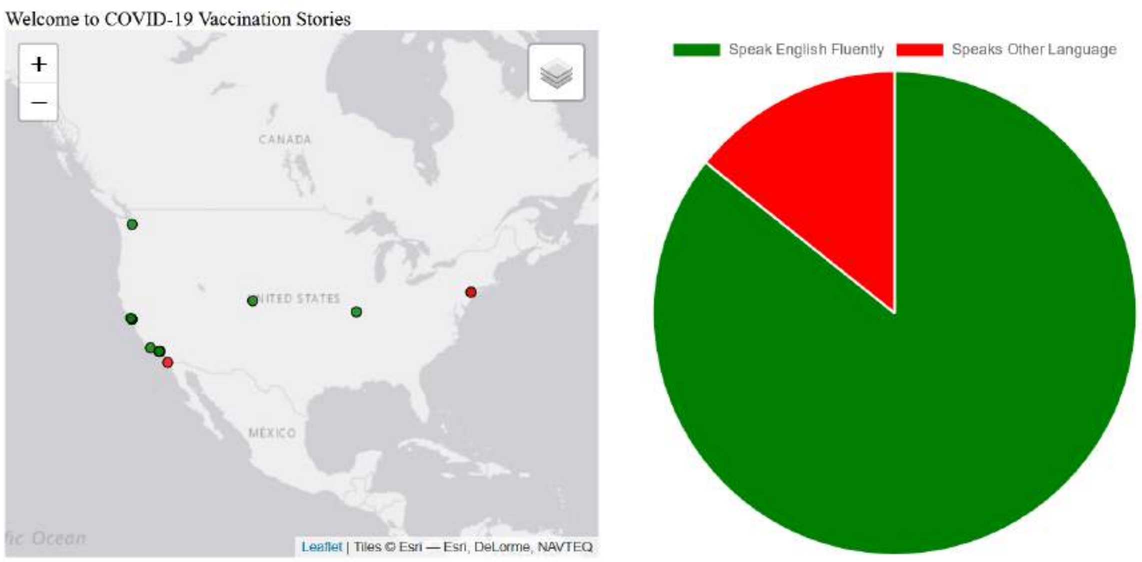
https://turfjs.org/

Turfjs is useful for running spatial analysis in our mapplications.

Here is my example repo using Turf.js to count the number of points inside a particular boundary:

https://github.com/albertkun/leaflet-tufjs-spatial-join

Chart.js

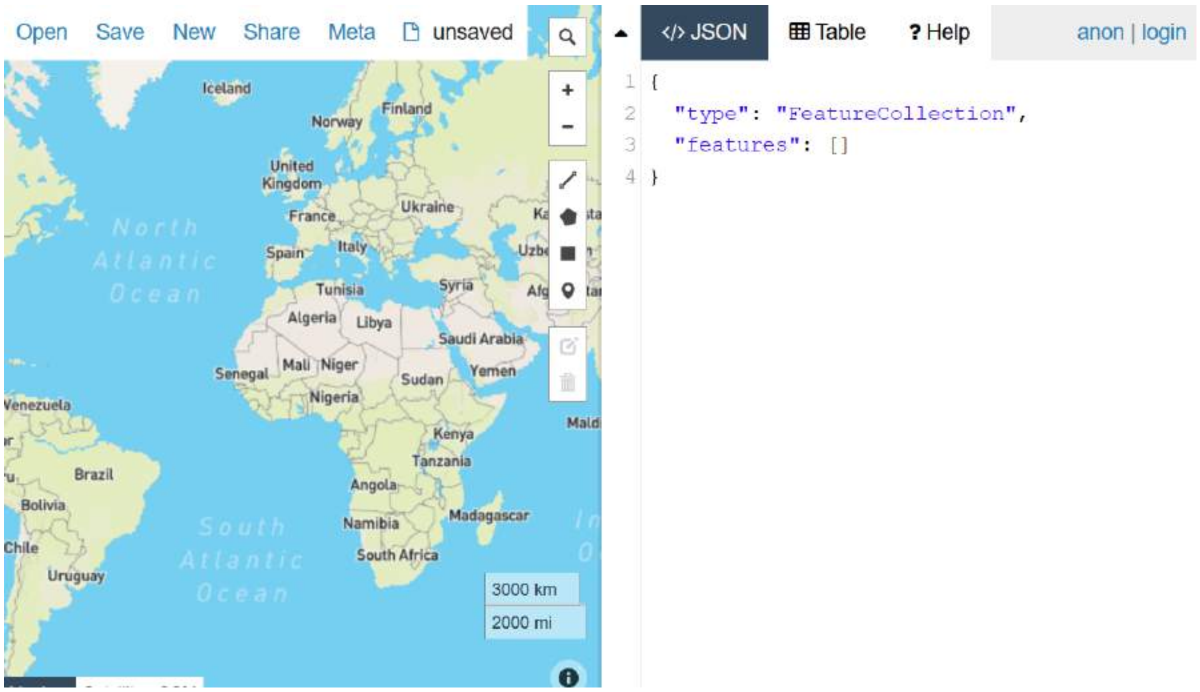

Welcome to COVID-19 Vaccination Stories
CANADA
UNITED STATES
MEXICO
Leaflet | Tiles © Esri — Esri, DeLorme, NAVTEQ
Speak English Fluently
Speaks Other Language

https://www.chartjs.org/

Chartjs is useful (but complex) library for creating charts in our mapplications.

