



# EECS E6893 Big Data Analytics

## HW3: Data visualization

### Tutorial I -- HTML/CSS/JS + D3 Basics

Hung-Kai Huang  
Fall 2026

# Agenda

- Webpages with HTML, CSS and JavaScript.
- SVG, D3 and updating data views.
- HW3 datasets and exercises.

# What is a WebPage

A webpage is three layers working together:

- **HTML** (structure/content): what — headings, paragraphs, images, SVGs, containers.
- **CSS** (style/presentation): how it looks — colors, fonts, layout, spacing, responsive rules.
- **JavaScript** (behavior/logic): what it does — fetching data, building charts, reacting to user input.

*example file structure for hw part1:*

```
part1/                                # Seattle Weather visualizations
  index.html
  styles.css                          # (external CSS)
  app.js                             # (your JS logic)
```

You may submit a single HTML file (with `<style>` and `<script>` inside) or separate CSS/JS files. External files are recommended but not required. JavaScript is required for data loading and D3 visualizations; CSS is optional except for the Q1 paragraph styling, which may be done in an internal `<style>` block.

# HTML

- HTML is a language for describing web pages.
- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- HTML is not a programming language, it is a **markup language**  
A markup language is a set of **markup tags**
- HTML uses **markup tags** to describe web pages

Tutorials for HTML: [https://www.w3schools.com/html/html\\_intro.asp](https://www.w3schools.com/html/html_intro.asp)

```
<!doctype html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>Minimal D3 Page</title>
```

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<link href="styles.css" rel="stylesheet">
```

```
<style>#note{font-size:16px;text-indent:1.5em;color:#333}</style>
```

```
<script src="https://d3js.org/d3.v7.min.js"></script>
```

```
<script defer src="app.js"></script>
```

```
</head>
```

```
<body>
```

```
<header>
```

```
<h1 class="title">Seattle Weather</h1>
```

```
</header>
```

```
<main>
```

```
<section id="controls">
```

```
<label>Bins <input id="bins" type="range" min="5" max="25" step="5" value="10"></label>
```

```
<label>Variable
```

```
<select id="varSelect">
```

```
<option>precipitation</option>
```

```
<option>temp_max</option>
```

```
<option>temp_min</option>
```

```
<option>wind</option>
```

```
</select>
```

```
</label>
```

```
</section>
```

`<!doctype html>` tells the browser “modern HTML.” Everything lives inside `<html> ...</html>`

`<head>` = metadata (title, charset, CSS/JS links, etc)

Character encoding: always set **UTF-8** so symbols render correctly

`<body>` = what actually renders on the page (content, SVGs, etc.)

# Elements, tags, attributes

- Element = opening tag, content, closing tag: `<p>Hello</p>`.
- Void elements (no closing tag): `<img>`, `<meta>`, `<link>`, `<br>`, `<hr>`, `<input>`.
- Tag: the literal markup tokens like `<p>` or `</p>`
- Attributes : a name="value" pair placed inside a start tag to add data, ids, classes, links, ARIA info, etc.
- Global attributes: id, class, style, title, hidden, data-\*

```
<section id="charts">
```

```
<div id="histogram" class="chart"></div>
```

```
<div id="pie" class="chart"></div>
```

```
<div id="line" class="chart"></div>
```

```
</section>
```

`<div>` = tag

`id="histogram"` = global attribute (unique hook for JS/CSS)

`class="chart"` = global attribute (style/select as a group)

`</div>` = closing tag

```
<section id="summary">
```

```
<h2>Observations</h2>
```

```
<p id="note">Write 2–3 insights here.</p>
```

```
</section>
```

```
</main>
```

```
<footer>
```

```
<a href="#top" id="backToTop">Back to top</a>
```

```
</footer>
```

```
</body>
```

```
</html>
```

Together they form one div element (empty content).

`<p>` = tag for a paragraph

`id="note"` = global attribute

`Write 2–3 insights here` = content

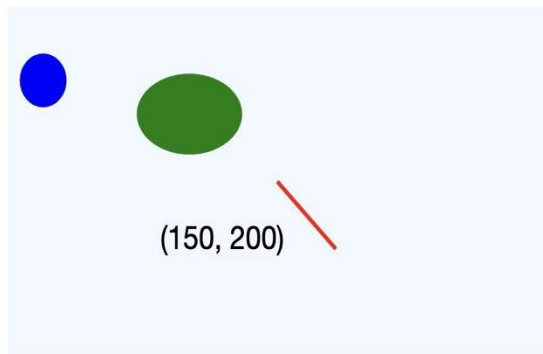
`</p>` closes the element.

# What is SVG

- SVG = Scalable Vector Graphics, an XML-based graphics language built into HTML.
- It's resolution-independent (no blur when zooming) and each shape is a DOM node you can style with CSS and manipulate with JavaScript/D3.
- Coordinate system: origin (0,0) at top-left; +x to the right, +y downward.
- Best for charts, icons, labels, interactive shapes; D3 targets SVG by default.

# SVG in HTML

