V1 Data Visuals!

"Data is a precious thing and will last longer than systems themselves."

- Tim Berners-Lee

Introduction

As the first module within the specialized Data Visualization curriculum, this module serves as an introduction to creating visualizations using JavaScript. The students will get a good idea of what's expected of them, as well as what they can expect of the class throughout the rest of the year.

Objectives

1. Students will be introduced to color schemes.
2. Students will be introduced to a few templates for Charts.js.
3. Students will use the necessary software needed for making visualizations in Chart.js.
4. Students will have the chance to create their own visualizations using Chart.js.

Agenda

1. Crazy Coolors
2. Charts, Charts, Charts: Charts.js Activities

Facilitation Notes

1. Make sure to help the students use Chart.js, if needed. Although many students have taken (or are in) CSI, giving them a helping hand with JavaScript is essential.
2. Try to share your computer screen (either on a projector or through Zoom) to help students follow your setup of Sublime and Chart.js easily.

Activities

Crazy Coolors (10 minutes)

Purpose:

Teach the importance of beautiful colors!

Materials:

The Importance of Colors Slideshow
Coolors Website

Directions:

1. Go through the slideshow to show how bad colors and good colors can make a difference.

2. Slideshow
3. Go to Coolors and show the class how to generate a color template and copy the hexadecimals to different platforms.

Intro to Chart.js (5 minutes)

Purpose:

Introduce students to Chart.js and basic templates.

Materials:

Chart.js Website

IMPACT's Chart.js templates

Directions:

1. Explain exactly what Charts.js can do and give a demonstration of switching up data and showing off different types of graphs made with Charts.js.
2. Explain how Chart.js is a JavaScript library that can be used to make seamless visualizations and graphs.
3. Go to the website linked above and show a few examples of visualizations created using Chart.js (such as a stacked bar chart or line chart). Try clicking the "Randomize Data" button to see how the visualizations change.
4. Discuss what features of these visualizations make them stand out. Important features include the transitions after "Randomize Data" is clicked and the information that appears when you hover over the charts.
5. Tell the students how these visualizations are interactive because of JavaScript, and briefly show them templates that can be used to make the visualizations.

Setting Up Notepad++ and Chart.js (10 minutes)

Purpose:

Set up a programming environment that the students can use to make visualizations through Chart.js.

Materials:

Notepad++ Homepage

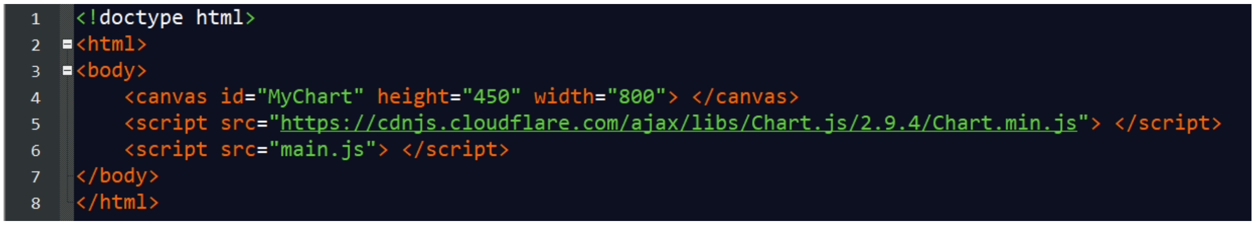
Chart.js Documentation

Directions:

1. Go to the Notepad++ homepage and ask students to download the version of the text editor that matches their computers' operating systems (Windows, Mac, or Linux).
    a. Many of them should have this editor from CSI, but make sure that everyone has some sort of editor that they know how to use.
2. Walk through the installation of the Notepad++ editor (if needed) and answer any relevant questions. If the students have other editors they prefer, that's completely okay!
3. Next, walk through how we can integrate Chart.js with HTML:
    a. Create an HTML file and save it to your computer.
    b. Write the lines of HTML below and explain what these lines mean:
        i. Everyone should be familiar with tags like **<html>** and **<body>**, so you don't need to explain this.
        ii. Explain how you need a **<canvas>** for the placement and dimensions of the visualization.

