Customizable Leaflet Maps

TDHI 2019

Jo Klein - GIS & Data Vis. Librarian
Maggie Murphy - Instruction & Humanities Librarian

UNC GREENSBORO
University Libraries

Download workshop materials from Google Drive at:

- go.uncg.edu/tdhi19-leaflet

Right-click on the "Demo map files" folder to download, then unzip/extract.

Objectives

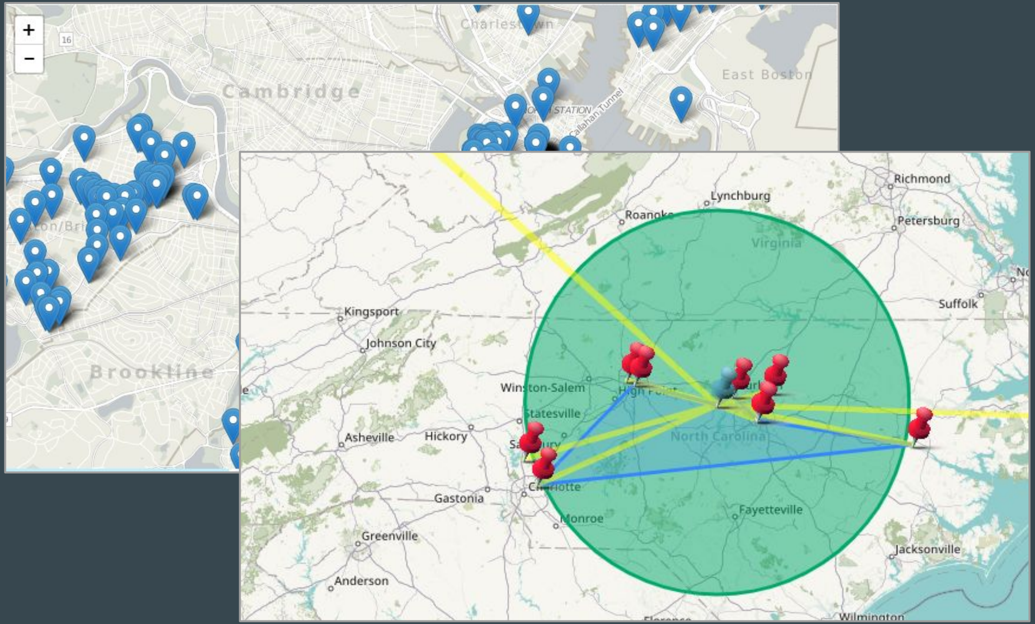
1. Learn what Leaflet is and why you would use it.
2. Explore the parts of a Leaflet map using an example.
3. Learn how to use Leaflet (and regular expressions) by making changes to the example map.

What is Leaflet?

Leaflet is an open-source Javascript library for creating mobile-friendly interactive maps.

How is Leaflet used?

Leaflet code is used alongside HTML, CSS, and Javascript in web-page or app development.



- [Example: Gropper's America]
- [Demo Map: Normal]

What does Leaflet do?

- Showing and interacting
  - Panning
  - Zooming
- Tiled base layers
- Feature layers (from the user, i.e. you!)
- Mobile-friendly maps

What does Leaflet *not* do?

- Provide data
- Analyze data*
- Map projections & manipulations
- User interface*

*on its own

Why use Leaflet?

- If you want a simple interactive map without complicated software
- If you want to learn or teach web development and coding
    - HTML & CSS
    - JavaScript/JavaScript Object Notation (JSON)
- If you want to teach or practice implementation of accessibility & web design guidelines
    - Mobile-friendly and works across devices
    - Visual design elements
- If you want to explore web-scraping and use of APIs.

Making a Leaflet Map

You'll Need:

1. Demo map files
    - http://go.uncg.edu/tdhi19-leaflet
2. Text editor
    - Notepad (Windows)/TextEdit (Mac)
    - JS Fiddle (in-browser)
    - Sublime Text
3. Web browser

~~4. Local web server~~
   - ~~Python's SimpleHTTPServer~~
   - ~~WampServer (Windows)/MAMP (Mac)~~

For next time

Parts of a Leaflet Map*:

1. An HTML page
    - map-normal.html
    - map-watercolor.html
2. Leaflet CSS styles
3. Leaflet JavaScript library
4. Images for custom markers (optional)
5. Geographic coordinates!

*Alternatively: Contents of the "Demo map files" Folder.

