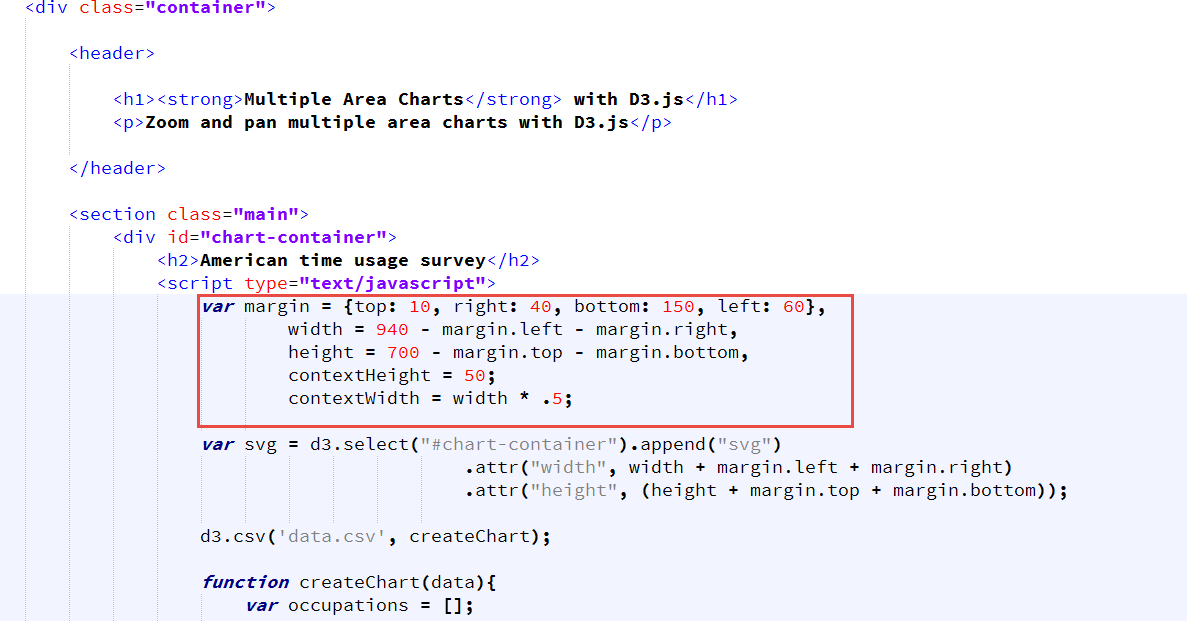
***Tutorial for Multiple Area Charts with D3.js using American time usage survey data set***

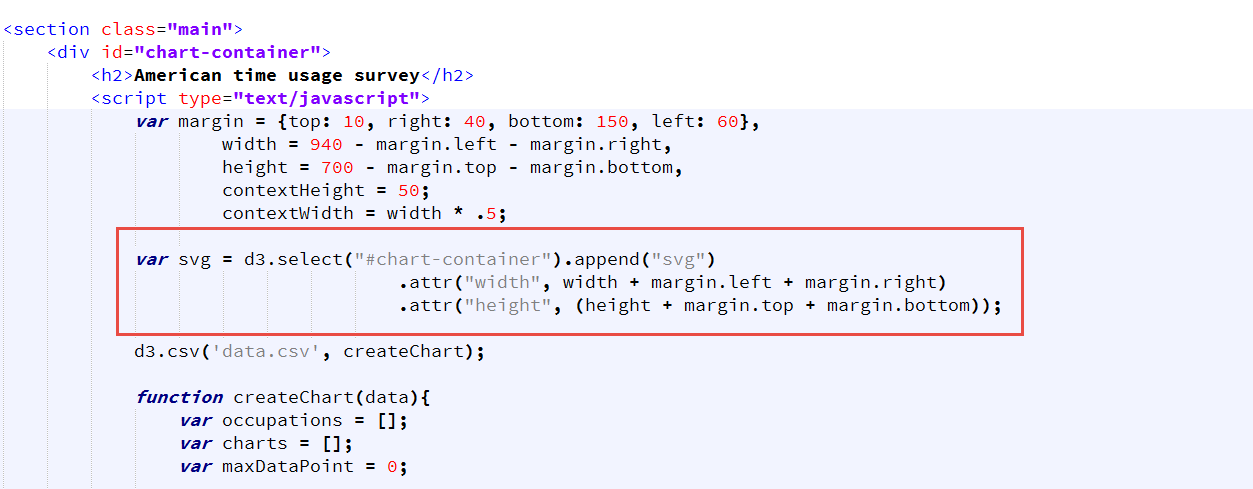
Create a HTML page and import D3 js library.



Next we will set up some variables to size and position our chart using in line JavaScript.



Next we will want to add an SVG tag to the page. D3’s select function conveniently uses CSS style string selectors.



