Startup when creating a new graph

Author:
Alex Peschel
alepes-8@student.ltu.se

March 8, 2021

Contents

1 Create a new graph (read first)

This document should give you a good push on the way to create a new graph. However if you wish to add a new type of graph you will need to do some extra coding, but this should still be able to help you some navigation though the code.

This is an overview of how to easily make a graph in the system and where to write the code. But I may have missed something even though everything seems to work.

Names and other variables may wanna be change compared to the references i make, pictures I have shown. But do the change while careful, some does not need to be changed. On top of that, note that if you have the time you may wanna create some statements so that the a couple functions within file three can be used again and again(Instead of writing everything from scratch for every graph.). The only difference is the data. Up to your own discretion and if you have time

2 File one

2.1 Directory:

"Digital-Vinter/app/views/graph.ejs" or "rcm-sommar-2019/app/views/graph.ejs" depending on the download and place you check.

2.2 step 1

As a start you should create one of this.

```
<label id="prov15" class="container">Accident correlation
  <input type="checkbox" checked="checked" id="accident_correlation">
  <span class="checkmark"></span>
  <a id="jumper" href="#type15">Hoppa</a>
</label>
```

Figure 1

A couple of changes need to be done before proceeding. Change the input id(accident_correlation) to something else, the Accident correlation at the label to the graph's name.Then the most important part is to change the href and the label id to something which does not exist at the moment. But try to keep the name similar so it's easier to know where to look. This lets the graph alternative show the specific requested graph.

2.3 step 2

You would have two alternatives, but at least one of the following should be written.

```
<div id="type5">
  <button class="hidebut"  id="Canvashidebutton" type="button" onclick="hideelements('type5','current_temp')">X</button>
  <canvas id="myChart5"></canvas>
</div>
```

```
137   </div>
138   <div id="type2">
139     <button class="hidebut" id="Canvashidebutton" type="button" onclick="hideelements('type2','Road_temp')">X</button>
140     <button class="hidebut" id="CanvasDownloadButton" type="button" onclick="CSVdownload(datagraftimestamp,data3graf3)">Ladda ner</button>
141     <canvas id="myChart2"></canvas>
142   </div>
```

Figure 2

Which one depends on what application you would like to apply to the graph. Do you only wish to be able to delete the graph then the first one is good enough, but if you would like to download the graph as well the second one is better. Similar to step 1, you will need to change the div id to the same as the href you decided over. Same with the button. Most important is to give the canvas id a proparet name to be able to keep track of it and use shortly after.

2.4 Step 3

Shortly after the step two in the code, you will be able to find this.

```
//If new chart canvas is added, add chart ID into the array. Used in RunGraphs to clear.
//Possible change is to maybe go through the page with JavaScript to find all the canvasid instead of manually inserting
var canvaschartidarray = ['myChart1','myChart2','myChart3','myChart4','myChart5','myChart6','myChart7','myChart8','myChart9','myChart10','myChart11','myChart12','myChartFricti
```

Figure 3

The thing you wrote as your canvas id will be written into this array so that it will be able to print out in the rungraph file.

2.5 Part 4

Depending on which situation you call on the graphs, county or stations, you can adjust whether the buttons in step 2 and the graph alternatives in step 1 will show.

```
243
244  //Hide the elements at start, We show later after graph load
245  hide1("hidebut");
246  hide1("province");
247  hide1("frictionbuttoncanvas");
248  hide1("frictionbuttoncanvascurrent");
249  if(stationsarrayid.length<=0){
250    for(var i =4;i<=12;i++){
251      hidebyid("prov"+i);
252    }
253    for(var y=1;y<9;y++){
254      hidebyid("type"+y);
255    }
256    hidebyid("type12");
257  }
258  if(chosenCounties.length<=0){
259    hidebyid("prov1");
260    hidebyid("prov2");
261    hidebyid("prov3");
262    hidebyid("type9");
263    hidebyid("type10");
264    hidebyid("type11");
265  }
266  if(boundsrect.length<=0 && boundscircle <=0){
267    hidebyid("prov13");
268    hidebyid("type13");
269  }
270  if(frictionarrayid.length<=0){
271    hidebyid("prov14");
272    hidebyid("type14");
273  }
```

Figure 4

3 File two(optional perhaps)

3.1 Directory

digital-vinter/app/public/javascripts/graph/Options.js

3.2 Step 1

This is unknown if the change makes any difference or not. During one instance when creating the bubble graph for stations code were added. However, in the instance when creating a county bubble graph for accidents no change in this file were needed.