Using regular expressions and Sublime Text

1. Open find & replace with Ctrl+H
2. Delete the pronouns since they don't figure into the code:
   a. Find all: `,(.+? / .+? / .+?),|(,,)` Replace all: `_<a href="`
3. Switch the order of the URL and the rest of the text:
   a. Find all: `"(.+),(https://.+)` Replace all: `"$2"> $1`
4. Clean up coordinates:
   a. Find all: `(,.+°.+W,)(.+),(.+)` Replace all: `_[$2, -$3]`

Handy resources for writing regular expressions:

- Quickstart guide to regular expressions
- RegExr, a tool to test and explore a regular expression

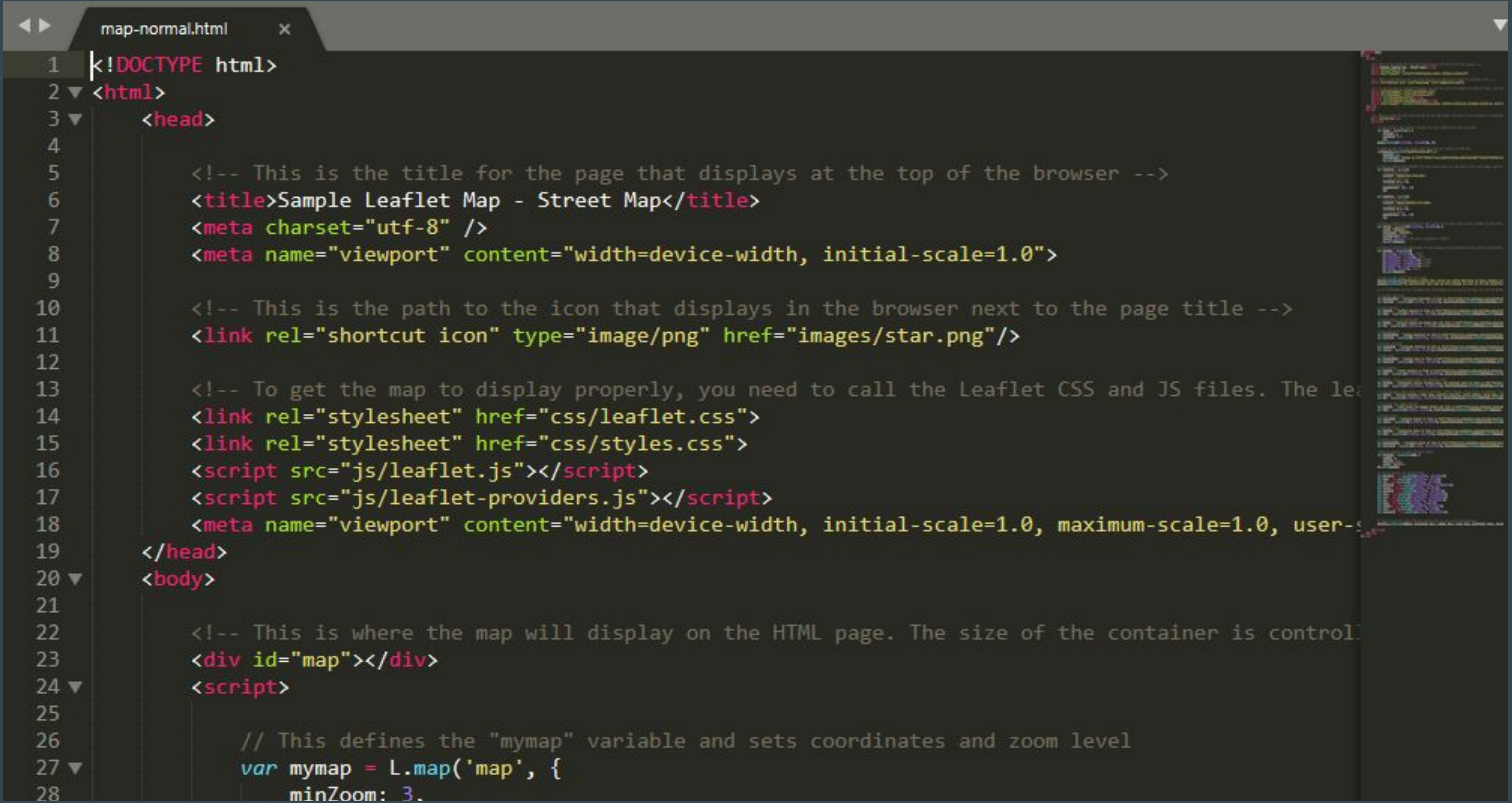
Keyboard Shortcuts:

- Ctrl+A = Select all
- Ctrl+Shift+L = Multiple cursors on selection
- Ctrl+→ = Move cursor(s) right (or up, left, down)
- Ctrl+Shift+→ = Select to the right (or up, left, down)

Parts of a Leaflet Map*:

1. An HTML page
   - map-normal.html
   - map-example.html
2. Leaflet CSS styles
3. Leaflet JavaScript library
4. Images for custom markers (optional)
5. Geographic coordinates!