```
<svg width="500" height="300"> <!-- some SVG -->
  <rect x="20" y="20" width="460" height="260" fill="aliceblue"></rect>
  <circle cx="50" cy="75" r="20" fill="blue"></circle>
  <ellipse cx="175" cy="100" rx="45" ry="30" fill="green"></ellipse>
  <text x="150" y="200">(150, 200)</text>
  <line x1="250" y1="150" x2="300" y2="200" stroke="red" stroke-width="3"></line>
</svg>
```



```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<svg width="150" height="100" xmlns="http://www.w3.org/2000/svg">
```

```
<rect width="100%" height="100%" fill="green" />
```

```
<circle cx="75" cy="50" r="40" fill="yellow" />
```

```
<text x="75" y="60" font-size="30" text-anchor="middle" fill="red">SVG</text>
```

```
</svg>
```

```
</body>
```

```
</html>
```



More about SVG: [https://www.w3schools.com/graphics/svg\\_intro.asp](https://www.w3schools.com/graphics/svg_intro.asp)

# Steps to create a HTML file and view in browser

- Step 1: Open a text editor or notepad on your machine. (VSCode)
- Step 2: Enter the lines of code:

```
<> myfirstpage.html >  html
1  <!DOCTYPE html>
2  <html>
3
4      <head>
5          |   <title>Page Title</title>
6      </head>
7
8      <body>
9          |   <h1>This is a Heading</h1>
10         |   <p>This is a paragraph.</p>
11     </body>
12 </html>|
```

- Step 3: Save the file as myfirstpage.html (go to File-Save As give File name: myfirstpage.html-choose save as type: All Files-click save)
- Step 4: View document in web browser (click on the html file and it will be opened in browser)



## **This is a Heading**

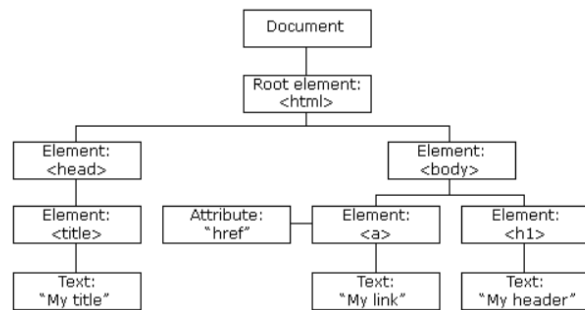
This is a paragraph.

# What is DOM

## What JS/D3 actually changes

- DOM = Document Object Model → the browser's in-memory tree of your HTML page.
- Each HTML tag becomes a node (element node, text node, etc.) in that tree.
- We “hook” nodes with id (unique) and class (reusable) for styling and scripting.
- JavaScript selects nodes, reads/sets attributes/text, and listens to events.
- D3 sits on top of the DOM: it binds data to nodes and updates them when data/UI changes.

The HTML DOM Tree of Objects



# About JavaScript

- JavaScript is a programming language
- used by Web browsers to create a dynamic and interactive experience for the user
- Some of the dynamic website enhancements performed by JavaScript are:  
Loading new content or data onto the page without reloading the page,  
Rollover effects and dropdown menus etc.
- Some of its most powerful features involve asynchronous interaction with a remote server.

JavaScript adds behavior: fetch data, build charts, handle user interaction.

# JavaScript in Web Pages

Three ways to include JS:

- External file (recommended) `<script defer src="app.js"></script>`

- Inline block (OK for tiny demos) 

```
<script>
    alert("Hello World!");
</script>
```

- ES Modules (when importing other JS files)

```
<script type="module">
    import { scaleLinear } from 'https://cdn.skypack.dev/d3-scale';
    console.log(scaleLinear);
</script>
```

## Loading with D3:

```
<script src="https://d3js.org/d3.v7.min.js"></script>
<script defer src="app.js"></script>
```

Your app.js runs after the DOM is ready.

# JavaScript Language Basics

- Variables