iii. Describe how **<script>** is used for incorporating JavaScript, and **src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.min.js"** is used to import the Chart.js library.

iv. The other **<script>** tag references a JavaScript file we'll use for making the graphs.



```html
<!doctype html>
<html>
<body>
    <canvas id="MyChart" height="450" width="800"> </canvas>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.min.js"> </script>
    <script src="main.js"> </script>
</body>
</html>
```

4. Ask the students to make a **main.js** file in the same folder as their HTML file.

Making a Line Graph in Chart.js (5 minutes)

Purpose:

Practice making basic charts using JavaScript and help students learn how to use Chart.js.

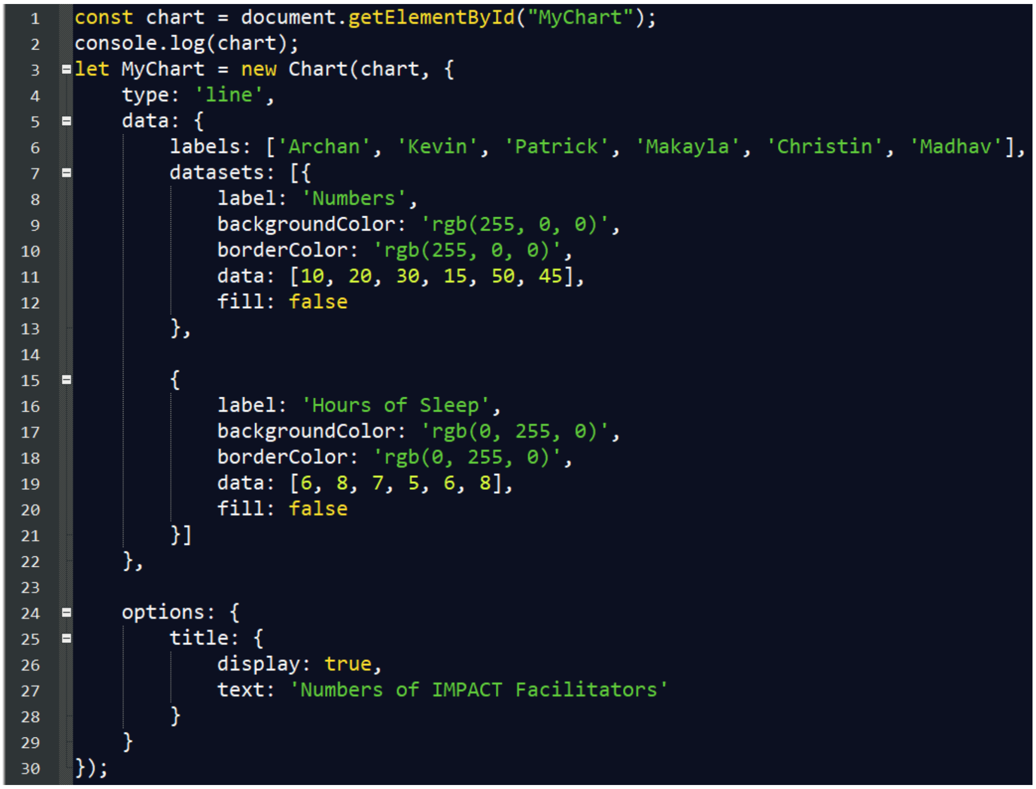
Materials:

Notepad++

Sample HTML and JavaScript Files (in Folder)

Directions:

