Geography 572
Lab #3: Advanced CartoCSS Styling

Lab Objectives:

- Create an aesthetic tileset based off of the pastiche lecture material
- Serve your own set of tiles using Mapbox Studio and CartoCSS
- Continue to build your knowledge of Illustrator, Photoshop, and HTML5/CSS

Evaluation:

This lab is worth **40 points** toward the Lab Assignments evaluation item, which is worth 25% of your overall course grade. A grading rubric is provided at the end of the lab to inform your work.

Schedule of Deliverables:

- **October 30th:** Lab #3 Assigned //client contract begins
- **November 6th:** Inspiration Board Due (2pts) //design & feedback from client
- **November 13th:** POIs list Due (6pts) //input & feedback from client
- **November 20th:** Lab #3 Due (32pts) //contract deadline

Challenge Description

The director of a local art museum wants to incorporate artistic principles into a tourism slippy map they'll display on their website. They want to promote the City's natural, cultural, historical, and/or economic points of interest, or POIs(i.e., places that tourists may wish to visit during their stay). Since the museum is renowned for its forward thinking, the map should incorporate an aesthetic time period or movement in cartography. The Director has made clear that a trained cartographer must be hired to design a map tile set that is innovative and aesthetically pleasing.

CartoCSS is a styling language specific to mapping that uses existing CSS principles and enhances them to be used for designing vector or raster tile sets online. Professional cartographers have found a way of styling web tiles via the CartoCSS framework to push the limits of what people expect from slippy maps. These maps are highly designed and are innovating the possibilities of the field.

Notes from the Director of the Chazen

Your inspiration board serves as a deliverable, to show that you understand how to interpret advanced map stylings that fit with a particular aesthetic look. Your tileset will be built to incorporate all zoom levels, from a view of the world (zoom 1) to street level view (zoom 15) and must be designed for all appropriate scenarios.

The tilesets will be embedded in a webpage, with basic HTML/CSS and Javascript provided to guide you with basic slippy map interaction. Your tilesets must include **at least** 30 POIs icons on the map (i.e., point markers with a basic description). POIs can include a variety of

locations, like restaurants, parks, etc. that might be useful in this scenario. You may choose any city for your scenario, as long as you follow the basic requirements for the POIs. It is important to keep in mind that the contract includes ***both*** the design of the tileset at every zoom level AND the insertion of this map with markers denoting POIs. Finally, you must make use of Mapbox Studio for Lab #3, an open source product for serving custom vector tiles provided by the company MapBox that is introduced in the lab.

1. Creating an Inspiration Board

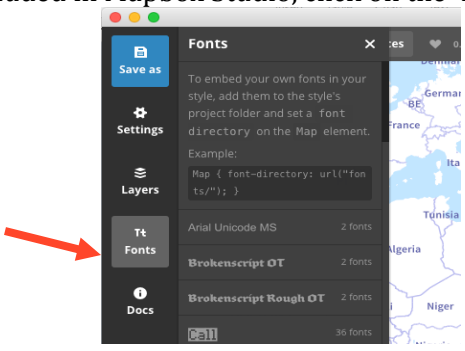
The first step in Lab #3 is to demonstrate the inspiration for your tileset, based off of two scenarios. Scenario One is incorporating one of the aesthetic styles of mapping as described in Lecture X, and providing appropriate examples from maps and art that reference specific details you plan on including in the map that you find appropriate for the map style. Scenario Two is using a specific map or media (such as a painting or video game) to determine the tileset style. Scenario One will be the example for this lab, creating a 'Pop Art' tileset based off of the work of Roy Lichtenstein, a famous 1960's artist who is famous for his comic-like paintings of people. Pop art maps are having a resurgence in the cartographic community, after a Cartographic Perspectives Journal article discussed their use for artistic expression in mapping.

Some visual variables to consider for your inspiration board include:

1. **Color:** What colors are you going to use? Make sure they are either relevant to the aesthetic style you are mimicking or included in your sample images.
2. **Texture:** Are there specific textures common to the style?
3. **Typefaces:** Mapbox Studio provides over 300 fonts, but you can (and definitely should) included your own to be used, which typeface best fit your map?
4. **Stylistic effects:** Are waterlines or certain effects like drop shadow common? Should your design be minimalistic? Should your line weight be heavy or light? Colors dark or transparent?

Your inspiration board should have sufficient images to describe what you will attempt to accomplish in your tileset and your artistic/cartographic vision. You should have at least 5 various specific pieces of inspiration for your tileset described in short sentences to guide you through the tileset design process. The sky is the limit, and if you have questions- look at the example gallery or ask your TA/Professor if you have questions. You are required to have a PDF of the inspiration board assembled and ready for mapping by **November 6th**.

To examine different fonts included in Mapbox Studio, click on the 'Fonts' button on the sidebar:



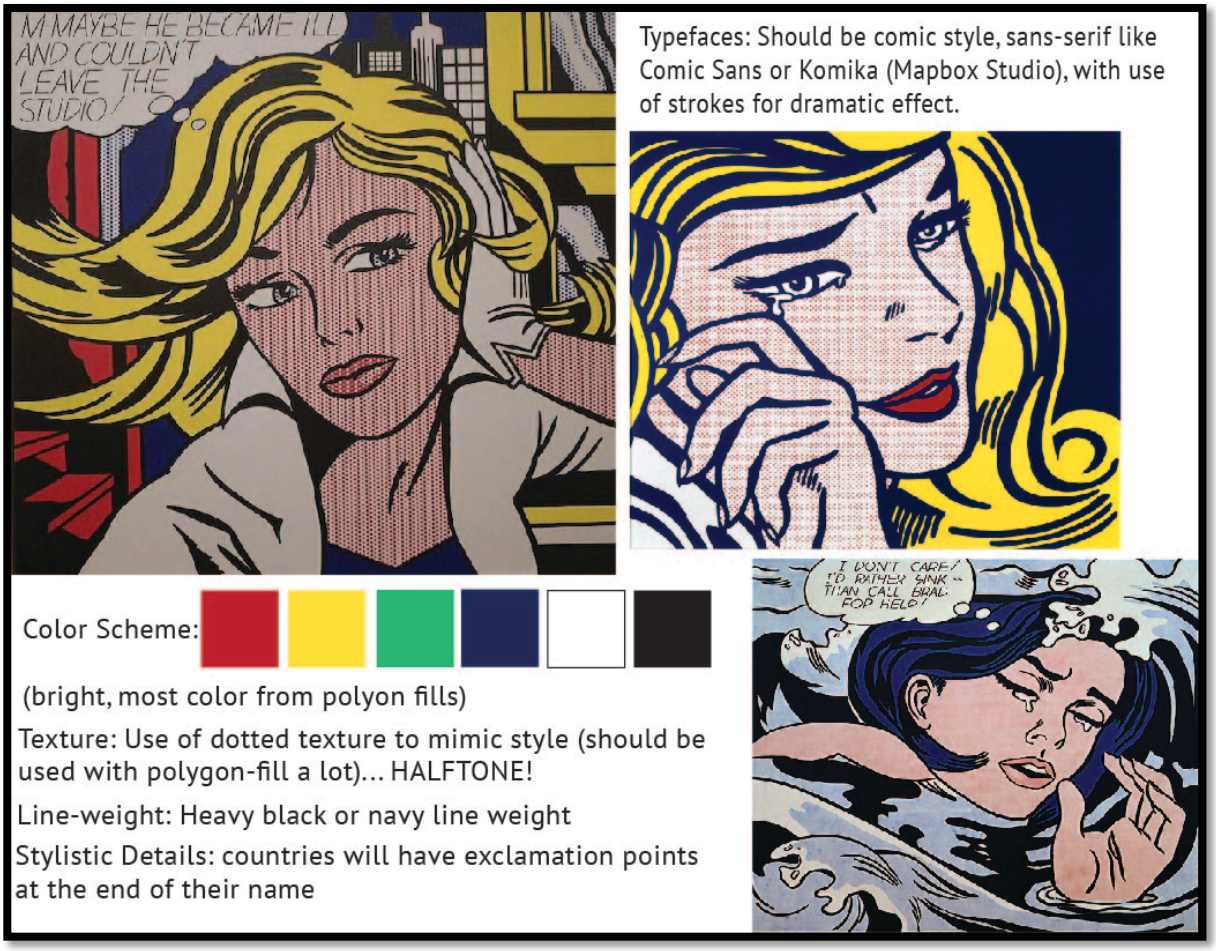


Figure 1: An example inspiration board, based off a Roy Lichtenstein painting.

2. Assembling Your POIs

The next step in Lab #3 is assembling some data for your places of interest to be added to your map. The information set for Lab #3 emphasizes features at the point dimensionality. While you can include linear or areal features in your map (e.g., a scenic drive, a park), you will need to represent them as points for Lab #3; while this may seem like a limitation, the conceptualization of features as points is common on web maps due to the provision of integrated map designs at multiple scales, the smallest of which require collapsing to points.

It is up to you to decide which features of interest will be included in your map; if you choose a large city, consider giving your map a specific theme rather than including all possible POIs that may be found on a tourism map. Regardless of which features you ultimately decide to map, you will need to collect five pieces of information about each feature:

1. **Name:** The specific name of the instance of the feature. This is different from the category, as several mapped features may receive the same iconic point symbol.

2. **Latitude:** In decimal degrees, using negative numbers for latitudes in the Southern Hemisphere. There are many places to get the coordinates of places, whether through Google Maps or a free geo-referencing service.
3. **Longitude:** In decimal degrees, using negative numbers for longitudes in the Western Hemisphere.
4. **Description:** A brief (100 words or less) description of the feature, perhaps with a link to additional information. You should compose the descriptions yourself such that the set of descriptions are at a common length and contain a common set of summary information. Both the name and description columns will be used to populate an information window upon clicking the icon.