```
let count = 0;           // reassignable (block scope)
const title = "HW3";     // not reassignable (block scope)
var legacy = 1;          // function scope (avoid)
```

Use const by default, switch to let only when you plan to reassign.

- Types

```
typeof 3           // "number"
typeof "hi"        // "string"
typeof []          // "object"
Array.isArray([])  // true
```

Primitive: number, string, boolean, null, undefined, bigint, symbol

Objects: arrays, plain objects, functions, dates, etc.

- Strings & template literals

```
const name = "D3";  
`Hello, ${name}!`    // "Hello, D3!"  
"Wind: " + 12.5
```

- Arrays

```
const xs = [3, 1, 4];  
xs.push(2);           // [3,1,4,2]  
xs.map(d => d * 2);    // [6,2,8,4]  
xs.filter(d => d > 2);  // [3,4]  
xs.reduce((a,b)=>a+b, 0); // 10  
for (const x of xs) { /* ... */ } // iterate values
```

- Objects (records/rows)

```
const row = { date: "2024-01-01", wind: 8.2 };  
row.wind           // 8.2  
row["date"]        // "2024-01-01"  
  
// Destructuring:  
const { date, wind } = row;
```

- Functions

```
function sum(a, b) { return a + b; }  
const sum2 = (a, b) => a + b;    // arrow function
```

- Control flow

```
if (x > 5) { ... } else { ... }  
  
for (let i = 0; i < n; i++) { ... } // classic loop  
for (const v of arr) { ... }       // iterate values
```

- Truthy / falsy (common gotcha)

Falsy values: false, 0, "", null, undefined, NaN.

```
if ("") console.log("no");    // not printed  
if ("ok") console.log("yes"); // printed
```

JS tutorial:

[https://www.w3schools.com/js/js\\_intro.asp](https://www.w3schools.com/js/js_intro.asp)

# About CSS

CSS is the language we use to style a Web page.

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen
- CSS saves a lot of work. It can control the layout of **multiple web pages all at once**
- External stylesheets are stored in CSS files

# How to use CSS in a page

- External stylesheet (recommended for most styling)

`<link rel="stylesheet" href="styles.css">`

- Internal stylesheet (inside `<style>` in `<head>`)

Great for small page-specific rules; required in Q1 to style your paragraph.

`<style>#observations{font-size:16px;text-indent:1.5em;color:#333}</style>`

- Inline style (avoid for maintainability)

`<p style="color:#333">...</p>`

Tip: Use external CSS for everything, plus some internal rule to meet the HW requirement.

## Example: style.css

```
/* Base */
:root { --ink: #222; --brand: #2f6fed; --bg: #fafafa; }
html,body{ margin:0; padding:0; font-family:system-ui, Segoe UI, Roboto, Arial, sans-serif; color:var(--ink); background:var(--bg); }
main{ max-width:980px; margin:24px auto; padding:0 16px; }

/* Headings & sections */
h1{ color:var(--brand); line-height:1.2; margin:16px 0 8px; }
h2{ line-height:1.25; margin:20px 0 8px; }
section{ margin:20px 0; }

/* Charts layout (pick one: Grid OR Flex) */
#charts{ display:grid; grid-template-columns:repeat(auto-fit,minmax(320px,1fr)); gap:16px; }
.chart{ background:#fff; border:1px solid #e8e8e8; border-radius:8px; padding:12px; }

/* Controls */
label{ margin-right:16px; font-weight:600; }
input[type="range"]{ vertical-align:middle; }
select{ padding:2px 6px; }

/* SVG & axes */
svg{ width:100%; height:auto; display:block; }
.axis text{ font-size:12px; }
.axis path,.axis line{ stroke:#444; shape-rendering:crispEdges; }