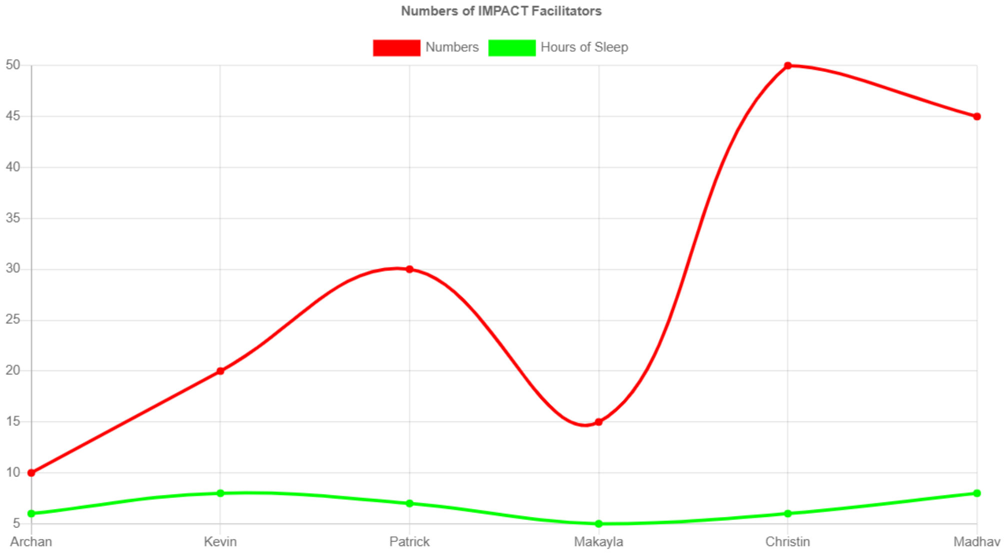
1. Walk through what each line of the **main.js file** should look like:
    a. The line with **document.getElementById()** accesses the canvas object in the HTML file.
    b. The **let MyChart = new Chart()** statement is important because it defines how our line chart will look.
2. Explain how there are 3 important components to our line graph: **type**, **data**, and **options**.
    a. Type is useful for what kind of chart you're making - we're using **'line'** because we want to make a line graph.
    b. Data contains a list of **labels**, which is basically the labels of the x-axis, and a list called **datasets**. Each element of the **dataset** list is a dictionary and a separate line on the line graph.
    c. Options are useful for configuring how the visualization will look like. Here's some documentation about how visuals can be configured.



```
const chart = document.getElementById("MyChart");
console.log(chart);
let MyChart = new Chart(chart, {
    type: 'line',
    data: {
        labels: ['Archan', 'Kevin', 'Patrick', 'Makayla', 'Christin', 'Madhav'],
        datasets: [{
            label: 'Numbers',
            backgroundColor: 'rgb(255, 0, 0)',
            borderColor: 'rgb(255, 0, 0)',
            data: [10, 20, 30, 15, 50, 45],
            fill: false
        },

        {
            label: 'Hours of Sleep',
            backgroundColor: 'rgb(0, 255, 0)',
            borderColor: 'rgb(0, 255, 0)',
            data: [6, 8, 7, 5, 6, 8],
            fill: false
        }]
    },

    options: {
        title: {
            display: true,
            text: 'Numbers of IMPACT Facilitators'
        }
    }
});
```



General Notes:

- Make sure to provide the students with the code, if needed, so they have an idea of how this works.
- To add more lines to the graph, you need to add more dictionaries to the **datasets** list. Inform students about this.
- Many of them should be familiar with this, but to run the code through Notepad++, open the HTML file, go to **Run** in the toolbar, and select a browser to run the graph on.

Templates and Practice (10 minutes)

Purpose:

Introduce students to a few Chart.js templates that they can use in their SLX projects and practice making Chart.js visualizations!