Your information must be stored in a *comma separated file*, or ***.csv***, for loading into Mapbox Studio. As its name implies, a *.csv* file is a flat text file that delimits rows using the comma (",") character. To create your *.csv* file, you can assemble your spreadsheet in Microsoft Excel (as shown in **Table 1**) and export using *Save As->Save As Type: CSV.* You are required to have your *.csv* file assembled and ready for mapping by **November 13th**.

	location	lat	long	painting
2	Albright-Knox Art Gallery	42.932976	-78.875618	Head: Red and Yellow, Picture and Pitcher
3	Art Institute of Chicago	41.8795	-87.624089	Alka Seltzer
4	Arthur M. Sackler Gallery	38.88765	-77.026295	Landscape in Scroll
5	Chrysler Museum	36.85611	-76.293546	Live Ammo (Ha! Ha! Ha!)
6	Cranbrook Art Museum	42.57273	-83.24947	Modular Painting with Four Panels, #7
7	Denver Art Museum	39.73685	-104.98946	The Violin
8	Des Moines Art Center	41.58393	-93.681963	Perforated Seascape #1, The Great Pyramids
9	Detroit Institute of Art	42.35911	-83.065223	Interior With Mirrored Closet, Modern Sculpture with Three Voids, Modern Sculpture (Maquette)
10	Fogg Art Museum, Harvard University	42.37402	-71.113725	Reclining Nude
11	Frederick R. Weisman Art Museum	44.97332	-93.236969	World's Fair Mural
12	Hirshhorn Museum and Sculpture Garden	38.88785	-77.022945	Modern Painting with Clef, Modern Painting with Sun Rays, Modern Sculpture with Black Shaft
13	Lowe Art Museum, University of Miami	25.7191	-80.275899	Modular Painting with Four Panels, #5
14	Milwaukee Art Museum	43.039865	-87.897594	Crying Girl
15	Musée d'Art Contemporain de Montréal	45.50769	-73.567092	Brushstroke Mural for the University of Dusseldorf
16	Museum of Contemporary Art (LA)	34.05339	-118.25071	Many
17	Museum of Contemporary Art (San Diego)	32.84442	-117.27772	Mirror Six Panels #3
18	Museum of Modern Art	40.76123	-73.977745	Many
19	Modern Art Museum of Fort Worth	32.74878	-97.36321	Atom Burst, Mr. Bellamy
20	Museum of Fine Arts	42.33889	-71.093812	Seascape
21	Nasher Sculpture Center	32.78794	-96.800355	Double Glass, Head with Blue Shadow, Peace Through Chemistry
22	National Gallery of Art	38.89177	-77.019954	Many
23	Nelson-Atkins Museum	39.04536	-94.580789	Still Life in Yellow and Black
24	Norton Simon Museum	34.1458	-118.15931	Long Modern Sculpture, Modern Painting for Expo '67
25	Parrish Art Museum	40.90523	-72.364566	Collage for Apple, Drawing for Leda and the Swan, Drawing for Peace through Chemistry
26	Philadelphia Museum of Art	39.96621	-75.181717	Still Life with Goldfish
27	Rhode Island School of Design Museum	41.82686	-71.40731	Pyramids II
28	Rose Art Museum, Brandeis University	42.3655	-71.262058	Forget It! Forget Me!
29	Saint Louis Art Museum	38.63955	-90.294083	Curtains, Goldfish Bowl II, Him, Sailboats
30	San Francisco Museum of Modern Art	37.78562	-122.40116	Many
31	Seattle Art Museum	47.60718	-122.33841	Study for Vicki!
32	Solomon R. Guggenheim Museum	40.78279	-73.959143	Many
33	Spencer Museum of Art University of Kansas	38.95963	-95.244903	No- Nox (drawing)
34	University Art Museum: California State Long Beach	33.7834	-118.11488	Study for Purist Still Life
35	Virginia Museum of Fine Arts	37.55572	-77.474145	Gullscape, Still Life with Folded Sheets
36	Whitney Museum of American Art	40.73942	-74.008884	Goldfish Bowl, Little Big Painting, Modern Sculpture with Velvet Rope, Still Life with Crystal Bowl
37	Yale University Art Gallery	41.30822	-72.930798	Many
38	Albertina (Batliner Collection)	48.20463	16.367796	Face (Green Nose), Reflections on Hair, Glass and Lemon in a Mirror
39	Basil & Elise Goulandris Foundation	37.83867	24.938869	Nude with White Flower, Sunrise
40	The Berardo Collection	38.69536	-9.209594	Mirror #4, Interior with Restful Paintings

Table 1 shows example POI information for where Lichtenstein Maps are worldwide

3. Intro to Mapbox Studio

Cartography has forever been altered with the influx of mapping on the web, using interactivity to animate and create more functional maps than ever before. Popularized by Google Maps and Mapquest for their navigation-oriented basemaps, the term ***slippy map*** describes a mashup that draws upon web mapping services to provide integrated, multiscale map designs, typically using a tile-based raster system.

A slippy map also exhibits basic interactive functionality, such as, panning, zooming (together where the 'slippy' term came from), overlay of different basemaps, and retrieval of details through information windows (e.g., your point icons). The advantage of slippy

map mashups is that their creation requires very little custom code by combining existing services.
While originally used as reference maps for Google Maps or Mapquest, slippy maps have undergone a revolution with CartoCSS. As original slippy maps were considered poorly designed and under-developed, the cartographic world has worked to alter the stigma around slippy maps. Companies like Mapbox incorporate CartoCSS into their business model, creating a platform to make custom map tiles that are best for what a cartographer is creating for a client. These range from basic transportation maps, to minimalist designs, to the highly designed (the latter of which is the concentration of this lab).

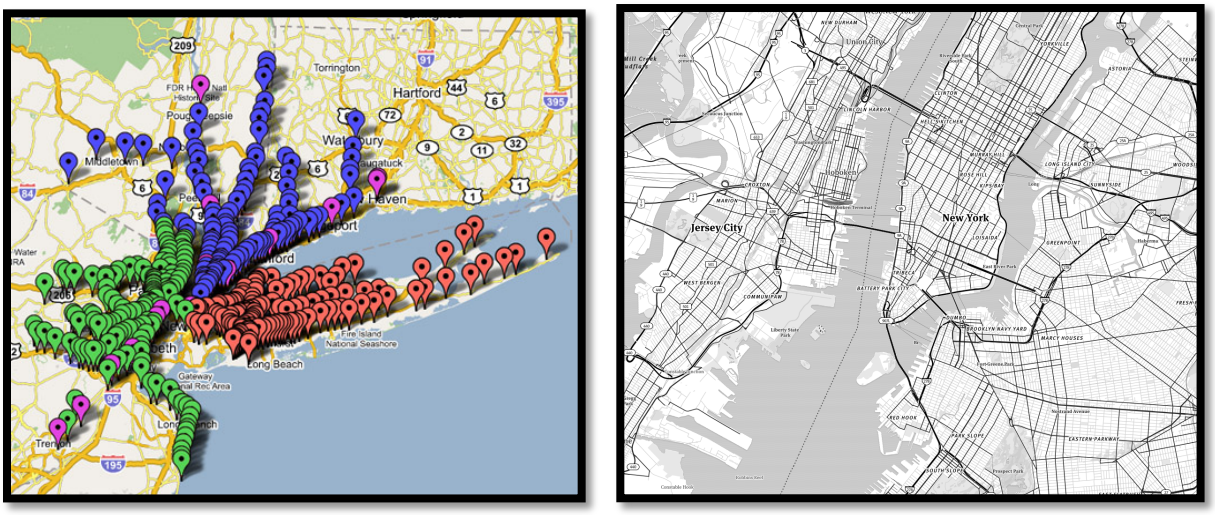


Figure 2 & 3: The transformation of tilesets, from the original Google Maps 'Pushpin' raster tileset to the highly designed Mapbox Studio vector tileset

a. Getting Started with Mapbox Studio:

Despite initial cartographic frustrations with the limitations in designing map tiles, the ability to customize the cartographic display has improved substantially in recent years through expansion and refinement of tools for styling and rendering your own basemap tiles.

Your Lab #3 slippy map mashup will make use of Mapbox Studio, an application developed by a company called Mapbox that supports custom styling and rendering of vector basemap tiles. MapBox is quickly becoming a leader in the geospatial industry, particularly in the area of web mapping. Mapbox Studio allows you load and style your own datasets, such as your set of POIs, or to draw from existing data services, such as OpenStreetMap or Natural Earth.
It is important to note that Mapbox Studio can be used for free, using their educational pricing that allows you a basic plan for free with a .edu email address: https://www.mapbox.com/education/. If you exceed the limit, which for the purposes of Lab #3 challenge you definitely will not, you would need to pay. Finally, we will be serving raster tiles for Lab #3, given the commonality of this solution. MapBox has been highly

innovative in the development of vector-based tiling, a promising solution for responsive web cartography in the future.

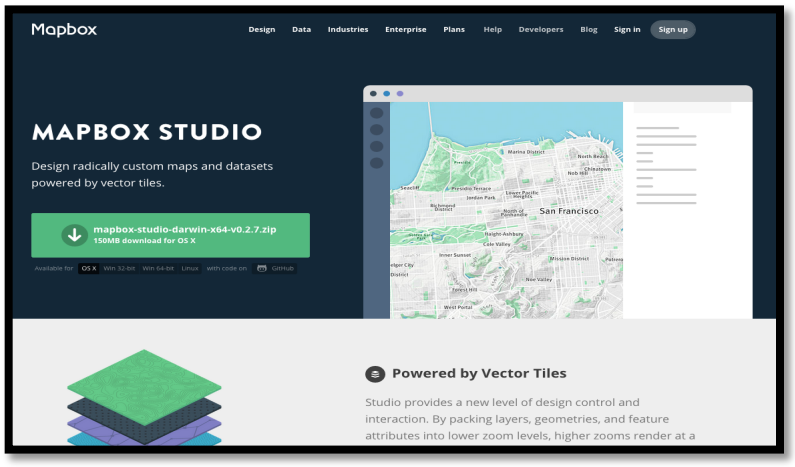


Figure 4: The Mapbox Website for Mapbox Studio

Get started with Mapbox Studio after registering for the Education account (you should be all set a minute or two after sending, but double check your SPAM folder) and go to the Mapbox website for Mapbox Studio: https://www.mapbox.com/mapbox-studio/ (**Figure 4**). If you prefer to work on your own machine, you then can install Mapbox Studio by selecting the appropriate operating system from the download options (Installing is extremely easy and highly recommended for your personal machine). Studio is also available on all machines in Science Hall 380 and M376. If you are working on a Science Hall machine, open Mapbox Studio to begin (**Figure 5**). It's important that you create an account and login; you can save everything to Mapbox's cloud and can be accessed on any computer.

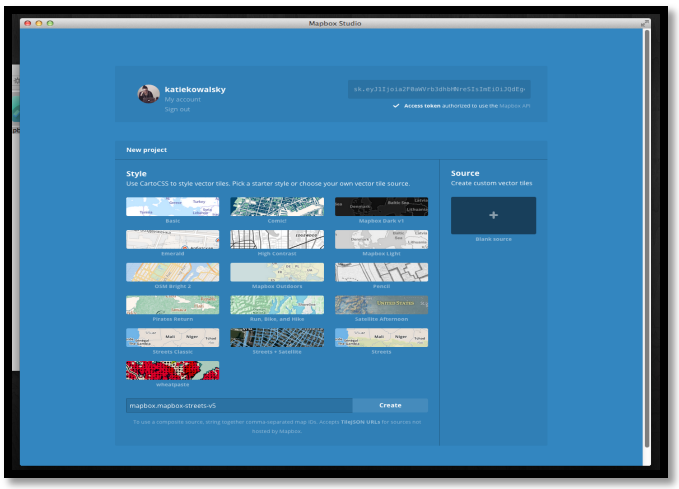


Figure 5: The Mapbox Studio Application.

