Lab: Creating Area Mashups

GEOG 319/658 **Exercise #9** **FALL 2014**

Creating Area Mashups

Due: Monday, November 24

We will be utilizing some of the code from Chapter 14 of the Peterson text. You should have already downloaded that code for the previous exercise.

A. A choropleth map based on flash flood events

Edit the code for Figure 14.23 (use file 14_22_Nebraska_Choropleth_Map in the folder B-Areas) to solve the problem described below.

Assume that you wish to portray the number of flash flood events for counties in Kansas. Ideally, we should standardize these data for choropleth mapping, but for simplicity we will map the raw counts.

Rather than using the XML file for the Nebraska data, these data are stored in the JSON file FlashEventsByCounty.json. You should download this file from Blackboard, open the file in Notepad++, and examine its contents. This is a special type of JSON file (PJSON, Pretty JSON) generated by ArcGIS from shapefiles. In this case, the counties are known as "features" and the "features" consist of "attributes" and "geometry". Within "attributes", we are interested in the "Count_", which specifies the number of flashflood events for a county. Within "geometry", the "rings" consist of points defining the bounds of each polygon.

1) Replace the title in the "head" section with a title more relevant to the flash flood event topic.

2) Replace the line that uses the downloadxml.js file with one that uses the jquery file that we used in the earlier exercise for JSON files. **Don't forget to include the jquery file at the same level as your html file.**

3) Set the zoom level to 7 and center the map on 38,-98.

4) Replace the population data for Nebraska with the code that will access the JSON file for the flash flood events. Here you will use a structure very similar to exercise #7. Rather than AddCityCircles2Map, use a name that reflects you are adding counties to the map (e.g. addCounties).

5) The addCounties function will be structured as follows:
  a) Determine the "length" of "features" within the JSON file.
  b) Put the flashflood event counts in an array.
    For a) and b), you did something very similar in exercise #7.
  c) Call a function called CalculateOpacity that will return the opacities for each county. You will pass
  this function one argument, an array of flashflood event counts.
  d) You will do the following for each county
    i) Get the Latitude and Longitude for each point from the JSON file

```
var pts=json.features[i].geometry.rings[0];
```

ii) Construct a polygon path array

```
var latLngs = new Array();
 for (var k=0; k<pts.length; k++) {
            latLngs[k] = new google.maps.LatLng(pts[k][1],pts[k][0]);
            }
```

iii) Draw the polygon and shade it

Here you will use the code beginning with “var polygon = new google.maps.Polygon” and ending with polygon.setMap(gmap). For “paths:pts”, replace “pts” with “latLngs”. Specify a “strokeColor” that you deem appropriate, a “strokeOpacity” of 1, a strokeWeight” of 1, and a “fillColor” that you deem appropriate. Depending on how you have set up your loop for the counties, you may have to change the subscript for the opacities array.

Note that you will delete any code dealing with handling the XML file.

6. The CalculateOpacity function will consist of the following code. Since we are providing this code, you need to delete similar code that already appears in the program.

```
function CaculateOpacity(attValues) {

            //Determine length of the attribute value array
             var n=attValues.length;

             //Initialize min and max values
             var min=attValues[0];
             var max=attValues[0];

             //Compute min and max values
             for (var i = 0; i < n; i++) {
                        if (attValues[i] < min) { min=attValues[i] }
                        if (attValues[i] > max) { max=attValues[i] }
                        }

             // Find the range of the data
             var range = max-min

             // compute opacity values
             var opacities = new Array(n);
             for (var i = 0; i < n; i++) {
                        opacities[i] = 1-((max - attValues[i]) / range);
             }

             return opacities;
            }
```

B. Creating a heat map of flash flood events

Peterson's book creates a heat map based on tornadoes (see Figure 14.25 of the textbook). Instead we will create a heat map of flash flood events. To accomplish this, we need to convert an ESRI shapefile containing information about the flood events to a Google Fusion Table. To accomplish this, we will use the Shape Escape website as follows:

1. Download the FlashFloodEventsInKansas.zip file from Blackboard and store it with your other files for exercise #9.
2. If you don't already have a Google account, create one (https://support.google.com/accounts/answer/27441?source=gsearch).
3. Go to the Shape Escape website (shpescape.com).
4. Click "shp2fusiontables".
5. Click "Continue".
6. Log in to your Google account.
7. Click on Browse and select the FlashFloodEventsInKansas.zip file.
8. Click "Upload".
9. When processing is complete, click on the Fusion Table ID link.
10. Select "File" and "Share". Change the access from Private to Public by clicking "Change".
11. Now copy the resulting Fusion Table and paste it into the code for file 14_24_Fusion_Table_Heatmap_Tornadoes (in the folder B-Areas).

The resulting code should look something like the following:

```
        // To view this table in a webpage, go to the following link: //https://www.google.com/fusiontables
        /DataSource?docid=1MYh7-//f3ipyGIkKEXfwfyvgg0woPyHNBvRrgJJA0T

          //Create a heat map from the Fusion Table
        layer = new google.maps.FusionTablesLayer('1MYh7-
f3ipyGIkKEXfwfyvgg0woPyHNBvRrgJJA0T', {
          heatmap: true
        });
        layer.setMap(map);
```

You must replace the code '1MYh7-f3ipyGIkKEXfwfyvgg0woPyHNBvRrgJJA0T' with the Fusion Table that you have created.

Also modify the code so that the title reflects that you are working with flash flood events.

Create a folder for exercise 9 on the webhosting service. Load all files related to parts A and B into this folder. Post the resulting URLs to the following Google Worksheet . Hand in a hard copy version of the two parts of the exercise.