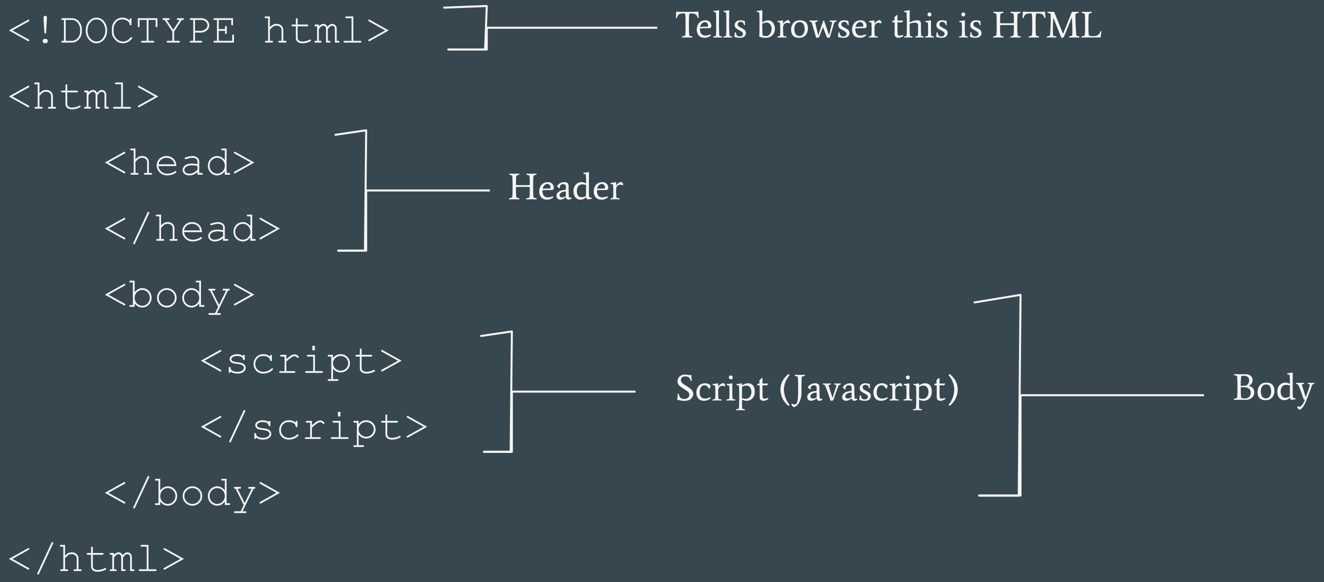
*Alternatively: Contents of the "Demo map files" Folder.



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

        <!-- This is the title for the page that displays at the top of the browser -->
        <title>Sample Leaflet Map - Street Map</title>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <!-- This is the path to the icon that displays in the browser next to the page title -->
        <link rel="shortcut icon" type="image/png" href="images/star.png"/>

        <!-- To get the map to display properly, you need to call the Leaflet CSS and JS files. The lea
        <link rel="stylesheet" href="css/leaflet.css">
        <link rel="stylesheet" href="css/styles.css">
        <script src="js/leaflet.js"></script>
        <script src="js/leaflet-providers.js"></script>
        <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-
    </head>
    <body>

        <!-- This is where the map will display on the HTML page. The size of the container is control
        <div id="map"></div>
        <script>