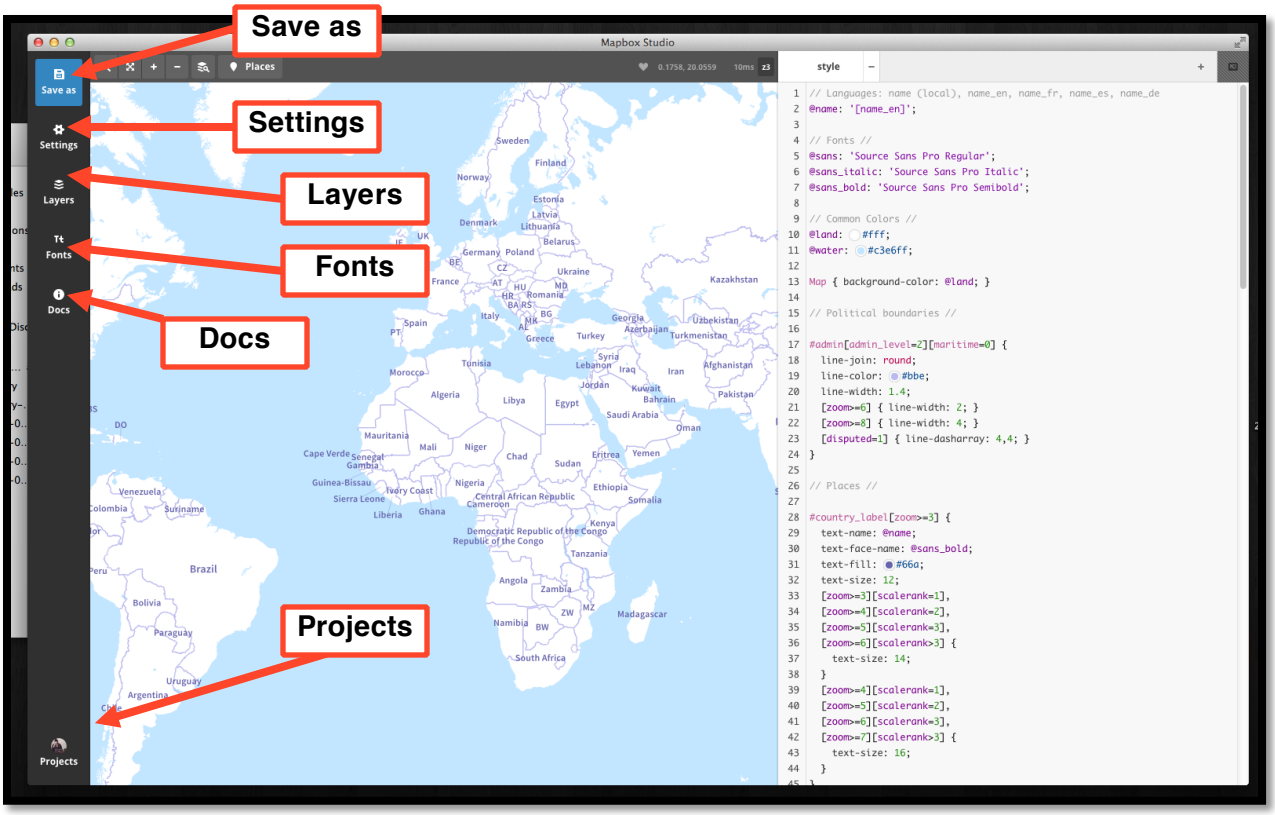


Figure 6: The Mapbox Studio Interface

Once open, click on a sample template to get a feel of where things are in Mapbox Studio. Under the 'Docs' tab, there is an **Interface Tour that is highly recommended** to get a sense of where things are in the application.

The application includes five tabs along the left bar (**Figure 6**):

- The ***Save As*** allows you to constantly save any changes made to your tileset (make sure you choose either the Desktop or your flash drive if working in 380).
- The ***Settings*** tab allows you to upload your style to Mapbox when finished to be viewed/hosted on their website, to name and describe it, and various settings like altering the center position, changing the min/max zoom and so on.
- The ***Layer*** tab is extremely important for CartoCSS, allowing you to add/change sources and sees the layer order for your source. You can also click on each source layer to see the data properties (that can be used as selectors in CartoCSS)
- The ***Fonts*** tab shows the variety of typefaces available and built into Mapbox Studio. Clicking on each will allow you to see each and every font possible. You can also see the documentation for adding your own fonts as well.
- The ***Docs*** tab gives you access to all of the acceptable CartoCSS documentation and has the interface tour, which is highly recommended that you use.
- The ***Projects*** tab is where you can create a new source or style project, log in and out and browse existing projects you might have. *Be sure to logout of your MapBox account at the end of your work session*.

b. Style vs. Source in Mapbox Studio

When you open Mapbox Studio, you'll find two options: Style or Source. These two options are kept separate in Studio, a division not implemented in Mapbox's earlier product: Tilemill. As you can see, the various style examples that come built in Studio (**Figure 10**) are on the Style side. That part of the app is where we'll do all of our CartoCSS styling.

So what's Source for? Mapbox has its tiles, Mapbox-Streets that is going to be used in most cases for this lab. However, say you need to want to use bathymetry data from Natural Earth and Mapbox doesn't have it included in their own vector tile source files. You can easily import any geographic file (Mapbox supports importing these data types: Shapefiles, GeoJSON, SQLite/PostGIS Databases, GeoTIFFS, VRTs, KML, GPX, and CSV) and upload it to be used in your tileset. However, Mapbox provides three different in-house datasets you'll probably use: Mapbox-Streets, Mapbox-Terrain, and Mapbox-Satellite. You can also combine multiple data layers under the Layer Tab by pressing 'Change Source' and stringing the names together with commas.

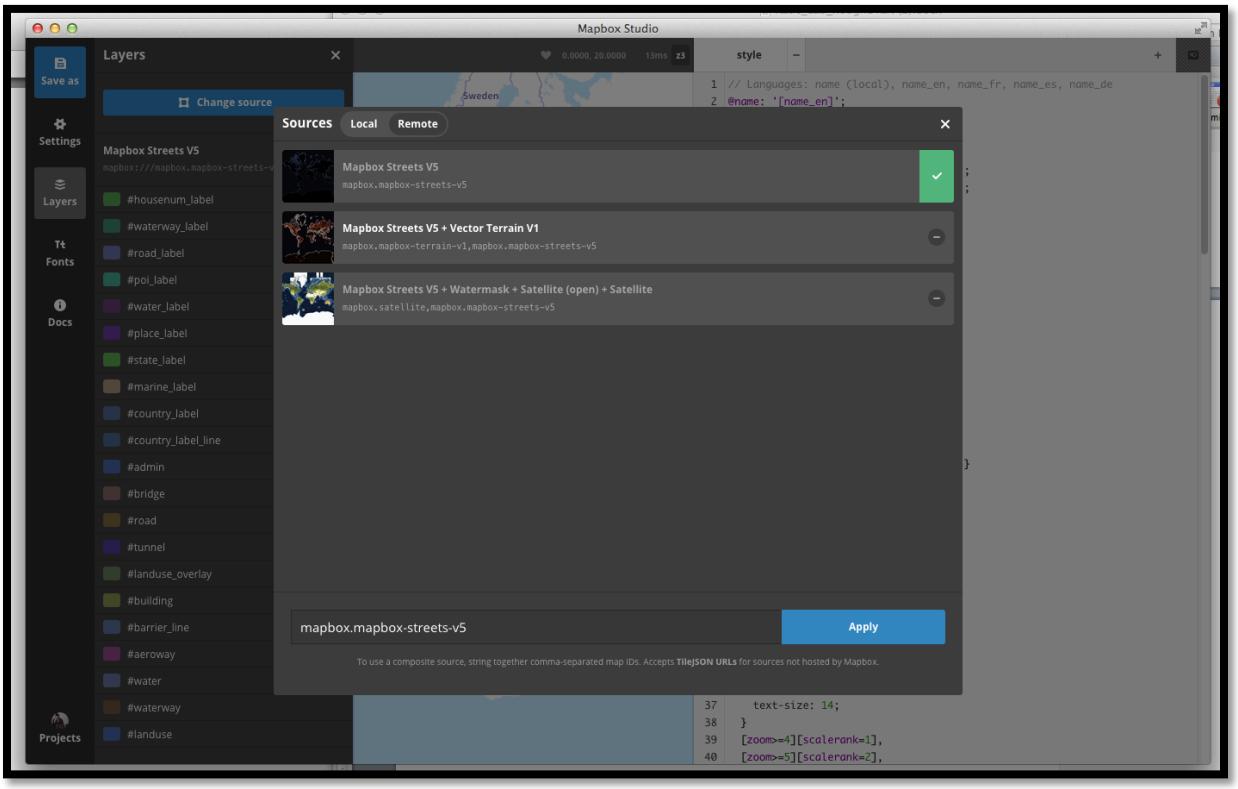


Figure 7: Mapbox Remote Data Sources

Why is this important? The layers tab shows what layers you have available, and what attributes you can use for selectors in CartoCSS. Sources control what you have available. Another way to view what the sources look like under the surface of your tileset is by pressing the layers view next to zoom in/out and the Places marker on the toolbar for Studio (**Figure 8**)

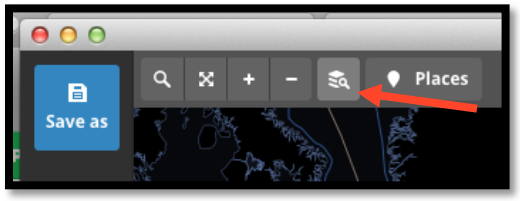


Figure 8: Layer View

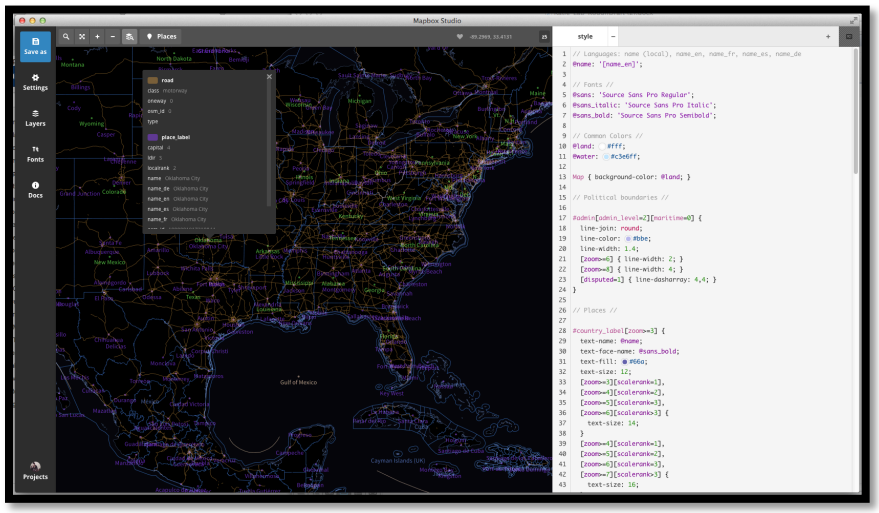


Figure 9: Layer View with attributes shown

The layer view allows you to see all possible vector tile layers. Clicking on an area (**Figure 9**) allows you to see all the layers 'on' that certain area and what attributes will be shown. This feature is a bit clumsy, but necessary for understanding certain features of Mapbox-Streets, Terrain, and Satellite. Using this can help you understand what values you'll use in selecting statements for CartoCSS.

c. Styling with CartoCSS

Create a new tileset in Mapbox Studio (do this by pressing 'New Project' and scrolling to the bottom where you'll see **Figure 10** and press 'create'), and then immediately save it. You'll notice that if you open the .tm2 file, there are a few different things in there. Any file with an .mss is a style sheet; you always start out with one called style. The next two files are an XML and YML. These files control your layer order, the spatial reference system being used, and all the various parameters involved in your tileset.

