

**Dataset used:** Şekline ve cinsiyete göre intiharlar, 2000-2022

**Explanation:** This code generates SVG-based bar charts for visualizing TUIK data. It comprises several functions, starting with the entry point plotSvg(data) that accepts data in a specific format as read from the excel file. It takes the necessary parts from the data array, sets up the SVG element, adds a title, generates axes with customizable labels and tick marks, and creates bars with support for dual bars per data point. The bars can be customized with colors, and the code includes provisions for adding vertical text labels using the writing-mode attribute. Constants allow for flexible dimension and spacing adjustments as needed.

**How to use:** Open the `plot.html` file in a browser. Click “Browse” button and select the given xls file.



**Methodology:** There are two files, plot.html and plot.js. The “XLSX” library is also used for reading an excel file. Due to web-browsers security limitations, the file cannot be read directly and so a file input was created. The user has to click the button and choose the correct file to see the visualization.

The `init` and `parseXlsx` functions provide the necessary functionality to read the excel workbook.

function init() {

    const fileInput = document.getElementById('fileInput');

    fileInput.addEventListener('change', (*event*) => {

        const file = event.target.files[0];

        const reader = new FileReader();

        reader.addEventListener('load', (*event*) => {

            const data = parseXlsx(event.target.result);

            plotSvg(data);

        });

        reader.readAsBinaryString(file);

    });

}

function parseXlsx(*fileContent*) {

    const workbook = XLSX.read(fileContent, { type: "binary" });

    const sheetName = workbook.SheetNames[0];

    const sheet = workbook.Sheets[sheetName];

    const data = XLSX.utils.sheet\_to\_json(sheet, { header: 1 });

    return data;

}

After uploading the file, the function `plotSvg` takes in the content of the file and performs the necessary operations to generate an SVG.

const dataToPlot = prepareData(data);

    const maxDataValue = Math.max(...dataToPlot.map(*item* => Math.max(item.value1, item.value2)));

*// Constants for offsets and dimensions*

    const svgWidth = 1600;

    const svgHeight = 800;

    const barPadding = 1;

    const margin = 55;

    const xAxisStartY = svgHeight - margin;

    const yAxisStartX = margin;

    const xAxisLabelYOffset = 15;

    const yAxisLabelXOffset = -10;

    const numYAxisTicks = 5;

    const barSpacing = 15;

    const numBars = dataToPlot.length \* 2;

    const totalSpacing = (numBars - 1) \* barSpacing;

    const barWidth = (svgWidth - 2 \* margin - totalSpacing/2) / numBars;

    const svg = createSvg();

    createTitle();

    createAxes();

    createBars();

Here, the dimensions and offsets are defined. The svgWidth and svgHeight define the total size of the plot. `barPadding` is the distance between two subsequent bars. `xAxisStartY` is the starting point of x-axis, and yAxisStartX is the beginning point of y-axis. `xAxisLabelYOffset` and `yAxisLabelXOffset` are offsets for the labels. `numYaxisTicks` is the number of ticks for the y-axis. `barSpacing` is the distance between every data point. `barWidth` defines the width of a bar based on the number of bars (found from the file). After defining these, the following functions are called in order.

1. `createSvg`

    function createSvg() {

        let svg = document.createElementNS("http://www.w3.org/2000/svg", "svg");

        svg.setAttribute("width", svgWidth);

        svg.setAttribute("height", svgHeight);

        document.body.appendChild(svg);

        return svg;

    }

Simply creates the main SVG where the plot will be drawn in.

1. `createTitle`

    function createTitle() {

        const chartTitle = data[1][0];

        const titleText = document.createElementNS("http://www.w3.org/2000/svg", "text");

        titleText.setAttribute("x", svgWidth / 2);

        titleText.setAttribute("y", margin - xAxisLabelYOffset);

        titleText.setAttribute("text-anchor", "middle");

        titleText.textContent = chartTitle;

        svg.appendChild(titleText);

    }

Creates a text type element for the title of the plot.

1. `createAxes`

This function creates two **line**s for the axes. It then draws ticks as lines in set intervals on top of the axes. It also creates **text** objects for indicating the values, as well as labels for the axes

1. `createBars`

 function createBars() {

        dataToPlot.forEach((*item*, *index*) => {

            const x = yAxisStartX + index \* (barWidth \* 2 + barSpacing);

*// Create the first bar*

            const bar1 = document.createElementNS("http://www.w3.org/2000/svg", "rect");

            const height1 = (item.value1 / maxDataValue) \* (svgHeight - 2 \* margin);

            bar1.setAttribute("x", x);

            bar1.setAttribute("y", xAxisStartY - height1);

            bar1.setAttribute("width", barWidth - barPadding);

            bar1.setAttribute("height", height1);

            bar1.setAttribute("fill", "blue");

            svg.appendChild(bar1);

            const valueLabel1 = document.createElementNS("http://www.w3.org/2000/svg", "text");

            valueLabel1.setAttribute("x", x + (barWidth / 2));

            valueLabel1.setAttribute("y",  xAxisStartY - margin);

            valueLabel1.setAttribute("writing-mode", "tb-rl");

            valueLabel1.setAttribute("text-anchor", "middle");

            valueLabel1.setAttribute("fill", "white");

            valueLabel1.textContent = item.value1;

            svg.appendChild(valueLabel1);

... same for the second bar

            const label = document.createElementNS("http://www.w3.org/2000/svg", "text");

            label.setAttribute("x", x + barWidth / 2 + barSpacing);

            label.setAttribute("y", xAxisStartY + xAxisLabelYOffset \* 2);

            label.setAttribute("text-anchor", "middle");

            label.setAttribute("fill", "black");

            label.textContent = item.label;

            svg.appendChild(label);

            const label1 = document.createElementNS("http://www.w3.org/2000/svg", "text");

            label1.setAttribute("x", x + barWidth / 2);

            label1.setAttribute("y", xAxisStartY + xAxisLabelYOffset);

            label1.setAttribute("text-anchor", "middle");

            label1.textContent = "M";

            svg.appendChild(label1);

...same for the second label

This function draws the bars as **rect** objects. Essentially, it creates two rectangles at every step, one for males and one for females. with starting point on the x axis with regular intervals, and a height that’s proportional with their respective values. It also writes the values on the bars with `valueLabel` variable. It makes use of `writing-mode` and `fill` attributes to rotate the text and make it white, respectively.