/* Table helper (for Part II) */
table{ border-collapse:collapse; width:100%; background:#fff; }
th,td{ border:1px solid #eee; padding:6px 8px; font-size:14px; text-align:left; }
thead th{ background:#f6f6f6; }
```

# How to use CSS in a page

- Hook css style file to HTML

In HTML head:

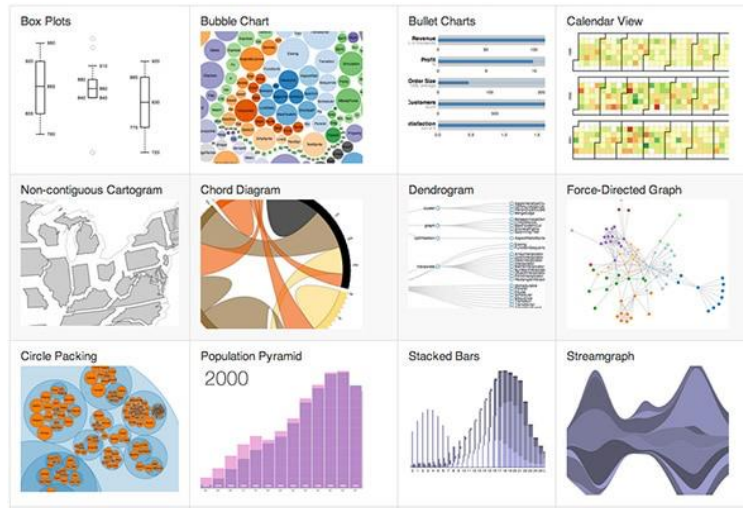
```
<link rel="stylesheet" href="styles.css">  
<style>#observations{font-size:16px;text-indent:1.5em;color:#333}</style>
```

CSS tutorial: [https://www.w3schools.com/css/css\\_intro.asp](https://www.w3schools.com/css/css_intro.asp)

# Introduction to D3.js



- D3 = Data-Driven Documents.
- D3.js is a **JavaScript library** for manipulating HTML based on data.
- It lets you bind data to HTML/SVG elements and map numbers to pixels so you can draw charts with code.
- D3 is a toolbox: selections, data joins, scales, axes, shapes, layouts, transitions.



More about D3: <https://d3js.org/>

Github: <https://github.com/d3/d3>

# Why use D3.js

- Dynamic and Interactive
- Directly binds data to DOM (Document Object Model)
- Extensive flexibility and control over visualization
- Large community and extensive documentation

Note: If your page loads CSV/JSON (e.g., `d3.csv(...)`, `fetch(...)`), the browser's security model blocks cross-origin reads on `file://`.  
**Run a local server so your page is served over HTTP.**

Two ways to run a server:

- Option A: Python (built-in)  
`cd path/to/your/project`  
`python -m http.server 8000`
- then open `http://localhost:8000`
- 
- Option B: VS Code Live Server
- Install the "Live Server" extension, then right-click `index.html` → "Open with Live Server"

# D3 Live Examples

- D3 Examples page: <https://observablehq.com/@d3/gallery>

Why Observable

Product

Enterprise

Resources

Pricing

Search

Sign in

Sign up

D3

Bring your data to life.

Public

2 collections

By Mike Bostock

Edited Aug 23

Paused

ISC

101 forks

Importers

847 Likes

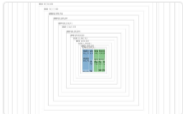
Fork

## D3 gallery

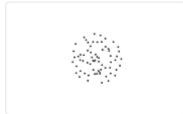
Looking for a good D3 example? Here's a few (okay, 169...) to peruse.

### Animation

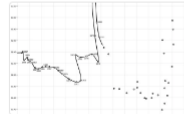
D3's [data join](#), [interpolators](#), and [easings](#) enable flexible [animated transitions](#) between views while preserving [object constancy](#).



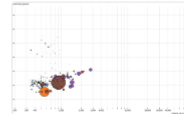
Animated treemap



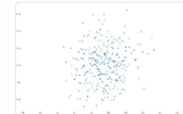
Temporal force-directed gra...



Connected scatterplot



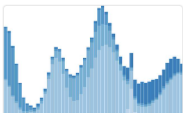
The wealth & health of natio...



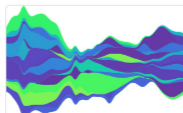
Scatterplot tour



Bar chart race



Stacked-to-grouped bars



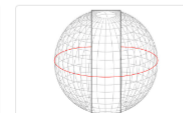
Streamgraph transitions



Smooth zooming



Zoom to bounding box



Orthographic to equirectang...



World tour

# Starting D3

- “src” in script tag

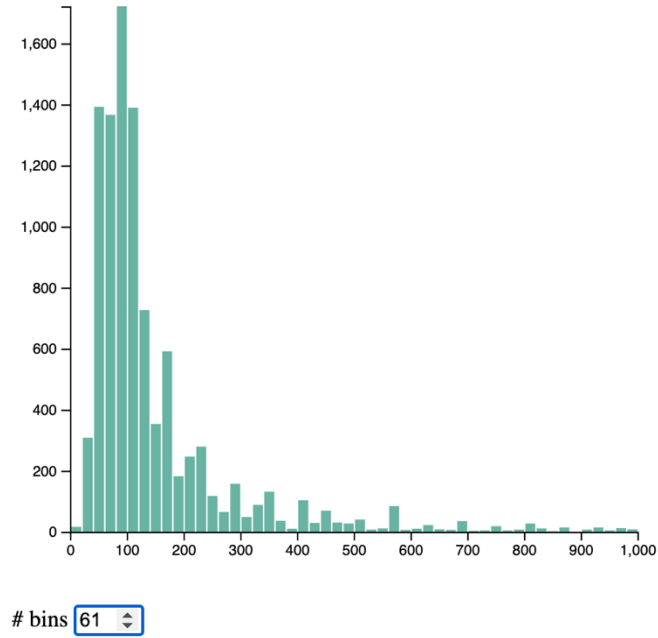
```
<> d3basic.html > html
1  <!DOCTYPE html>
2  <html>
3
4      <head>
5          <script src="https://d3js.org/d3.v4.js"></script>
6      </head>
7
8      <body>
9          <p>Hello</p>
10     </body>
11 </html>
```

- Load data in D3