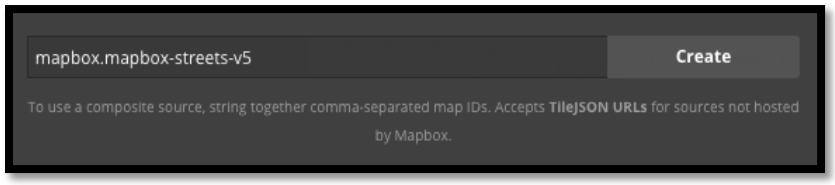


Figure 10: Create a blank tileset on the New Project Page

The .tm2 folder is all of your tileset in one package. You should house anything additional you might want to include in there. If you're adding fonts, textures, or images, keep them

inside the folder to be easily called in your stylesheets. The .tm2 works similar to shapefiles in the way that all of the different parts (the XML, YML, and .MSS files) are crucial and can't be deleted if you want your tileset to work. Unlike Tilemill, Studio's predecessor, it's simple to move around and store your tileset in the .tm2 folder on flashdrive storage or on the cloud with services like Box, Dropbox, or Github.

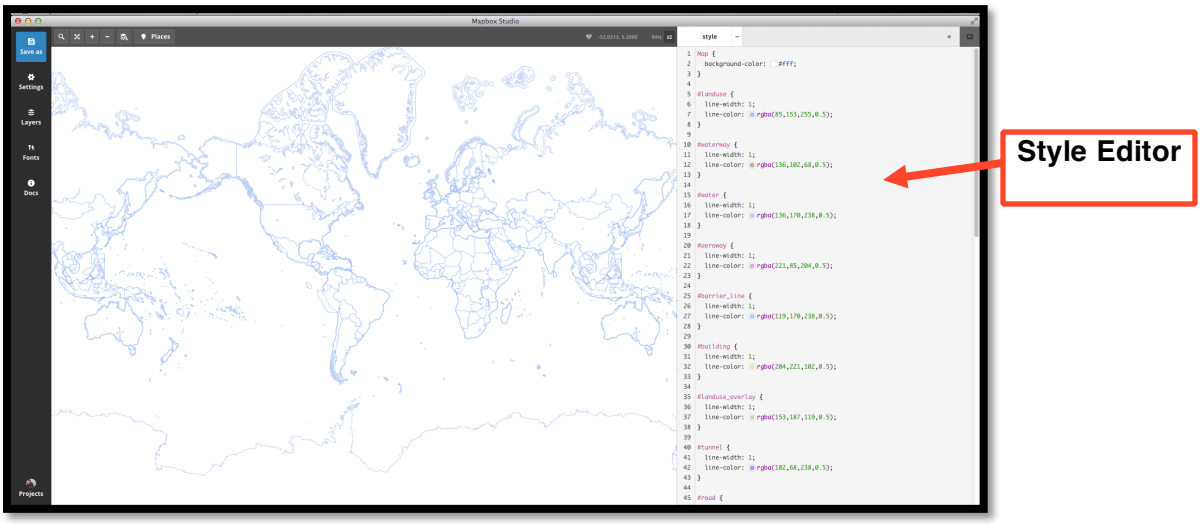


Figure 11: Mapbox Studio Style Editor

The Style Editor is the heart of Mapbox Studio. As you can see, there's already some CartoCSS styling put into the style sheet, which is why you can see all of the various country names, outlines, water, etc. You can also examine everything at various zoom levels by using the zoom in/out buttons or the Places button. The Places view (**Fig 12**) allows you to see various areas that are 'tagged' by Mapbox Studio to see how your styles affect certain areas.

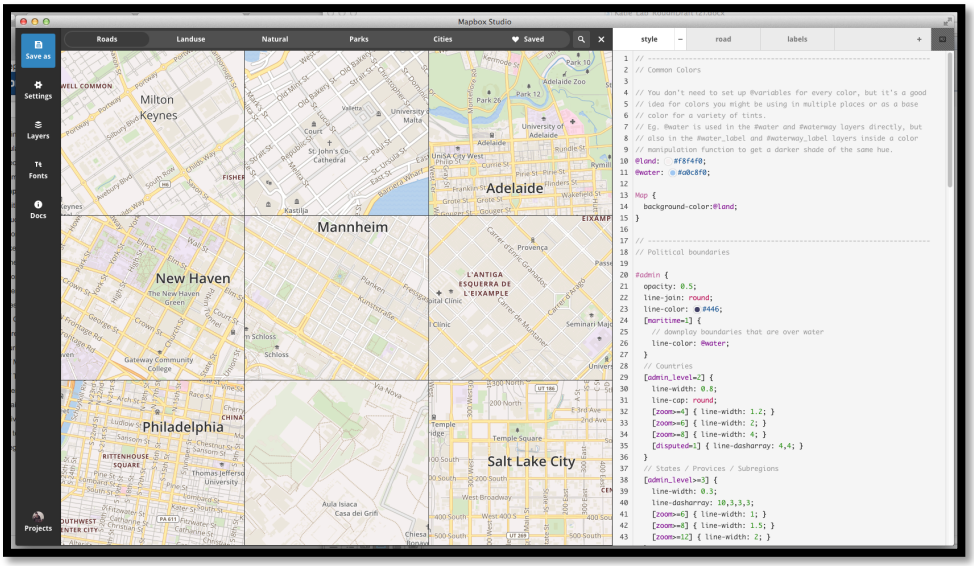


Figure 12: Places View in Mapbox Studio

Mapbox Studio supports the styling of basemap tiles through their ***CartoCSS*** styling specification. In Lab #1, you learned about CSS and the components of style rules (e.g., selectors, properties, values, ids, classes, etc.); return to the second part of Lab #1 if you are not yet comfortable with these concepts. The CartoCSS specification leverages the CSS syntax to allow you to style map features, rather than page elements. Hopefully you now see the value of Mapbox Studio: it makes map design much more like web design. The skills learned to build your web portfolio in Lab #1 are now directly applicable to making elegant and aesthetically pleasing tilesets!

Like CSS, CartoCSS apply style rules to map layers by referencing unique IDs (e.g., #waterway_label). You can see in **Figure 11** that default style rules were added to the Map itself, for which there are a small subset of styling rules, as well as each uploaded layer using the unique id you gave it during upload. CartoCSS then is organized into ten different groups of styles based on map features; each group is described as a *symbolizer*, a term derived from the Mapnik rendering engine atop which Mapbox Studio is built. Symbolizers include:

- ***Line***: styles for line features and the strokes of polygons;
- ***Polygon***: styles for the fill of polygons;
- ***Point***: styles for point features;
- ***Text***: styles for labeling point, lines, and polygons;
- ***Shield***: styles for symbol annotation for points and lines;
- ***Line Pattern***: styles for applying textures (e.g., dashing) to lines;
- ***Polygon Pattern***: styles for applying textures to polygons;
- ***Markers***: styles for manipulating iconic point features

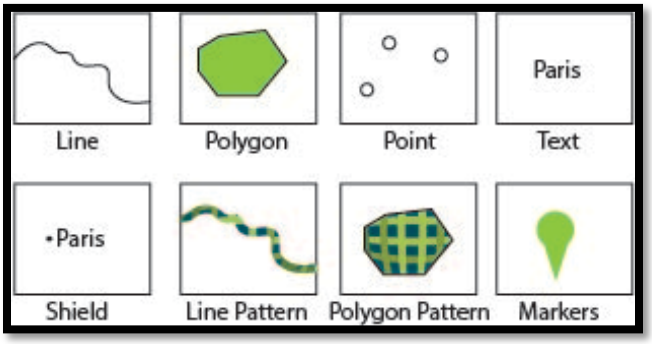


Figure 13: Sample Symbolizers

For Lab #3, you will style your tilesets at every zoom level, which means you will be designing for small and large scale simultaneously. This unique feature of slippy maps means that you have to incorporate your design scheme at the country, state, and road scale. Mapbox Studio's use of CartoCSS is most often used with constricting certain design considerations at different zoom levels. For examples beside this lab, look at the built in example tilesets (BUT DO NOT COPY!).

Let's set up the essentials of the pop-art tileset (**Code Bank 1**). The first things I like to do are create global variables so I don't have to use hex codes every time. I also want to declare my fonts so I can use nicknames to call them when needed. To set up your map,

you must use the Map{} function, which creates a map object to be added to the tileset. There are several things you can do with Map{}, but declaring your background-color (or if you were to have a background-pattern) is typical for most CartoCSS tilesets.

```
1     //colors used a lot
2     @red: #BC243B;
3     @blue: #1D4E89;
4     @yellow: #FBD82B;
5     @black: #000;
6     @white: #fff;
7     @green: #019875;
8
9     @sans: 'Komika Hand Regular';
10    @sans_italic: 'Komika Hand Italic';
11    @sans_bold: 'Komika Hand Bold';
12
13    Map{
14        background-color: #fff;
15     }
16
```

Code Bank 1: Global variables in Mapbox Studio

After declaring global variables, it's time to actually add things to the map! I always start with water because it rarely is different for zoom levels. It's worth mentioning that Mapbox-Streets separates the labels for the layers into a different layer. You won't want to use text functions for non-label layers. To color the water, you just use the selector for water and polygon-fill. You can also change the opacity or use change the polygon-gamma number. Changing the polygon-gamma affects anti-aliasing for polygon features, prevents gaps in the water coloring.