            // This defines the "mymap" variable and sets coordinates and zoom level
            var mymap = L.map('map', {
                minZoom: 3,
```

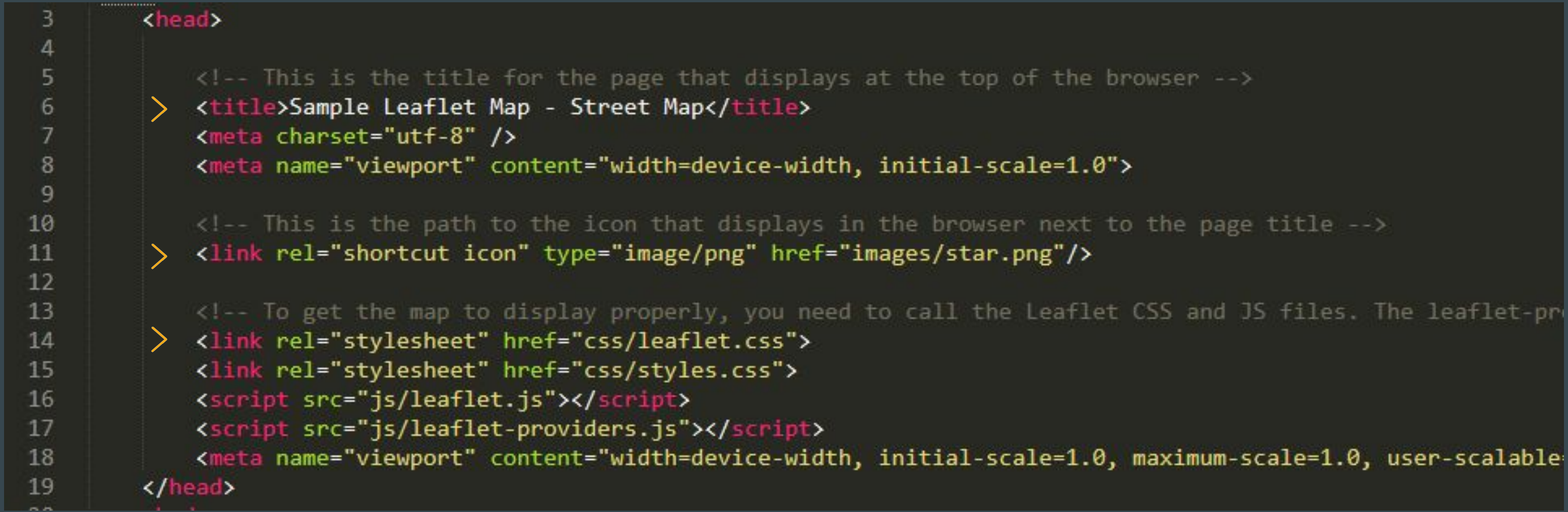
Leaflet Reference/Documentation

Structure of an HTML webpage



```
<!DOCTYPE html>
<html>
    <head>
    </head>
    <body>
        <script>
        </script>
    </body>
</html>
```

1. Header



```
 3  <head>
 4
 5      <!-- This is the title for the page that displays at the top of the browser -->
 6  >   <title>Sample Leaflet Map - Street Map</title>
 7      <meta charset="utf-8" />
 8      <meta name="viewport" content="width=device-width, initial-scale=1.0">
 9
10      <!-- This is the path to the icon that displays in the browser next to the page title -->
11  >   <link rel="shortcut icon" type="image/png" href="images/star.png"/>
12
13      <!-- To get the map to display properly, you need to call the Leaflet CSS and JS files. The leaflet-pr
14  >   <link rel="stylesheet" href="css/leaflet.css">
15      <link rel="stylesheet" href="css/styles.css">
16      <script src="js/leaflet.js"></script>
17      <script src="js/leaflet-providers.js"></script>
18      <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable
19  </head>
```

line

6 Change the page title

11 Change the page icon to **star.png**

14 Add the filepaths to the Leaflet CSS and JS files

2. Body - Initializing map, customizing view & markers

line

- 23 Add map container: `<div id="map"></div>`
- 27 Creating a map, setting the minimum zoom: give your map variable a name (using find-all & replace), or leave it set to **mymap**. Change the minimum zoom to **3.**
- 31 Customizing the map view: set your map coordinates to Davis Library's coordinates (in brackets)
- 34 Customizing base maps: we'll come back to this one!
- 42 Customizing marker icons: add the filepath to **red-icon.png** under the **testIcon** variable.
- 51 Add the filepath to **default-icon.png** under **homeIcon.**

2. Body - Adding coordinates, popups, & shapes

line

59 Adding a circle: add the coordinates to Davis Library (in brackets) to set the centerpoint of a circle.

68 Adding a polygon: Add coordinates (in brackets) for the corresponding institutions from the data table as points in the polygon.

77 Adding popups to shapes: bind popups to the appropriate shape variables

88 Define the popup variable for each institution using the dataset, using UNCGPop (line 84) as a template.

89 Define the coordinates and marker for each institution, bind popups to the markers, and add to your map using the dataset, using UNCG (line 86) as a template.

2. Body - Adding lines

line

134 Define a polyline variable named Polyline, and change its color to 'yellow'.

142 Define a new variable for each institution, with corresponding coordinates (in brackets) from the dataset, using uncg as a template.

156 Set coordinate pairs for endpoints of the polylines for each unc to other institution pair (keep all brackets, remove "...").

Save, and double-click file to open in browser.

What happened? Check for mymap!

2. Body - Changing the basemap

line

34 Replace the tile provider name with one from the Leaflet Providers list (we used 'Stamen.Watercolor').

36 Copy & paste over the attribution field from the plain javascript code for that provider.

Save, and double-click file to open in browser.

What's an API key?

An application programming interface (API) key is a **unique identifier** used to **authenticate** a program, developer, or user to a website's API.

Commonly used to:

- Prevent malicious use or abuse of the API
- Identify the entity using the API.

Resources and Tutorials

- Maptime Boston: Leaflet intro
- MaptimeTO: Leaflet basics
- Leaflet's website