```
d3.csv('seattle-weather.csv', d3.autoType).then(data => {
  // numbers & dates parsed for you
  const wind = data.map(d => d.wind).filter(Number.isFinite);
  render(wind, 10);
});
```

# D3 examples

## (a) Histogram with interactive component



```

t2 > <> histogrammb.html > ...
1  <!DOCTYPE html>|
2  <meta charset="utf-8">
3
4  <!-- Load d3.js -->
5  <script src="https://d3js.org/d3.v4.js"></script>
6
7  <!-- Create a div where the graph will take place -->
8  <div id="my_dataviz"></div>
9
10 <p>
11   <label># bins</label>
12   <input type="number" min="1" max="100" step="30" value="20" id="nBin">
13 </p>
14
15
16 <script>
17
18   // set the dimensions and margins of the graph
19   var margin = {top: 10, right: 30, bottom: 30, left: 40},
20     width = 460 - margin.left - margin.right,
21     height = 400 - margin.top - margin.bottom;
22
23   // append the svg object to the body of the page
24   var svg = d3.select("#my_dataviz")
25     .append("svg")
26     .attr("width", width + margin.left + margin.right)
27     .attr("height", height + margin.top + margin.bottom)
28     .append("g")
29     .attr("transform",
30       "translate(" + margin.left + "," + margin.top + ")");
31
32   // get the data
33   d3.csv("https://raw.githubusercontent.com/holtzy/data_to_viz/master/Example_dataset/1_OneNum.csv", function(data) {
34
35     // X axis: scale and draw:
36     var x = d3.scaleLinear()
37       .domain([0, 1000]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })
38       .range([0, width]);
39     svg.append("g")
40       .attr("transform", "translate(0," + height + ")")
41       .call(d3.axisBottom(x));
42
43     // Y axis: initialization
44     var y = d3.scaleLinear()
45       .range([height, 0]);
46     var yAxis = svg.append("g")
47

```