In order to add a border, I have to add a double colon and name this interior statement and can now style the water layer's line properties. From Mapbox's manual: "Within a layer, styles can be broken up into 'attachments' with the :: syntax. Think of attachments like sub-layers. Add attachment to data layers with heavy filtering to keep file sizes low."

```
1 #water {
2  opacity: .75;
3  polygon-fill: @blue;
4  polygon-gamma: .2;
5  ::line{
6    line-color: @blue;
7    line-width: 1.5;
8    line-join: round;
9    line-cap: round;
10  }
11 }
```

Code Bank 2: styling #water in Mapbox Studio

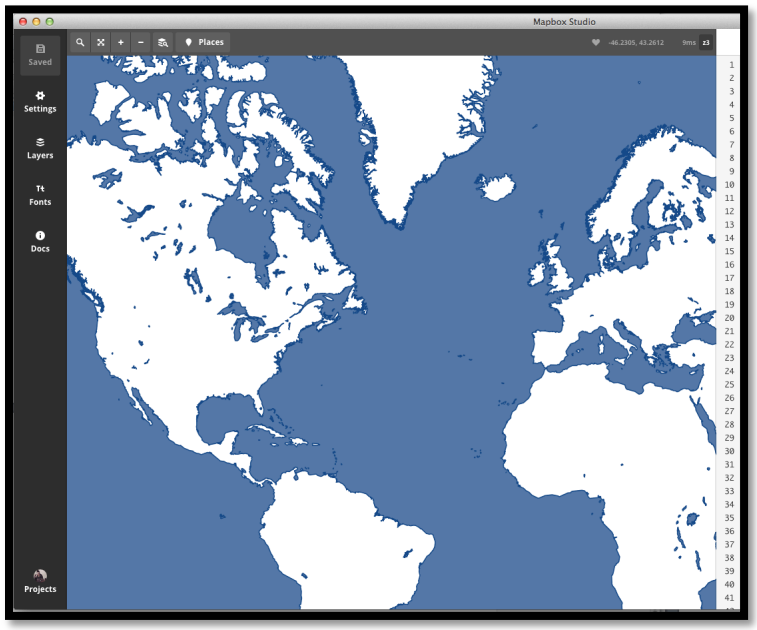


Figure 14: #water drawn in Mapbox Studio

The next thing to style are the country lines. The #admin selector has various attributes that can be used to specify my styling. If you look at the layer tab (**Fig 9**), you'll see that the admin_level can be used to specify whether the administrative lines are for country, state, region, county, etc. Having it less than or equal to 2 guarantees that only country lines are showing. I also don't want the borders on the water showing, so I can use the maritime attribute to only show the borders that don't touch water.

To have a line show up, you have to declare the line-color and line-width. Because borders can appear jagged sometimes, I set the line-cap and line-joins to be round and add a line-smooth element. Also, to increase the line width at a higher zoom, I just need to use the zoom selector and set the line-width to a higher number at different zooms.

```
#admin[maritime=0][admin_level<=2]{
 line-color: @red;
 line-cap: round;
 line-join: round;
 line-smooth: .75;
    [zoom>=3]{
         line-width: 1.15;
    }
    [zoom>=4]{
     line-width: 1.5;
    }
}
```

Code Bank 3: Styling country lines in Mapbox Studio

Figure 15: country lines added to Mapbox Studio

Now that we've explored how to style lines and polygons, it's time to learn how to label. As mentioned earlier, layers are separated in Mapbox-Streets by the geometry and the labels. This makes it much easier to manage and avoid having to constantly use those attachments. If you look in the documentation tab, you'll notice that under the text selectors, there are two different key words: text or shield. We're going over both of these two symbolizers, starting with text for country labels.

Text requires text-name and text-face-name in order to work. The logic is that you have to declare what typeface will be displayed and what text actually should be displayed as well. You can choose which option you want displayed, whether you want the local name of the country (i.e. Middle Eastern countries will be in Arabic, the USA + UK will be in English, etc.) or their English, French, German, or Spanish names (**Fig 16**). Since you're calling a specific object in one of those arrays, it gets formatted like 'text-name: [name_en];'. Recall the global type variables we declared in Code Bank 1? We'll use this for text-face-name. Simple call your @sans_bold to choose a bold typeface for the country labels. While not necessary, you probably want to customize your text-fill and text-size. For all of my text, I make a text-fill that's @white and use a @text-halo-fill that's black with a text-halo-radius of 2 to make it look like a comic book. Halos work just like a stroke in Illustrator in this instance. To put my text on an angle and customize spacing in between lines and characters, I used lines 8-11 of Code Bank 4. Experiment using various text functions to get your labels to be where you want.

A common concern with adding country labels is changing the text size of various countries in order to not make the map look too busy. Just like the label challenge lab in 370, adding country labels at a high zoom requires using the country's relative scalerank for placement of labels. Mapbox Studio has an attribute of scalerank to use, unlike in 370 where it was by

the cartographer's discretion. Lines 12-18 of Code Bank 4 show how you might alter the scaleranks. Again, use your knowledge of cartographic label placement rules and experiment.

Figure 16: Text-name in #country_label

```
1 #country_label[scalerank<=8]{
2   text-name: [name_en];
3   text-face-name: @sans_bold;
4   text-size: 12;
5   text-fill: @white;
6   text-halo-fill: @black;
7   text-halo-radius: 2;
8   text-orientation: 5;
9   text-character-spacing: -1;
10  text-wrap-width: 80;
11  text-line-spacing: -10;
12  [scalerank<=1]{
13    text-size: 24;
14  } [scalerank=2]{
15    text-size: 18;
16  } [scalerank=3]{
17    text-size: 14;
18 }
19 [zoom>=4]{
20    [scalerank<=1]{
21      text-size: 30;
22    } [scalerank=2]{
24      text-size: 24;
25    } [scalerank=3]{
```

```
26       text-size: 18;
27       }
28     text-size: 16;
29    }
30 }
31
32 #country_label_line {
33     line-width: 3;
34 }
```

Code Bank 4: Styling country names in Mapbox Studio

The rest of Code Bank 4 adjusts the text-size at higher zooms, where it will better to scale up the country label. For countries that are labeled in the water (You can check this under source view), you'll want to add a #country_label_line. Line are very easy to style in this regard, you only need to define a line-width, but there are certainly plenty of functions you could use. Now that you can label using the text selector, try labeling the oceans. Start with designing for Zoom 3, but add in statements to adapt the labels for higher zooms. Not every layer has an attribute like scalerank, but you can either use another attribute or adjust it based on zoom.

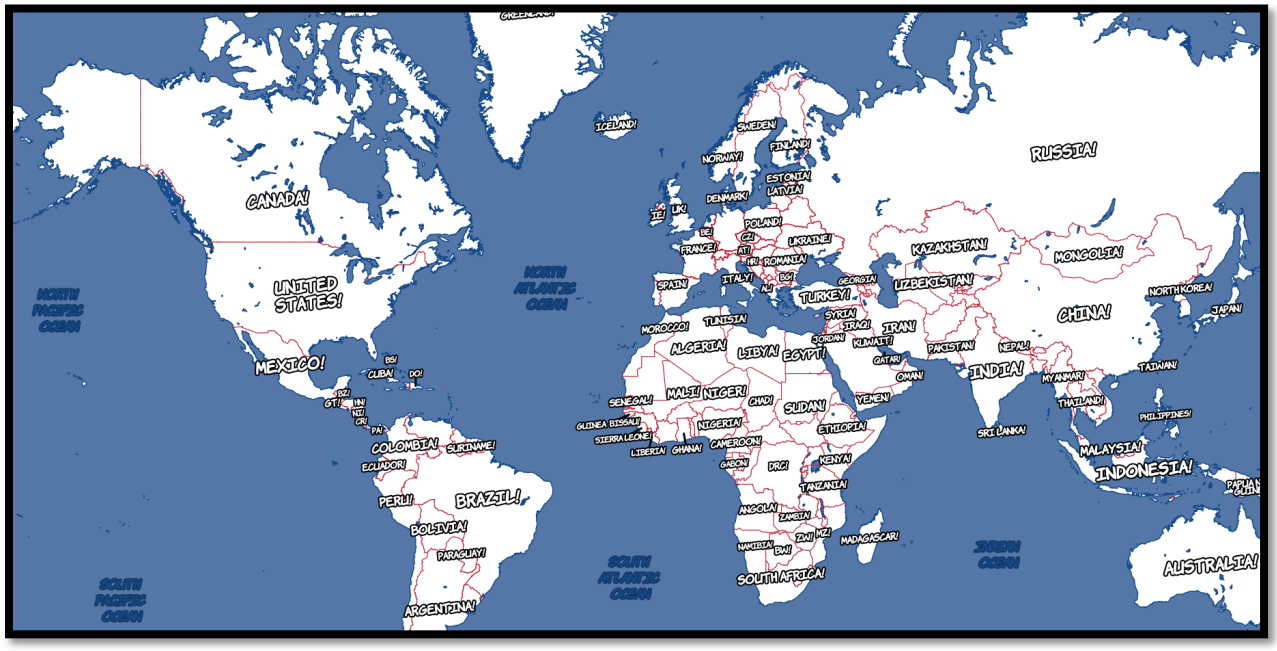


Figure 17: Country Labels and Ocean labels added to zoom 3

Figure 20 shows how much you should have on your map by now. As you can see, there's a lot of white, which is a little too boring for our purposes of creating a Lichtenstein style map. When I was choosing my sources (**Fig 7**), I also added Mapbox Terrain, so I could style the land based off of landcover patterns. Mapbox Terrain includes hillshade, contours, and landcover (Mapbox Satellite and watermask have even more options, test out the different additional sources for how you can style your map and have it fit with your inspiration), so

I can use polygon selectors to style how these look. This is where I can finally use the halftone texture pattern I added in the .tm2 folder. Once you know how to design landcover, you can use the same logic for the landuse and building layers.

```
1 #landcover[class='scrub']{
2   polygon-fill: @green;
3   polygon-opacity: .35;
4   polygon-pattern-file: url("img/halftone5.svg");
5   polygon-pattern-comp-op: soft-light;
6 }
7
8 #landcover[class='snow']{
9    polygon-fill: @blue;
10   polygon-opacity: .45;
11   polygon-pattern-file: url("img/halftone5.svg");
12   polygon-pattern-comp-op: overlay;
13 }
14
15 #landcover[class='grass']{
16   polygon-fill: @green;
17   polygon-opacity: .5;
18 }
19 #landuse[class='park'],[class='wood']{
20   polygon-fill: @green;
21   polygon-pattern-file: url("img/halftone5.svg");
22 }
```

Code Bank 5: Styling landcover in Mapbox Studio

For all of these different landcovers, I choose to style them as polygons, testing which area is going to cover the most surface area, which turned out to be grass. I simply added a polygon-fill and make it 50% transparent. Scrub landcover gets layered over the grass, so I'm using this to add a halftone pattern. First I add a polygon-fill that's a very transparent green. In order to add my pattern, I have to add the url reference. I put the halftone svg (all image types are acceptable, I just used svg in Illustrator based on personal preference) in a folder inside the .tm2 folder called img and just have to follow the example in the Docs tab.

This is an opportunity to play with Mapbox Studio's use of comp-op (short for compositing operators from Photoshop) tools. To understand what each does, there's documentation on Mapbox's website here: https://www.mapbox.com/guides/compositing-reference/ about the differences of each operator. For those familiar with Photoshop, you can play around with specific operators you want to use. For this landcover, I'll be adding the pattern with a soft-light comp-op that makes the pattern a lot less dramatic and harsh to look at. It's softer than the overlay mode, which is used on the snow landcover in lines 8-13. I add one last layer using the landuse layer for woods and parks to follow the same pattern for scrub, but

without transparency or the comp-op. This has a three-layer effect with green overlays, making a beautiful way of symbolizing natural features, as seen in Figure 18.