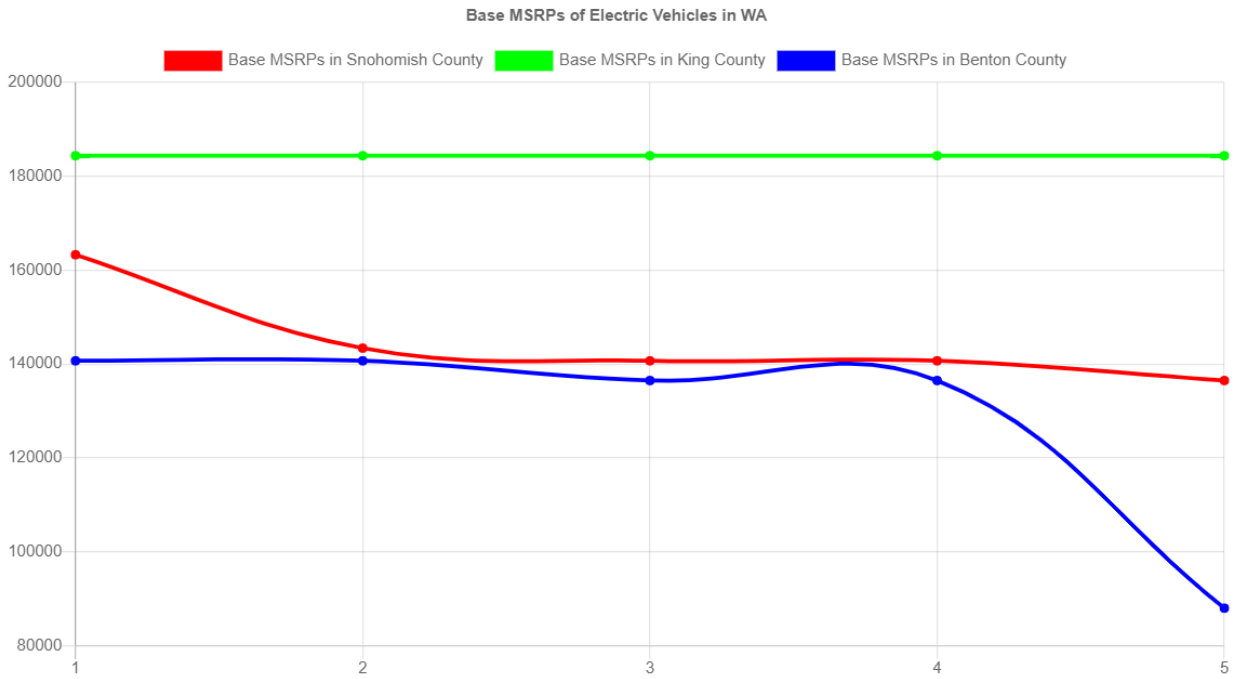
Materials:

Chart.js Templates and Documentation

Electric Vehicles in Washington Dataset (Open-Source) - Google Sheets Link

Directions:

1. Show the students that Chart.js can be used to make a lot of different graphs, even stacked bar charts.
2. Go to the website with the templates and tell the students that these templates are great resources to use for their SLX project. Also, make sure to mention that Chart.js has a detailed documentation of its features. **Make sure to mention how students can use these resources for the SLX project.**
3. Then, facilitate an activity to help the students practice their skills.
   a. Show the students the Electric Vehicles Dataset.
   b. **<u>Exercise</u>:** Ask them to make a line graph, using Chart.js, of the top 5 Base MSRPs in Snohomish County, King County, and Benton County. Tell them that the labels should be the numbers from 1 to 5, where 1 is the highest and 5 is the fifth highest.
4. If they're stuck, make sure to help them! Remind them of sorting and filtering data, if they need a hint.
5. To complete the exercise successfully, the students should create a graph like this. If you have time, discuss whether a line graph was the best visualization to use and why/why not.



Visualization Assignment (20 minutes)

Purpose:

Familiarize students with use of Chart.js and practice creating embedded visualizations.

Materials:

Notepad++ Text Editor
Chart.js
Chart.js knowledge
Candy.csv dataset

Directions:

1. Give students the dataset near the end of class with the instructions for them to create the best visualization they can before the next class period.
2. Students will upload pictures of their visualizations to a Padlet during the next module and share if they thought theirs was truly spectacular.
3. Send students a link to the instructions document, just in case the students need more clarification: Instructions Document.
4. Provide the students with the remainder of the module (~15 minutes) to work on their visualizations.