Mapping API's: Leaflet - Custom Image Markers

Welcome to the Essential ArcGIS Task Sheet Series. This series supplements the Iowa State University GIS Geospatial Technology Training Program short course series. The task sheets are designed to provide quick, easy instructions for performing mapping tasks.

This task sheet builds upon the previous task sheet: *Mapping API's: Leaflet - Getting Started* **PM2082-14r** and *Mapping API's: Leaflet - Adding Markers* **PM2082-14s**. Please refer to those documents for the initial Leaflet map setup. The code for this task sheet (**customMarkersLeaflet.html**) and the previous task sheets can be found on the ISU Geospatial Technology Program GitHub page at https://github.com/ISUEOGTP/GISTaskSheets. This task sheet will take you through the steps of customizing icons using your own icon images.

1. Introduction

a. An alternative option to the default blue teardrop marker icon used by Leaflet is to display points using your own icon images. This can be done by defining a custom icon using an image.

b. The image to the right shows the results of the *Mapping API's: Leaflet - Getting Started* task sheet. In the next steps we'll add a custom image marker with pop-up.

2. Define the Icon

a. Image markers are created by first creating an icon instance and then defining the options or properties of the icon. There are ten options that can be set for an image marker. Though only the iconURL option is required. For the purpose of this task sheet we will use just 7 of these options. Refer to http://leafletjs.com/reference.html#icon for a full list of options.

b. Start creating your icon by creating a new variable and setting it equal to **L.icon**.

```
var greenIcon = L.icon({});
```

c. Now you will be adding the desired icon options to your custom icon. For this demonstration we are going to use the Google Maps green arrow icon. Grab the URL for the icon here http://maps.google.com/mapfiles/arrow.png as well as the shadow image here http://maps.mapfiles/arrowsshadow.png. *Note: of course you could also make your own icon and shadow image. PNG files typically work the best as they allow for transparent areas.*

d. Set the **iconURL** and **shadowURL** options to the Google Maps icon URL in between `{}` of `L.icon({});`.

```
iconUrl: 'http://maps.google.com/mapfiles/
arrow.png',
shadowURL: 'http://maps.google.com/mapfiles/
arrowshadow.png',
```

e. Next, provide the settings for **iconSize** and **shadowSize** in pixels. In this example we use the values **[39,34]** representing width and height in pixels for both icon and shadow. *Note: icon options should be separated by commas.*

```
iconSize: [39, 34], // size of the icon
shadowSize: [39, 34], // size of the shadow
```

f. Add the **iconAnchor** option to the **greenIcon** variable. This option identifies the icon pixel that should be located at the marker's latitude/longitude location. The values are relative to the icons top left corner. For this demonstration we have an arrow icon that is centered at the bottom of the image and with an image width of

21 pixels and height of 30 pixels. To find the appropriate **iconAnchor** location, divide the width by two in order to center the arrow point on the marker location. For this example, use the values [11,30]. You might notice this doesn't actually center the arrow point at the correct location. You will need to be careful when you get images from other sources. In this case, the green arrow from Google has additional white (transparent) pixels to the right of the arrow. The actual width of the full image is 39 pixels, but the point is located at 10.25 so we will use **[10,34]** as the value.

```
iconAnchor: [10, 34], //icon pixel which will
                      //correspond to the marker's
                      //location relative to its
                      //top left  corner of image
```

g. Next, add the **shadowAnchor** option. In this example it will use the same values as the **iconAnchor** of **[10, 34]**.

```
shadowAnchor: [10, 34],
```

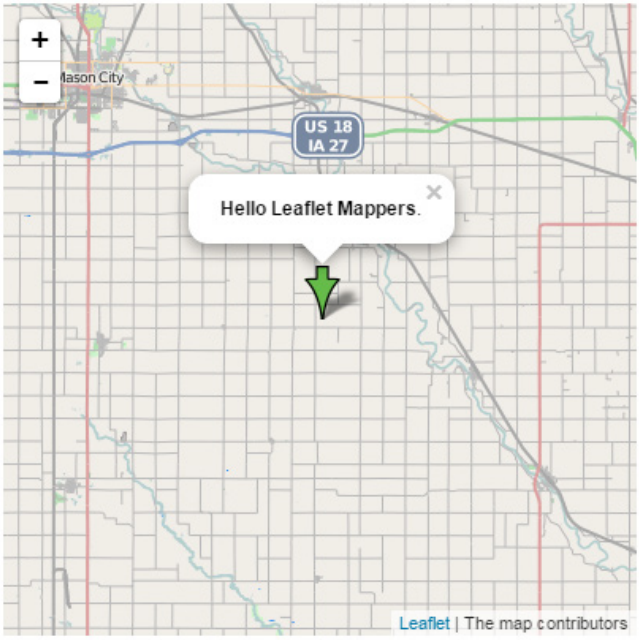
h. Finally, add the **popupAnchor** option that specifies the point from which the pop-up should open relative to the **iconAnchor**. If you use the values **[30,-40]** you will see the pop-up display too far above and to the right of the marker. If you change the values to **[1,-35]** the pop-up will be centered on the green arrow icon. *Note: you can place the definition for the greenIcon variable before or after the map constructor or save it in an external file.*

```
popupAnchor:  [1, -35] //point from which the
                       //popup should open
                       //relative to iconAnchor
```

3. Draw the Marker

a. After defining the icon, you need to add the code that draws the marker. To do this, add the variable **marker1** equal to **L.marker** including the lat/long location and set the option **icon** to the **greenIcon** variable.

```
var marker1 = L.marker([43, -93], {
     icon: greenIcon,
}). addTo(map);
```



b. Finally, bind the **Hello Leaflet Mappers** pop-up to the marker with the following code.

```
marker1.bindPopup("<b>Hello Leaflet
Mappers.</b>");
```

4. Other Icon Options

a. The three other options for icons are **className**, **iconRetinaUrl**, **shadowRetinaUrl**. className can be assigned to both icon and shadow images, and the **iconRetinaUrl** and **shadowRetinaUrl** options allow you to provide URLs for icons to be viewed on Retina screen devices.

b. You can place the definition for the new icon before or after the map constructor or save them in an external file and link to the file

Contact:

Bailey Hanson bahanson@iastate.edu, 515-520-1436 or Professor Christopher J. Seeger, ASLA, GISP cjseeger@iastate.edu, 515-509-0651 for more information about the Geospatial Technology Program. This task sheet and more are available at www.extension.iastate.edu/communities/gis

March 2015 PM2082-15e