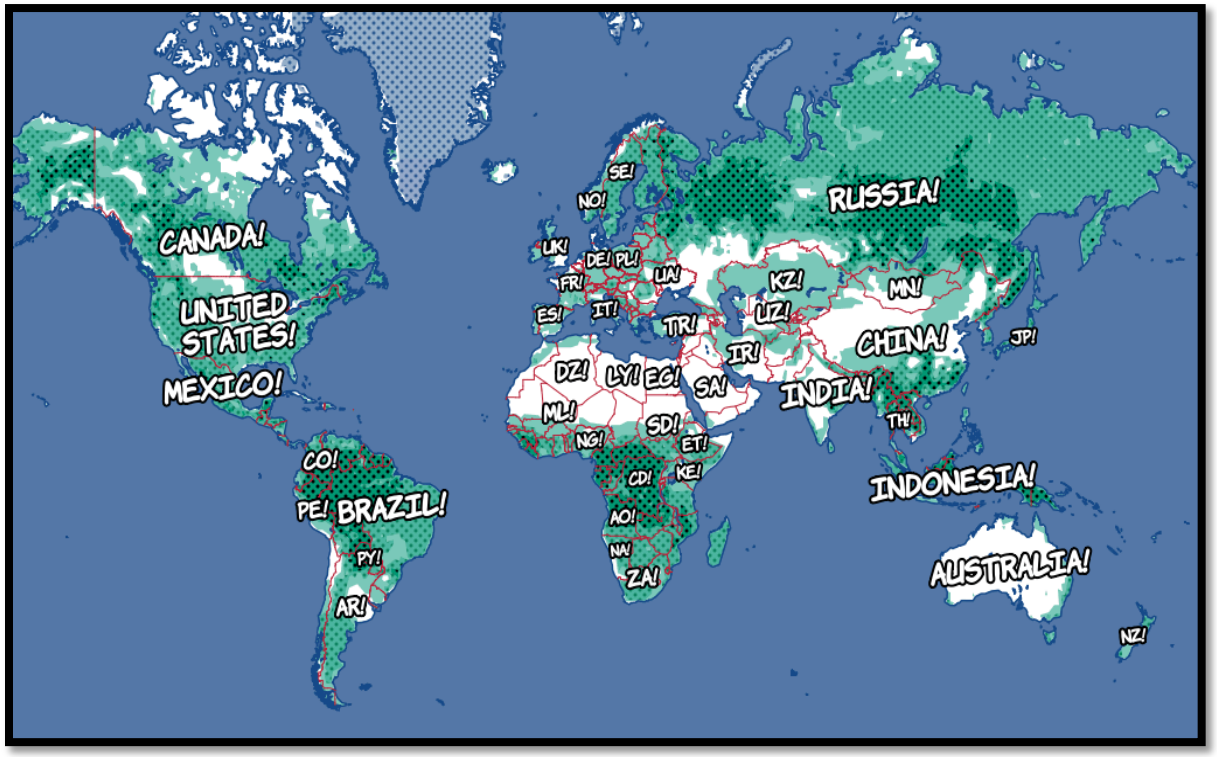


Figure 18: Adding landuse and landcover patterns for natural features

The same theory for landuse and landcover can be used for other types of landuse features. Experiment using different colors in your color palette and different features in the documentation. A fascinating feature of Mapbox Studio is the building-fill, and building-height functions that can be used with #building in urban areas. This is separated from polygons because of the limitations to styling buildings and the ability to add a height to them.

As our map starts to come together at a world view, it's time to add things that will be added at higher zooms like city names and the aforementioned shield. Shields incorporate the image file being used as the city marker and the text next to it, labeling the location. This is why the operators are a little different for shield versus text. Unlike in Illustrator or ArcMap, where you might position manually where every label will go for a city, you use the shield-text-dx and shield-text-dy to move the label around. The layer with the city labels has a built in attribute called 'ldir' which assigns a label placement direction that are similar to those taught in 370. Using those along with the shield-text-dx/dy will allow you to move label placement around the shield easily.

Code Bank 6 shows the complete code required for adding a city label for any city with a scalerank larger than 2 (this will only show capitals) and once the zoom is larger than 13,

this code is defunct (at that close of a zoom, you don't really need city labels). Similarly to adding patterns, I have to link the file I'll be using as my shield. I also need to use shield-face-name and shield-name like I did when adding country labels in Code Bank 4.

```
1  #place_label[type='city'][scalerank<2][zoom<=13]{
2    shield-file: url("img/dot.svg");
3    shield-face-name: @sans;
4    shield-name: "[name_en]";
5    shield-unlock-image: true;
6    shield-transform: scale(.5,.5);
7    shield-placement: point;
8    shield-size: 12;
9    shield-text-dx: 3;
10   shield-text-dy: 3;
11   shield-fill: @white;
12   shield-halo-fill: @black;
13   shield-halo-radius: 1.25;
14   [ldir = 'N'],[ldir = 'E']{
15     shield-text-dx: 3;
16     shield-text-dy: 3;
17   }
18   [ldir = 'W']{
19     shield-text-dx: -3;
20     shield-text-dy: 3;
21   }
22   [ldir = 'S']{
23     shield-text-dx: 3;
24     shield-text-dy: -3;
25   }
26   [zoom>=6]{
27     shield-size: 16;
28   }
29  }
30
```

Code Bank 6: Styling shields + shield-text in Mapbox Studio

In order to have my city name adjacent to the shield and not have the text on top of the shield, I have to set shield-unlock-image as true. Also, depending on how your marker is sized, you'll need to use the transform function to scale it down. I also want to place it as a point label, so I use the shield-placement function. After setting the design of my places with shield-size, shield-fill, shield-halo-fill, and shield-halo-radius, I want to define the shield-text-dx/dy. The best way to learn how to place shields and have them change with ldir is with practice. Experiment with this until you're satisfied; looking at the sample templates provided by Mapbox and see how other designers use the shield. At closer zooms, add more #place_labels of smaller cities, towns, or even villages.

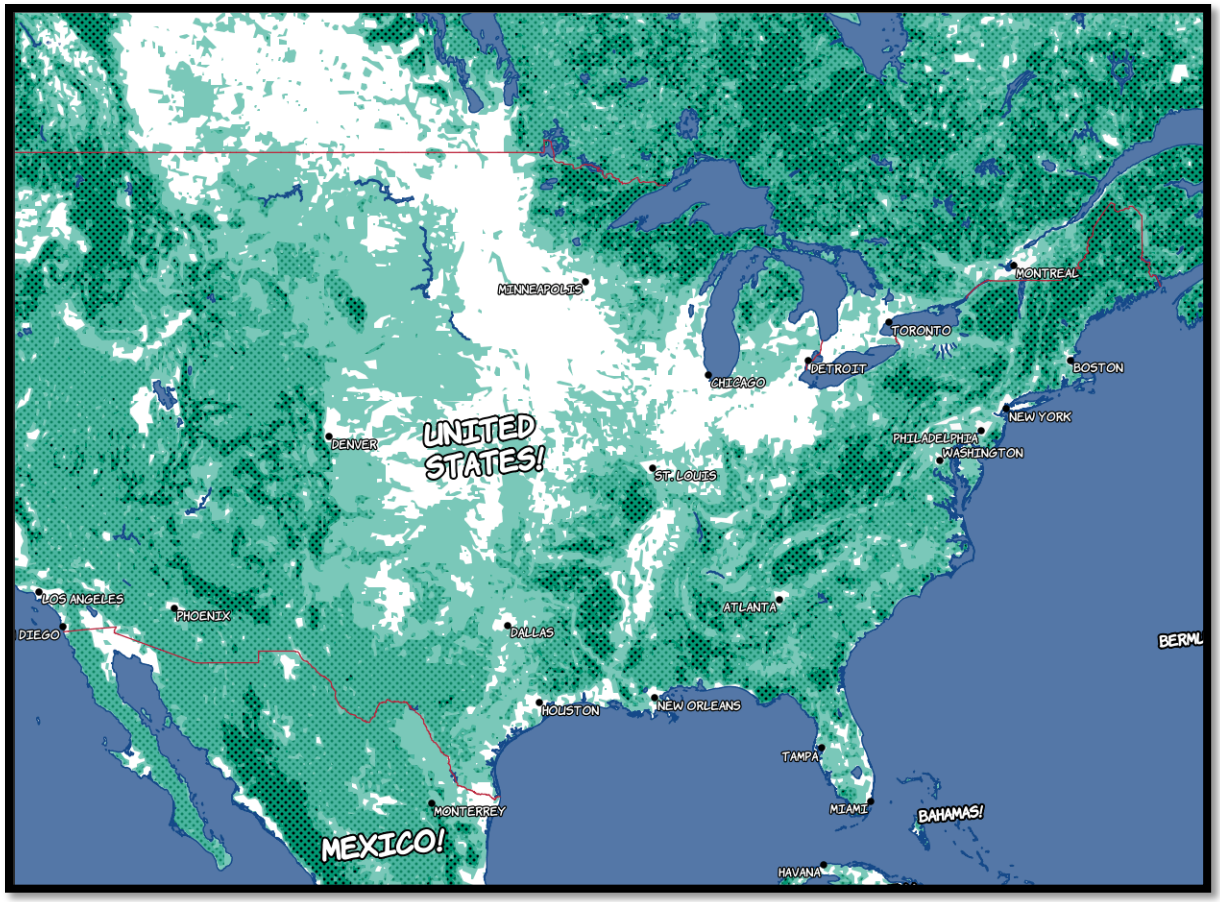


Figure 19: Shields and shield-text added to major US cities

Lastly, we have to add roads so the city view looks realistic. When designing at this point, it's useful to switch back and forth into the Places view and see how your design works around the world. Recall on Page 12 the discussion about attachments; with roads this is where we want to add this to create an outer thick black line and a interior white line. This mimics the text with the black halo and white fill. Adding a line-join and line-cap that's round can also help to remove jagged edges that occur. Roads have a ton of attributes that be utilized to style differently, take advantage of that to make different size and color roads.

```
1 #road[zoom>=6] {
2  line-join: round;
3  line-cap: round;
4    ::case {
5      line-width: 3;
6      line-color: @black;
7    }
8    ::fill {
9      line-width: 1.5;
10      line-color: @white;
11    }
```

```
12   [zoom>=14]{
13     ::case {
14     line-width: 4;
15     line-color: @black;
16     }
17     ::fill {
18     line-width: 2;
19     line-color: @white;
20     }
21   }
22 }
```

