import { Injectable } from '@angular/core';

declare var google: any;

@Injectable()

export class ChartService {

constructor() {

google.charts.load('current', {packages: ['corechart', 'bar', 'sankey']});

}

drawStackedChart(element, chartData, auxData, hAxisTitle, vAxisTitle, title) {

const dataArray = this.formDataArray(chartData, auxData);

let data = google.visualization.arrayToDataTable(dataArray);

let options = {

title: title,

//height: 500,

chartArea: {width: '70%'},

isStacked: true,

hAxis: {

title: hAxisTitle,

minValue: 0,

format: 'percent'

},

vAxis: {

title: vAxisTitle

},

legend: {

position: 'top'

},

animation: {

startup: true,

duration: 1200,

easing: 'in'

}

};

let chart = new google.visualization.BarChart(element);

chart.draw(data, options);

}

private formDataArray(chartData, auxData) {

let dataArray = [];

for (let i = 0; i <= chartData.bars.length; i++) {

dataArray[i] = [];

for (let j = 0; j <= chartData.criteria.length; j++) {

if (i == 0) {

if(j==0) {

dataArray[i][j] = '';

}

else {

dataArray[i][2\*j-1] = chartData.criteria[j-1];

dataArray[i][2\*j] = {type: 'string', role: 'tooltip'};

}

}

else if (j == 0)

dataArray[i][j] = chartData.bars[i-1];

else {

dataArray[i][2\*j-1] = chartData.prioritiesMatrix[i-1][j-1];

dataArray[i][2\*j] = chartData.bars[i-1] + ': ' + (auxData[i-1][chartData.criteria.length]\*100).toFixed(2) + '% | ' +

chartData.criteria[j-1] + ': ' + (chartData.prioritiesMatrix[i-1][j-1]\*100).toFixed(2) + '%';

}

}

}

return dataArray;

}

drawNodesChart (element, chartData) {

const chart = new google.visualization.Sankey(element);

const dataArray = this.formNodesDataArray(chartData);

let data = new google.visualization.DataTable();

data.addColumn('string', 'From');

data.addColumn('string', 'To');

data.addColumn('number', 'Valor');

data.addRows(dataArray);

// Set chart options

const options = {

sankey: {

height: 900,

node: {

label: {

fontName: 'Arial',

fontSize: 18,

colors: ['#DFCA1C', '#1E4DC3', '#880700', '#26D717', '#cab2d6', '#ffff99']

},

},

link: {

colorMode: 'gradient',

},

iterations: 32

},

};

chart.draw(data, options);

}

formNodesDataArray(chartData) {

let dataArray = [];

let k = 0;

for (let i = 0; i < chartData.criteria.length; i++, k++) {

dataArray[k] = [chartData.goal, chartData.criteria[i], chartData.criteriaPriorities[i]];

}

for (let i = 0; i < chartData.criteria.length; i++) {

for (let j = 0; j < chartData.alternatives.length; j++, k++) {

dataArray[k] = [chartData.criteria[i], chartData.alternatives[j], chartData.prioritiesMatrix[j][i]]

}

}

return dataArray;

}

drawSimpleChart(element, chartData, hAxisTitle, vAxisTitle) {

const dataArray = this.formSimpleDataArray(chartData);

let dataTable = new google.visualization.DataTable();

dataTable.addColumn('string', 'Criterio');

dataTable.addColumn('number', 'Prioridad');

dataTable.addColumn({type: 'string', role: 'tooltip'});

dataTable.addColumn({type: 'string', role: 'style'});

dataTable.addRows(dataArray);

const chart = new google.visualization.BarChart(element);

const options = {

chartArea: {width: '70%'},

hAxis: {

title: hAxisTitle,

minValue: 0,

format: 'percent'

},

vAxis: {

title: vAxisTitle

},

legend: {

position: 'none'

},

animation: {

startup: true,

duration: 1200,

easing: 'in'

}

};

chart.draw(dataTable, options);

}

formSimpleDataArray(chartData) {

let dataArray = [];

const colors = ['#3366cc', '#dc3912', '#ff9900', '#109618', '#990099', '#0099c6', '#dd4477']

for (let i = 0; i < chartData.vertical.length; i++) {

dataArray[i] = []

dataArray[i][0] = chartData.vertical[i];

dataArray[i][1] = chartData.horizontal[i];

dataArray[i][2] = chartData.vertical[i] + ': ' + (chartData.horizontal[i]\*100).toFixed(2) + '%';

dataArray[i][3] = colors[i]

}

return dataArray;

}

}