4 File three

This file will probably be the one most worked with if any changes to the graph is needed. How they look, what they show, and how they show it.

4.1 Directory

Digital-Vinter/app/public/javascripts/graph/graphs.js

4.2 Bar Graph

4.2.1 Step 1

Start creating a place where the values will be sent into

```
//current data air temp
/**
 * Collects data and send to generate function
 * @param {*} weatherdata This is the current air temp sent in
 * @param {*} station_name station name
 */
function databarchartcurrent(weatherdata,station_name){
        var typeofgraph = "current_air";
        var stationame = station_name;
        var datatempvar= weatherdata[0].air_temperature;
        //var datatempvar= weatherdata.air_temperature;
        generatedataforbar(typeofgraph,datatempvar,stationame);
}
```

Figure 5

The weatherdata[0].air_temperature can be written in many other ways. But what I mean is that you will take out the air temperature which is right now(position 0) from the sent in weather data. This way the most current data is placed in the graph. change the functions name.

4.2.2 Step 2

Then as seen in step 1 it calls on the generatedataforbar.

```
var colornamelist = [];
/**
 * Generats variables for the stations used as data for the graphs.
 * @param {*} value This specify which graph to generate data for like "
 * @param {*} datagraf This is the data for the graph
 * @param {*} stationame The station or province name
 */
function generatedata(value, datagraf, stationame){
    if(stationnamelist.includes(stationame)){
            var dataFirst = {
            label: stationame,
```

Figure 6

This saves the station name, color and the data for the graph in the right location. So make sure that the type of graph in the if statement is the same as the one in step one function.

4.2.3 Step 3

The date where it is pushed in to step 2 will be used in a similar function as below.

Figure 7

If you wanna change the name of the graph just change what is written in "text". Also change the myChart10 and the chart11(chart11 should be changed in all the places needed in the picture).

4.2.4 Step 4

Go to the necessary getweather data place and rungraph so that what you have done gets data but also start to run.

4.3 Linear Graph

Step 1 Similar to bar one, you wish to send you data into a function manage the drawing. Though, to manage multiple data points you will have a loop.

Figure 8

4.3.1 Step 2

Under the function generatedata

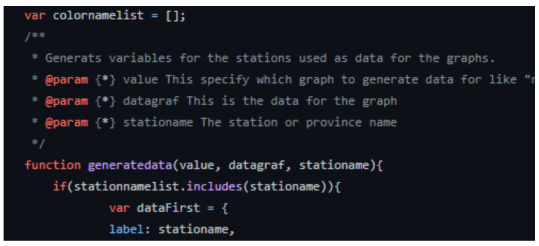

Figure 9

add the functions name in

```
        if(value=="roadtemp"){
                data3graf3.push(dataFirst);
        }
        if(value=="airtemp"){
                datagrafair.push(dataFirst);
        }
        if(value=="humidity"){
                datagrafhum.push(dataFirst);
        }
        if(value=="windspeed"){
                datagrafwindspeed.push(dataFirst);
        }
        if (value=="avgprovairtemp"){
                datagraftempprov.push(dataFirst);
        }
}
```

Figure 10

So that the data is pushed to the right placed and later handled in the best way. But you should not need to change any setting in the generatedata function

4.3.2 Step 3

The following picture represent step 3 for a linear chart.

```
var lineChart1 = null;
//function to create road_temp graph
/**
 * This function will generate road temp graph with the data in arrays generated from generatefuctions
 */
function roadtemp(){
if(lineChart1!=null){
lineChart1.destroy();
}

var speedCanvas = document.getElementById("myChart2");
Chart.defaults.global.defaultFontFamily = "Lato";
Chart.defaults.global.defaultFontSize = 18;



var speedData = {
  labels: datagraftimestamp,
  datasets: data3graf3
};

var chartOptions = {
    scales: {
        xAxes: [{
            ticks: {
                fontSize: 15
            }
        }]
    },
    title:{
        display:true,
        text: "Vägtemperatur",
  legend: {
    display: true,
    position: 'top',
    labels: {
      boxWidth: 80,
      fontColor: 'black'
    }
}
  }
};

lineChart1 = new Chart(speedCanvas, {
  type: 'line',
  data: speedData,
  options: chartOptions
});
lineChart1.update();
}
```

Figure 11

Under the speedData the "label" is the x axes and the "datasets" is the y axes. The chart options is the extra setting of the graph, for example title. most options is already determined in 4.3.1. But most important is the part where you create a new graph is the last lines which starts with;

Figure 12

4.3.3 Step 4

Make sure most of the values are changed, function name is changed and that your values are written where they should throughout the file. Example for this is on the top and on the bottom in the code.