Code Bank 7: Styling roads in Mapbox Studio

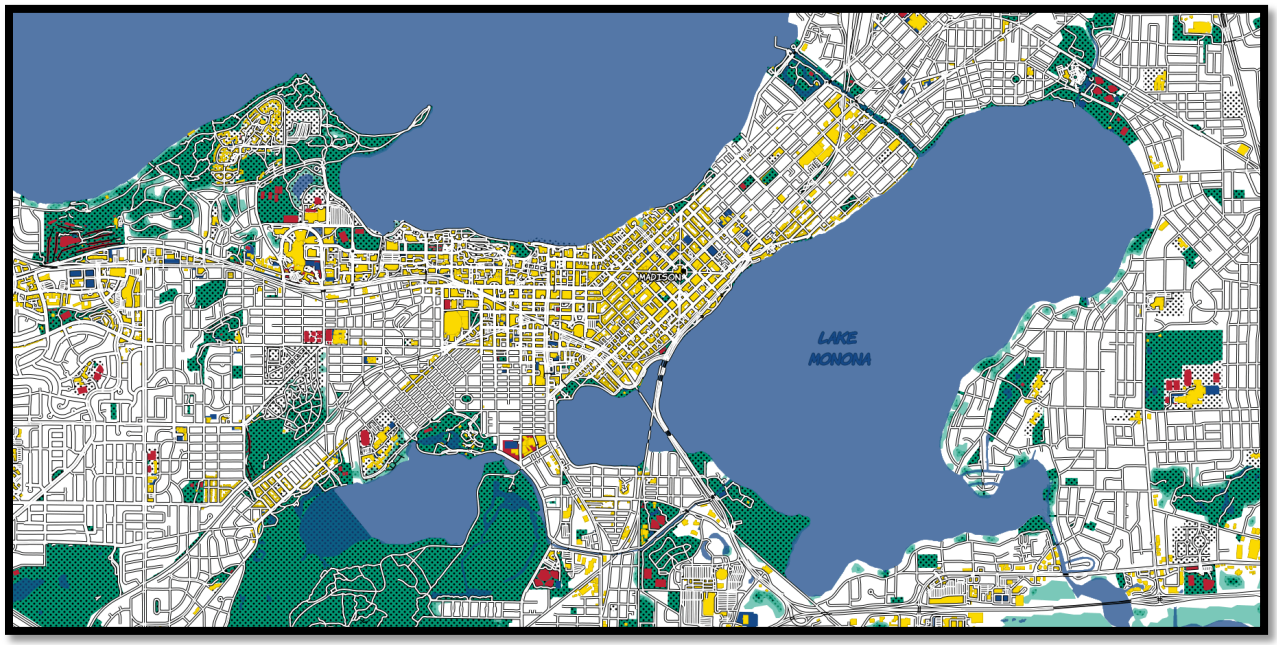


Figure 20: Roads added at a high zoom level of Madison, WI

Congrats! Look at what you've accomplished in just a few lines of CartoCSS! Keep playing with different layers and styling things. When you feel like you're done, move on to Section E to upload your map and create a functional slippy map.

e. Uploading to Mapbox

After saving your final tileset, it's time to upload to Mapbox and serve your tiles. Under settings, you'll see a bright blue button that says 'Upload' and after it's completed, you can go to Mapbox.com and look at your uploaded tiles. On their website, you'll see Projects, Styles, and Data next to your username when logged in. We're going to create a new Project with this style, so click on Styles to see your uploaded tileset. If it's not showing up right away, you can view the uploads page to see how long it's going to take. Click on the new project and you'll see your description (if you added one in Settings in Studio), the zoom levels it supports and 'New Project'. Click on that and open up Mapbox Editor

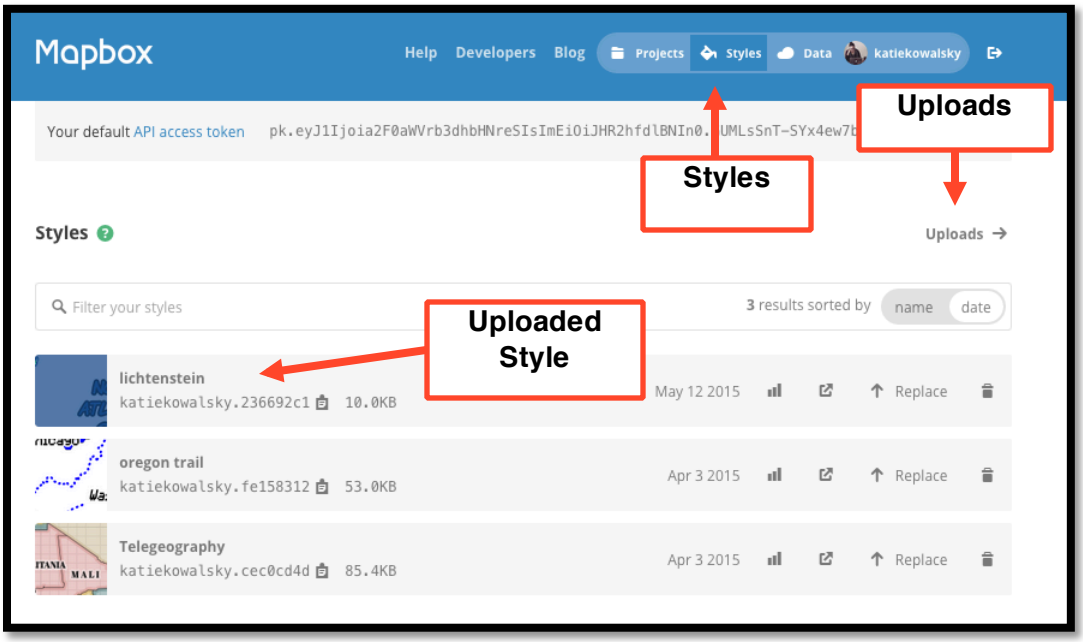


Figure X: Styles Page on Mapbox.com

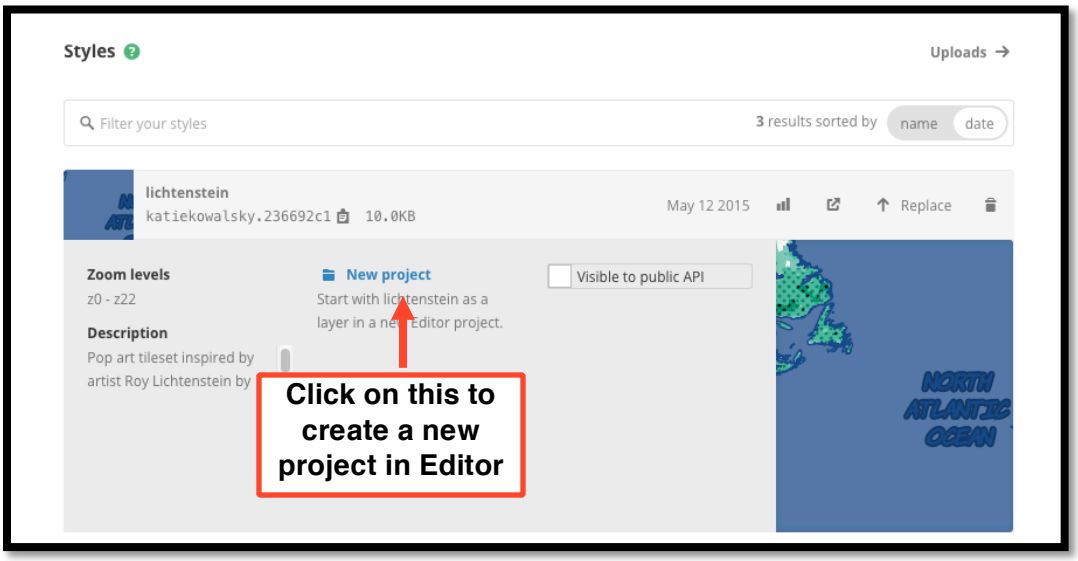


Figure X: Creating a new project on Mapbox.com (Clicked on Uploaded Style)

f. Mapbox Editor

Mapbox Editor is an online Graphical User Interface (GUI) that allows users to make a webmap without any javascript knowledge. There are three components to Editor, the Style tab, Data tab, and Project tab. The style tab is where you would choose your tileset (either one of Mapbox's default styles or your own in this instance). The Data tab is where all of your data is managed. You could add markers, lines, or polygons one at a time, or import geoJSON, CSV, KML or GPX files. There's also a layer view button to view every data point and edit it. Lastly, the Project tab has settings for the name and description, and the ability to embed this in a webpage without any code knowledge. Let's get started with importing data.

Since you're opening your tileset in a new project, you've already bypassed the 'Style' Tab in Mapbox Editor and the Data tab should be opened by default. Press import and navigate to your CSV that you created earlier and press Open.

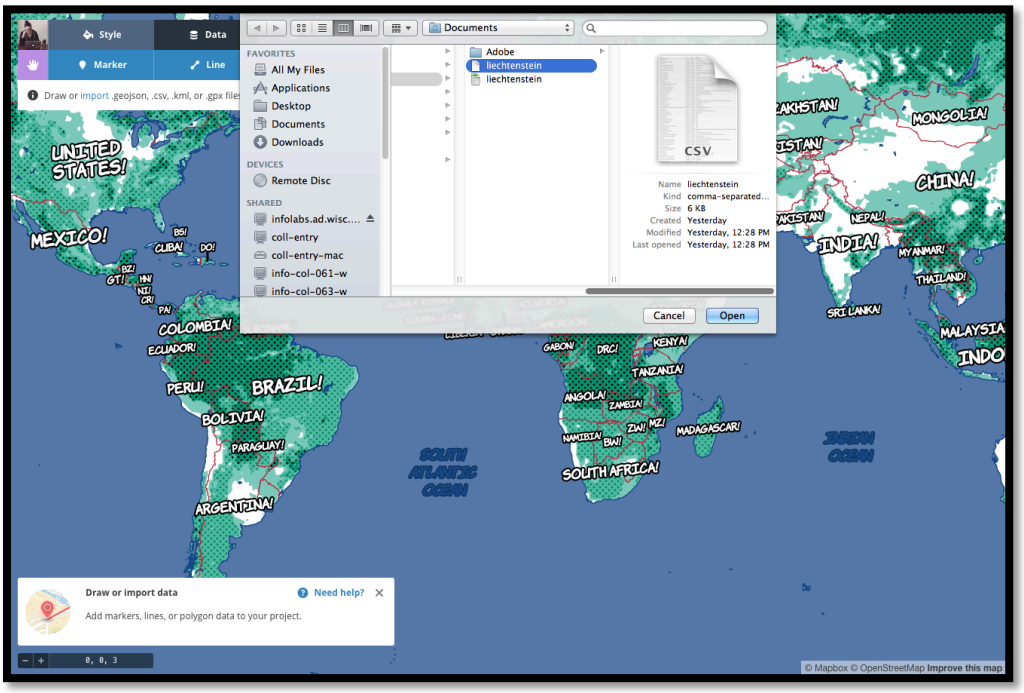
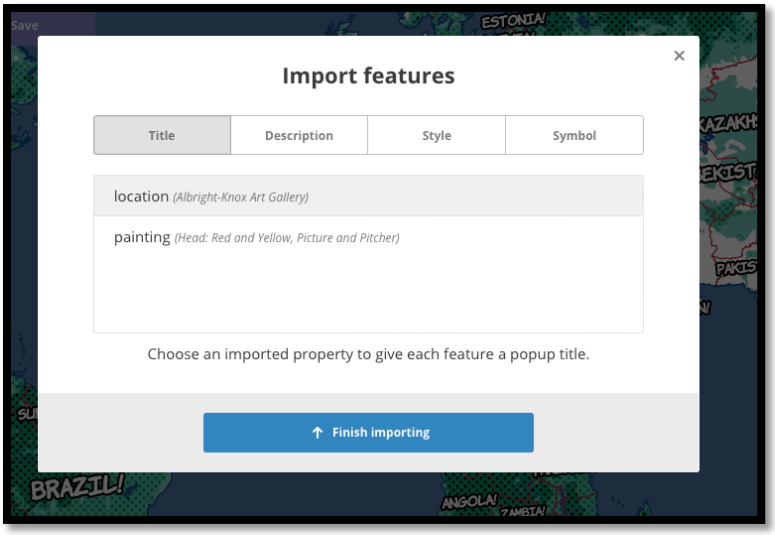
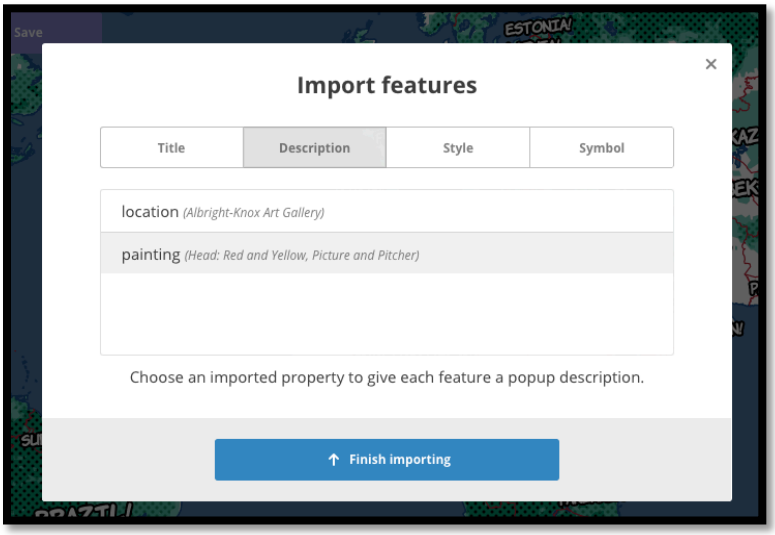