Below is an example repository demonstrating how to use Chartjs with your Leaflet data: https://github.com/albertkun/leaflet-chartjs

GeoJSON.io

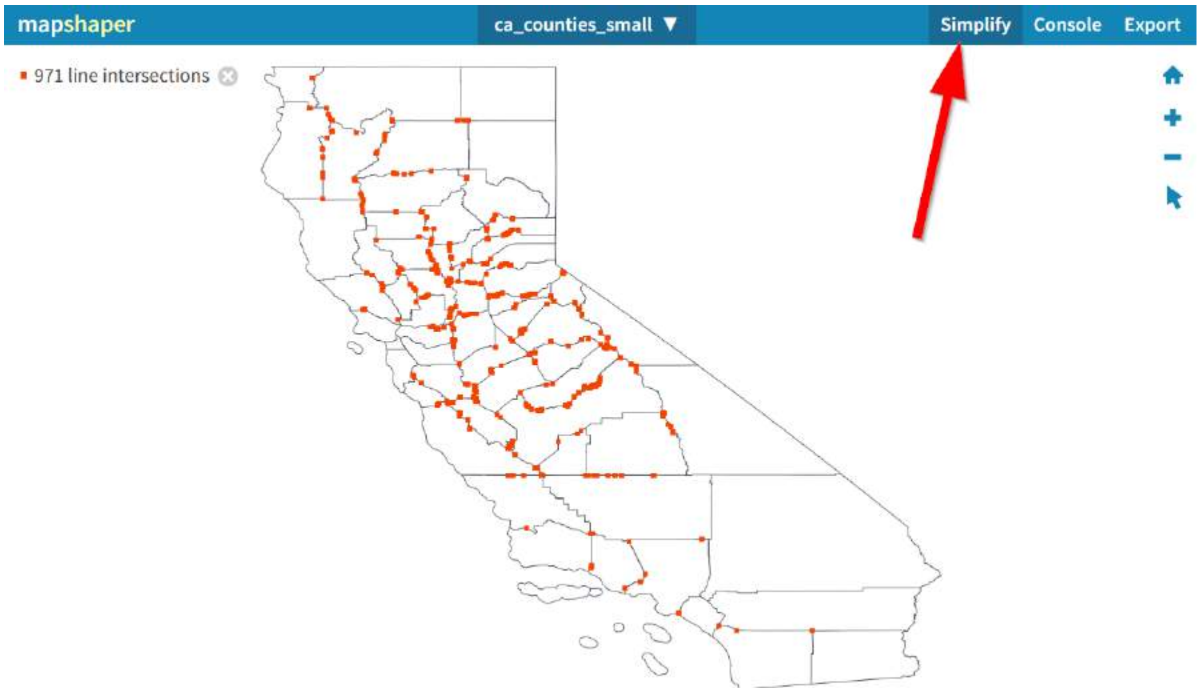

Open Save New Share Meta unsaved
JSON Table ? Help anon | login
3000 km
2000 mi

```
{
  "type": "FeatureCollection",
  "features": []
}
```

http://geojson.io/

Remember this tool? GeoJSON.io is useful for creating, converting, or quickly editing spatial data online.