4.3.4 Step 5

Write in the rungraph and the weather data files so that the respective data is sent in, but also that the functions start up.

4.4 Bubble Graph

4.4.1 Step 1

To make sure that the sent in data is handled in the right way you will have a similar function as below.

Figure 13

Change the data taken from the input weather data, otherwise it should be pretty similar. The input needed is an x, y and r axes. The r axes now allows the change of data points on a graph.

4.4.2 Step 2

Create the code which will then create the graph with the data take out in step 1. This one way to do it, still holding options, datasets and type.

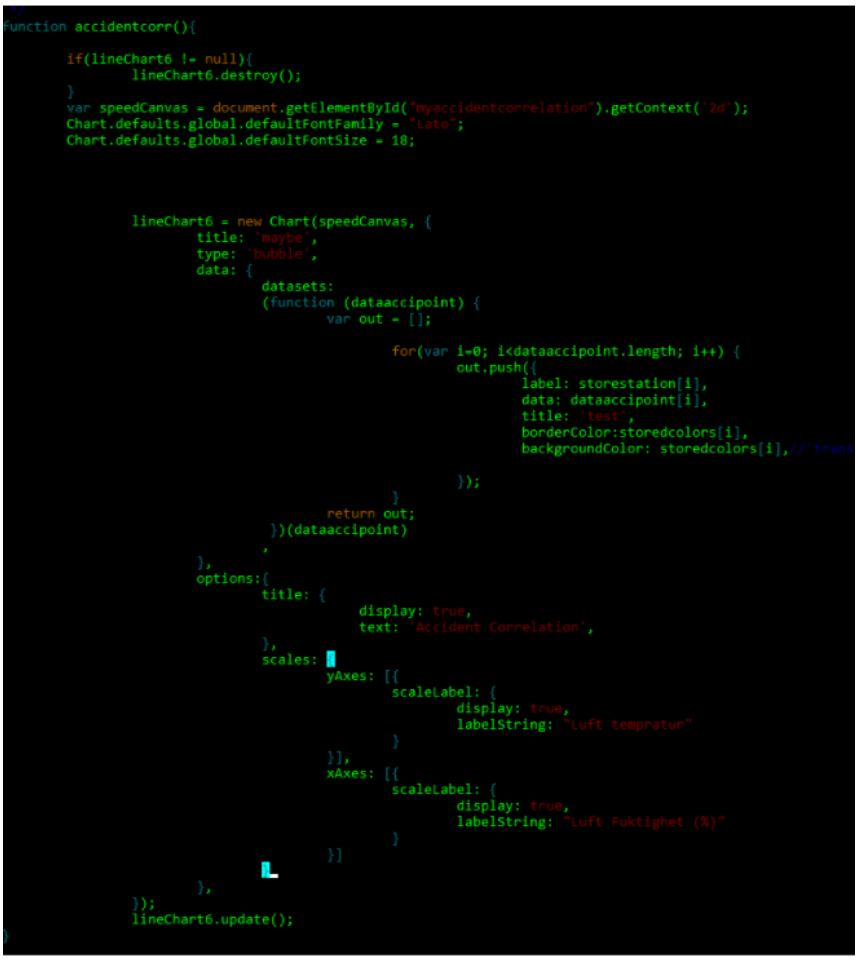

Figure 14

Where

Figure 15

is defined just before. take away small titles here and there where it says test, otherwise this would provide the needed information to create a bubble graph where you can customize it.

4.4.3 Step 3

Check that the values are where they should, check that the function name is changed and you changed everything where it is needed.

4.4.4 Step 4

Go to rungraph and the needed file where you get the data. Like getweatherdata or getaccidentdata.

5 File four

5.1 Directory

Digital-Vinter/app/public/javascripts/graph/RunGraphs.js

5.2 Step 1

Make sure to add the functions you created in file three to the rungraph file. For bar you may not need to do it for both but make sure to add the needed, so check similar functions and add it and see if it works.

6 File five.

I will not write a specific directory, but I can give multiple.

- Digital-Vinter/app/public/javascripts/graph/getAccidentData.j
- Digital-Vinter/app/public/javascripts/graph/getWeatherData.js
- Digital-Vinter/app/public/javascripts/graph/getFrictionData.js

You will with high possibility ´though at least one of this, or a whole new file, get the necessary data to show the wanted values on the table. I will not give example due to it being different depending on what data and what graph. But you should know where to look at least.