Lab 2: City Highs

Thursday, January 26, 2017

In this lab, you will be using Processing to draw a simple bar chart like the one described in Assignment 2.

Basic Requirements:

1. **Dataset**
   You will be given a simple comma delimited file (CSV) called "data.csv". This file will have the following properties:
   a. It has two columns, the first column contains categorical data, and the second column contains quantitative data.
   b. The first row has labels for each column.

2. **Bar and Line Charts**
   In Processing, do the following:
   a. Parse the CSV file and read in the data
   b. The canvas should display the data in the CSV file as a bar chart. The bar chart should display each data item as a bar.
   c. The axes need to be labeled using the labels from the first row of the CSV file. The X-axis should display the categorical data and the Y-axis should display the quantitative data from the CSV file.
   d. Implement mouse hovering. This means that when the user moves the mouse over a bar, the bar should be highlighted. Furthermore, additional text information should be displayed (like a tooltip). The additional information should be the value of the data item from the CSV file and should be displayed as "(Saint Kitts, 81)", showing both x and y values of the data item. For testing intersection, test the geometry of the mouse position with the rectangles.

Tips:

1. Read in the data and then parse it into the format you want.
2. Most of you already know Java. Remember that Processing is built upon Java and you can use all the readers that you already know.
3. However, Processing offers some simpler methods like createInput(), createReader(), loadStrings(), loadTable() (Note that loadTable() does not work when you convert your application to JavaScript.).
4. Below is an example of loadStrings(). You are welcome to use other advanced data structures and methods.

```
String path = "data.csv";                          // define the path

void loadStrings(){
   String[] lines = loadStrings(path);             // load the file as each line is a string
   String[] firstLine = split(lines[0], ",");      // the first line is the title
   xName = firstLine[0];                           // the name of the x-axis
   yName = firstLine[1];                           // the name of the y-axis
   names = new String [lines.length - 1];          // the rest is the record
                                                   // for the dimension has the categorical data
   values = new int[lines.length - 1];             // for the dimension has the numeric data

   for(int i = 1; i < lines.length; i++){          // traverse each line
      String[] row = split(lines[i], ",");         // split the line using ","
      names[i - 1] = row[0];
      values [i - 1] = (int) parseFloat(row[1]);   // the second one should be the numeric data
                                                   // parseFloat is to convert the string to the float
                                                   // Or you can use Integer.parseInt. It is a method in
                                                   Java to convert string to integers.
   }
}
print(xName + "," + yName);                        //check
print(names);
print(values);
```

Notes:

1. The system should be resizable and you may not assume a fixed dimension for your canvas. In other words, if the canvas size becomes larger, your visualization should also become larger. If the canvas size becomes smaller, your visualization should not be "cut off".
    a. To add resizing to your Processing code, add the following line in your setup function:

        surface.setResizable(true);

    b. Note that you need to remove this line is you wish to post the program online. This line of code is not compatible with Processing.js and will prevent your code from running in a browser.
    c. Pay attention to the size of the spacing (e.g., spacing between the bars and circles). The spacing should not be a static number (e.g., hard coded to 10 pixels), but should be dynamically determined based on the canvas size and the number of data elements.
    d. If done right, I should be able to load another data file of any size into your application.
2. Do not assume a fixed range for neither the X- nor Y-axis. For instance, do not assume that the values in the Y-axis are always between 0 and 100. Determine the lower and upper bound based on what you read from the data and scale the Y-axis accordingly. Similarly, do not assume that the number of data items in the X-axis is always below a certain number. Your visualization should adapt to the data that it is given and display the data in the most appropriate manner. In other words, you can't assume there is a hard-coded range of the dataset. However, you may assume that the data will not be too big.

3. Your code should be modular. I recommend have a class for the bar chart and one for the line. Feel free to add as many classes as you see fit.

Submission:
Submit your work via **Blackboard** by at the end of class or by midnight. Use the naming convention: "FirstnameLastname_l2.zip" (e.g., AlvittaOttley_l2.zip). The labs are not graded but submission counts toward your participation grade.