Next step is to load the CSV file using following statement,

d3.csv('data.csv', createChart);

After the CSV file loads, loop through the first row to grab each occupation and store it in an array.

for (var prop in data[0]) {

if (data[0].hasOwnProperty(prop)) {

if (prop != 'Time') {

**occupations.push(prop);**

}

}

};

Also loop through the entire data set to ensure that every value is a number while converting the time value to a JavaScript Date object. Also want to find the maximum data point that can be used to set y-axis scale.

data.forEach(function(d) {

for (var prop in d) {

if (d.hasOwnProperty(prop)) {

d[prop] = parseFloat(d[prop]);

if (d[prop] > maxDataPoint) {

**maxDataPoint = d[prop];**

}

}

}

// D3 needs a date object, let's convert it just one time

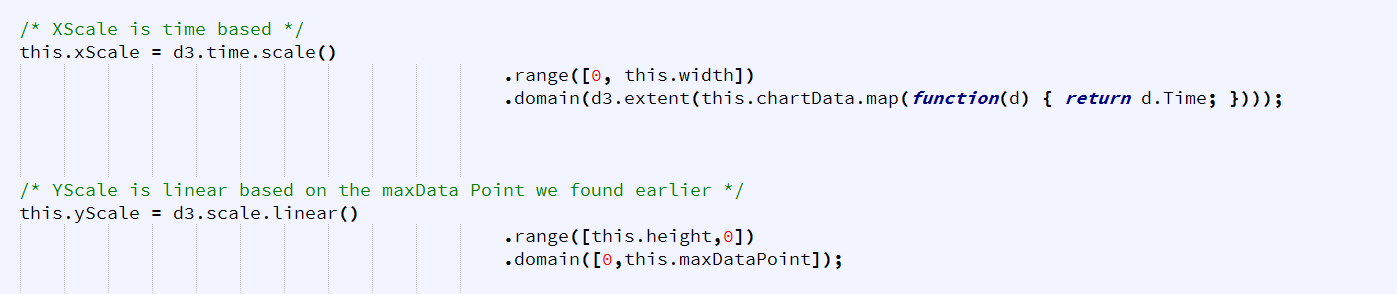
**d.Time = new Date(0,0,1,d.Time);**

});

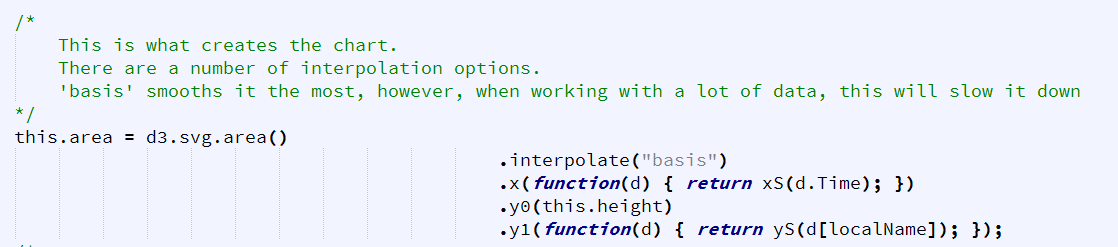
Now create the context brush. This is the tool that will allow users to zoom and pan the charts.



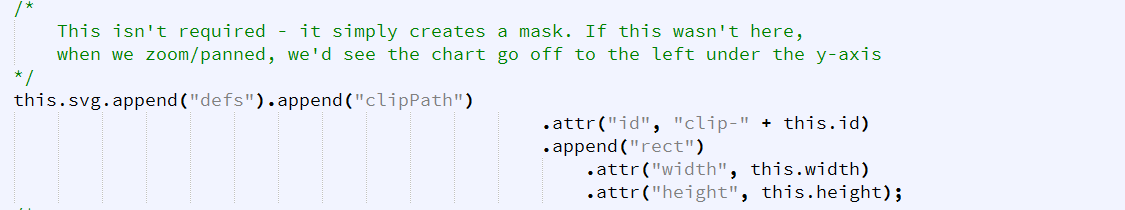
Next we need to create the scales for x and y axis. Scales can be time-based or quantitative (linear, logarithmic, or power). Since our x-axis is going to represent the time, we will use a time-based scale. The y-axis represents the time usage in percentage, so we'll use a linear scale.



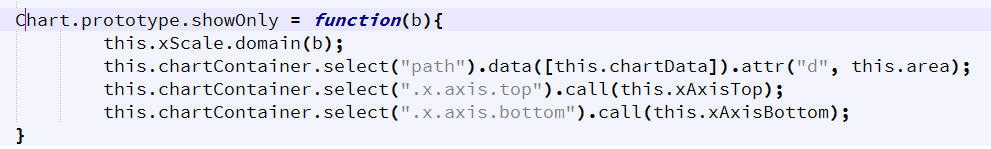
The call to area method creates the chart. There are a number of interpolation options we can use. 'basis' is the smoothest, however, when working with a lot of data it will also be the slowest.



We will create a definition with which to display the chart. In this example, it acts as a container for the chart. Without it, we would see the chart go off the left and under the y-axis when zooming and panning.



The function **showOnly** is called when the chart is panned or zoomed. It's passed a start and end date. The chart will need to be redrawn to match these date limits. Therefore, we'll need to reset the xScale domain, reset the area path and update the axis.



Finally, give a color to each occupation using style.css file