Figure X: Import CSV data into Mapbox Editor

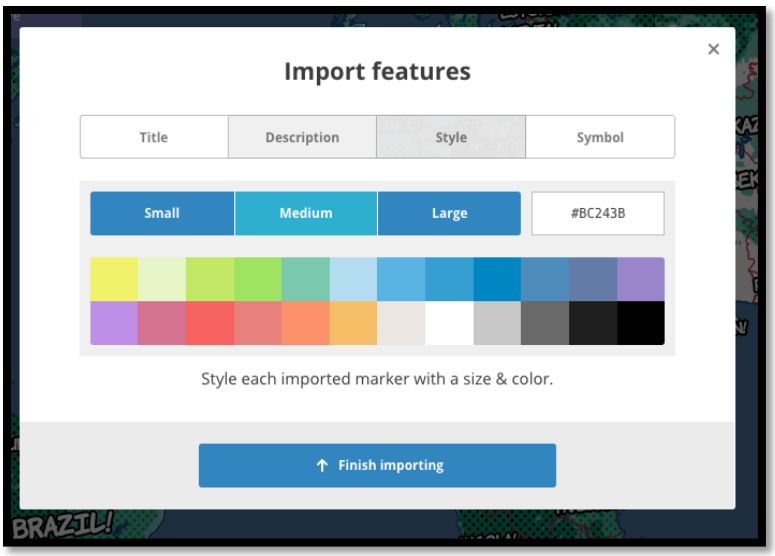
When you import your CSV, you'll be prompted to the import features windows, which apply the settings for all of your markers. Do this now, otherwise changing style for all of the markers has to be done one at a time. There are four steps you go through for importing and designing your data pop-ups. They are illustrated on the next page.



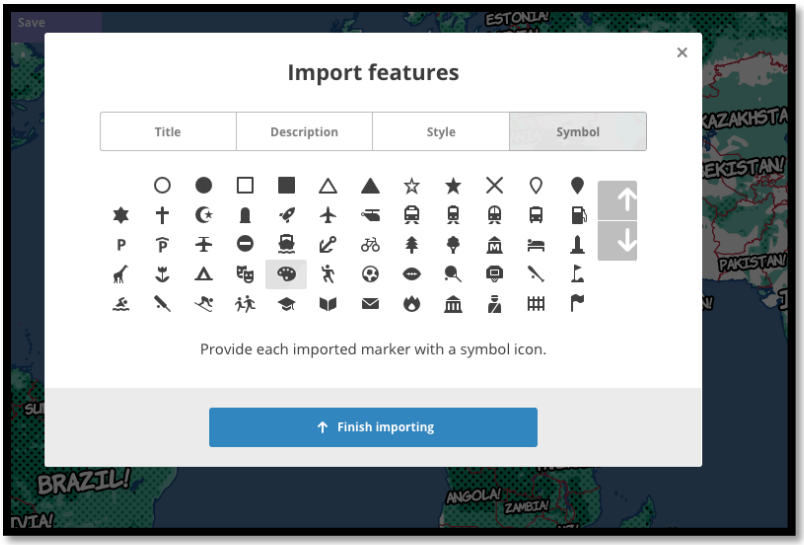
Step One: Choose your title. The importer tool allows you to choose any column to be your header that isn't used for latitude & longitude. In this instance, I'm going to choose the location name.



Step Two: Choose your description. This will be your description you added in the CSV



Step Three: You can choose the marker size and color (feel free to use your own hex code in the input window to the right of the size prompt).



Step Four: If you want, you can add a symbol icon denoting the type of place your marker represents.

Once you've pressed 'Finish Importing', you'll have all of your points added to your map:

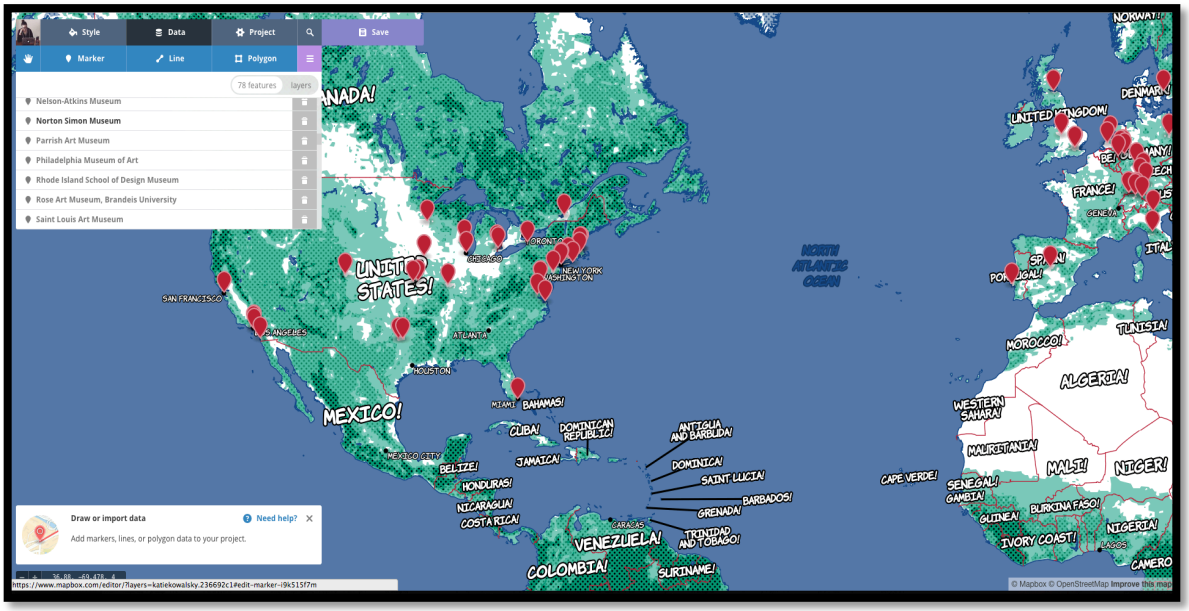


Figure X: Map with imported features

Clicking on the layers button (the purple button next to Polygon) in the Data tab will show you a list of all the features added to the map. Double check that everything is spelled correctly and you can also edit your description or the style of the marker if necessary. Save your project and go to the Project tab & under Settings you can add a description and name your project. You'll also see code to embed your map on a website.

Evaluation Rubric: Slippy Map Iconicity Challenge (40pts)

Inspiration Board Check-In (6)

5.5-6.0pts: The icon library is ready for use in TileMill. All four of the above considerations for icon design were taken into account. All icons hold up when rasterized into the *.png* format.

4.5-5.0pts: The icon library needs a few small tweaks before it is ready for use in TileMill. One of the four above considerations for icon design was not fully taken into account during design. Several icons break down when rasterized into the *.png* format.

3.5-4.0pts: We need to rethink aspects of the icon library before it is ready for use in TileMill. Several of the four design guidelines were violated systematically. Multiple icons break down when rasterized into the *.png* format.

3.0 pts or less: The submitted icon library did not meet the expectations of the assigned challenge, or was not submitted at all.

Due November 6th

POIs Check-In (6)

2.0pts: You have assembled your *.csv* file.

0pts: You have not assembled your dataset

Due November 13th

Mapbox Studio Tileset (32): Due Online November 20th

30-32pts: The slippy map mashup is attractive, informative, and engaging; the map is a vast improvement over most web map mashups. The icon library unambiguously signifies the POIs and their higher-level categorization. The icons themselves are at the highest level of the visual hierarchy and hold up at all map scales. Your tileset is loading properly and reinforces the aesthetic style of your icon library. The tooltip and legend solutions are working properly and are well-designed. The aesthetic style of the tileset is consistent across zoom scales; what a stunning example of multiscale map design!

26-29pts: The slippy map mashup overall is successful as a web map. The icon library mostly works in representing the POIs and their higher-level categorization, but there are one or two icons that remain problematic. The icons work against the basemap and hold up at most, but not all, map scales. Your tileset is loading properly, as are the tooltip and legend solutions, although these solutions could be improved with further refinement. The aesthetic style of the tileset is mostly consistent across zoom scales, but there are unrefined transitions in linework styling between one or two zoom level changes.

22-25pts: The slippy map mashup only marginally improves upon typical web maps. Significant problems remain in the icon library, both in the representation of the unique POIs and their higher-level categorization. Many of the icons do not work against the basemap and/or across map scales. Your tileset is loading properly, but the tooltip and/or legend solutions are non-functional or illogical. Little effort was taken to develop an aesthetic style that is consistent across scales. Much more work needs to go into your linework generalization as you go from a large to a small cartographic scale (e.g., elimination of unneeded detail, simplification of lines and polygons, restyling the sizes or colors of features to maintain a proper visual hierarchy).

Below 21pts: The submitted slippy map mashup did not meet the expectations of the assigned challenge in multiple and critical ways. Please speak with Rob and Rashauna about strategies to improve the design.