MapShaper

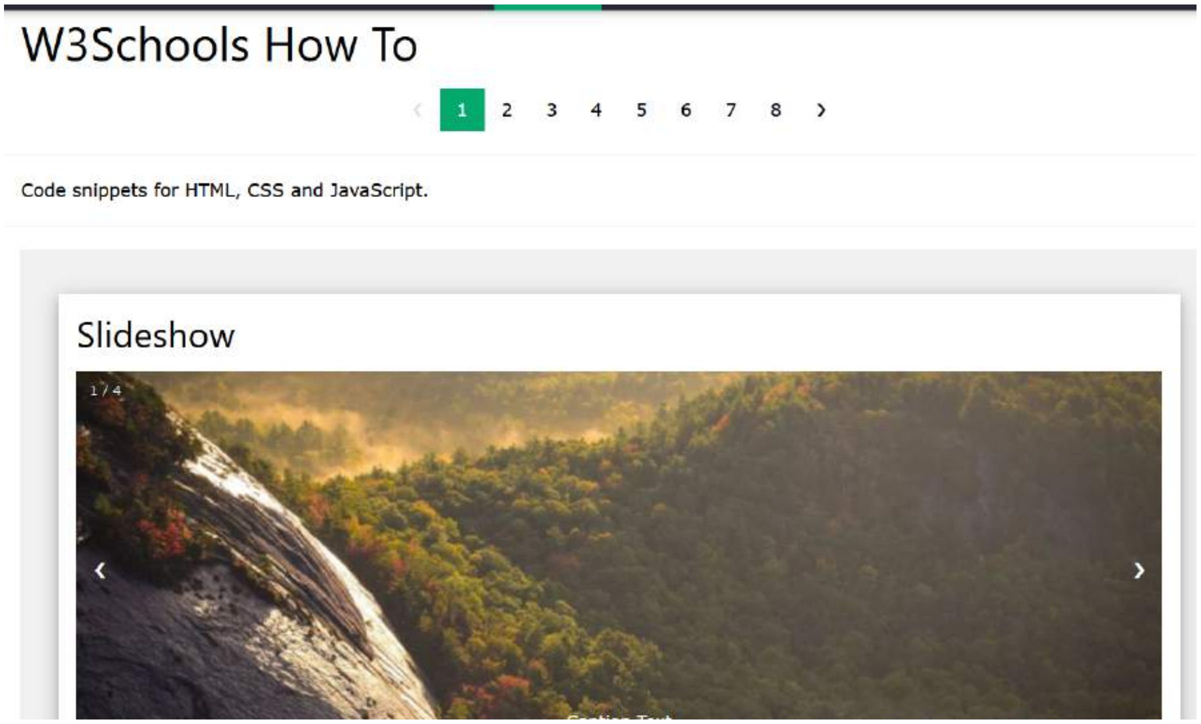


https://mapshaper.org/

This online tool helps to reduce file sizes of GeoJSONs and do other geoprocessing of GIS data:

> Main Tip: Choose the right tool for the job!

W3 Schools for Building Content from scratch



https://www.w3schools.com/howto/default.asp

Similar to Mozilla Developer Network for documentation, W3 schools provides a lot of nice how-to tutorials on how to implement common website features, interfaces, and functions.

Extending Leaflet with Plugins

Owing to the opensource nature of Leaflet, people have written many reusable tools to help with common tasks. As a matter of fact, if you have written a function for Leaflet that you think is reusable, you can go ahead and submit to their list of plugins too! Ah, the awesomeness of open source!

Remember! Balance Open Source with an open and ethical mind

Of course, open source has its drawbacks too, learning to customize someone elses poorly written code with poor documentation can be a huge timesink.

Balancing the trade-off between trying to reuse someone’s code and creating your own is an important step in becoming a seasoned developer!

Working within and with other projects bring us to our final lab topic of utilizing other people’s Leaflet widgets and plugins.

Last update: 2023-06-01

Final Lab Exercise

Note

This is a group assignment. Only one person per group needs to do this.

Join up with your group. With the remaining time, look at some of the plugins below and try to implement them into your mapplication or your own group projects:

Due 6/8 (if not done in class)

Instructions

1. Go to the Git Practicing repo and do a `git pull` (if you have not cloned it already, then clone it). If you run into errors on the branch, you may need to run `git pull --rebase`
2. Try out one of the following Leaflet plugins from the list below OR explore one from this list and add to the end of the table: https://leafletjs.com/plugins.html
3. Find **your group** in the table, put the **tool** name, and **comments** about the tool, Plugin Review section below.
4. Optional: If you were able to get the example up and running, put your GitHub pages example under the "**Example Implementation**".
5. Make a **Pull Request** to this Git Practing Repo and contribute your changes!

Plugin List

Plugin Name	Link
UI	
Sidebar v2	https://github.com/noerw/leaflet-sidebar-v2
Sidebar v2	https://github.com/turbo87/sidebar-v2/

Plugin Name	**Link**
Leaflet Control Window	https://github.com/mapshakers/leaflet-control-window
Leaflet Sleep	https://cliffcloud.github.io/Leaflet.Sleep/
Markers	
Beautify Marker	https://github.com/masajid390/BeautifyMarker
Icon Pulse	https://github.com/mapshakers/leaflet-icon-pulse
Parallax Marker	https://dagjomar.github.io/Leaflet.ParallaxMarker/
Leaflet Swoopy	https://wbkd.github.io/leaflet-swoopy/
Others	
Leaflet Hex Timeslider	https://github.com/albertkun/leaflet_hex_timeslider

Leave a review of **one** of them in the Git-Practicing Repo:

- https://github.com/albertkun/23S-ASIAAM-191A-Git-Practicing/blob/main/review.md

Feel free to try others not in this list and add it to the doc.

Pay attention to how important good documentation is and how your own group projects `readme.md` should be structured. Creating a `branch` will be helpful when testing new features. Refer to lab 8 for a refresher on branches.

Submission

As a group, have **one person** make a pull request in the following repo with your comments on a plugin:

- https://github.com/albertkun/23S-ASIAAM-191A-Git-Practicing/blob/main/review.md

Last update: 2023-06-01

Group Peer Review Exercise

1. Make a copy of this Google Doc by clicking on the link below: https://docs.google.com/document/d/1pMH7JQmiZyUL11znFMluZqcl4e9bUFNZ8mOKg76vZgI/copy#
2. Go to the websites for your pair group and fill out Part 1 (5 minutes)
3. Take notes during the presentations and fill out Part 2
4. Fill out Part 3 while discussing with your group mates (3 minutes)
5. Group A discuss Part 3 with Group B (5 minutes)
6. Group B discuss Part 3 with Group A (5 minutes)

Last update: 2023-06-01