```

48 // A function that builds the graph for a specific value of bin
49 function update(nBin) {
50     // set the parameters for the histogram
51     var histogram = d3.histogram()
52         .value(function(d) { return d.price; }) // I need to give the vector of value
53         .domain(x.domain()) // then the domain of the graphic
54         .thresholds(x.ticks(nBin)); // then the numbers of bins
55
56     // And apply this function to data to get the bins
57     var bins = histogram(data);
58
59     // Y axis: update now that we know the domain
60     y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously
61     yAxis
62         .transition()
63         .duration(1000)
64         .call(d3.axisLeft(y));
65     // Join the rect with the bins data
66     var u = svg.selectAll("rect")
67         .data(bins)
68     // Manage the existing bars and eventually the new ones:
69     u
70         .enter()
71         .append("rect") // Add a new rect for each new elements
72         .merge(u) // get the already existing elements as well
73         .transition() // and apply changes to all of them
74         .duration(1000)
75         .attr("x", 1)
76         .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
77         .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
78         .attr("height", function(d) { return height - y(d.length); })
79         .style("fill", "#69b3a2")
80     // If less bar in the new histogram, I delete the ones not in use anymore
81     u
82         .exit()
83         .remove()
84 }
85 // Initialize with 20 bins
86 update(20)
87
88 // Listen to the button -> update if user change it
89 d3.select("#nBin").on("input", function() {
90     update(+this.value);
91 });
92
93 });
94 </script>

```

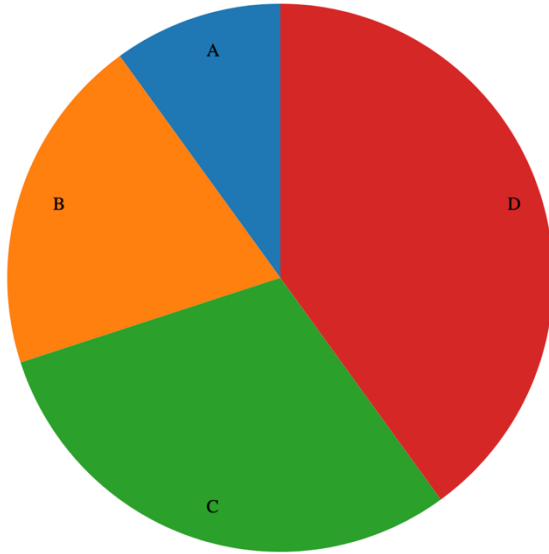
## (b) Table with CSS style

| Name    | Age | Job      |
|---------|-----|----------|
| Alice   | 25  | Engineer |
| Bob     | 30  | Designer |
| Charlie | 28  | Teacher  |

```
t2 > <? table.html > <? html > <? head > <? meta
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <script src="https://d3js.org/d3.v5.min.js"></script>
6   <style>
7     /* Table styles */
8     table {
9       border-collapse: collapse;
10      width: 100%;
11    }
12    th, td {
13      border: 1px solid black;
14      padding: 8px 12px;
15      text-align: left;
16    }
17    th {
18      background-color: #f2f2f2;
19    }
20  </style>
21 </head>
22 <body>
23   <table></table>
24   <script>
25     // Sample data
26     const data = [
27       { Name: "Alice", Age: 25, Job: "Engineer" },
28       { Name: "Bob", Age: 30, Job: "Designer" },
29       { Name: "Charlie", Age: 28, Job: "Teacher" }
30     ];
```

```
31 // Select the table
32 const table = d3.select("table");
33 // Append table headers
34 const thead = table.append("thead");
35 thead.append("tr")
36   .selectAll("th")
37   .data(Object.keys(data[0]))
38   .enter().append("th")
39   .text(d => d);
40 // Append table rows and cells
41 const tbody = table.append("tbody");
42 tbody.selectAll("tr")
43   .data(data)
44   .enter().append("tr")
45   .selectAll("td")
46   .data(d => Object.values(d))
47   .enter().append("td")
48   .text(d => d);
49 </script>
50 </body>
51 </html>
```

(c) Pie chart



```
t2 > <> pie.html > <html> <head> <script>
1 <doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <script src="https://d3js.org/d3.v5.min.js"></script>
6 </head>
7 <body>
8   <svg width="500" height="500"></svg>
9   <script>
10    // Sample data
11    const data = [
12      { label: "A", value: 10 },
13      { label: "B", value: 20 },
14      { label: "C", value: 30 },
15      { label: "D", value: 40 }
16    ];
17
18    // SVG setup
19    const width = 500;
20    const height = 500;
21    const radius = Math.min(width, height) / 2;
22    const svg = d3.select("svg")
23      .append("g")
24      .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");
25
26    // Color scale
27    const color = d3.scaleOrdinal(d3.schemeCategory10);
28
29    // Pie layout
30    const pie = d3.pie().value(d => d.value);
31    const path = d3.arc().outerRadius(radius - 10).innerRadius(0);
32    const labelArc = d3.arc().outerRadius(radius - 40).innerRadius(radius - 40);
33
34    // Bind data, create pie chart slices
35    const g = svg.selectAll("arc")
36      .data(pie(data))
37      .enter().append("g")
38      .attr("class", "arc");
39
40    g.append("path")
41      .attr("d", path)
42      .style("fill", d => color(d.data.label));
43
44    g.append("text")
45      .attr("transform", d => "translate(" + labelArc.centroid(d) + ")")
46      .attr("dy", ".35em")
47      .text(d => d.data.label);
48
49    </script>
50 </body>
51 </html>
```

# D3 versions in the examples

| Example code         | D3 v7                                          |
|----------------------|------------------------------------------------|
| d3.histogram()       | d3.bin()                                       |
| CSV callback         | d3.csv(path, d3.autoType).then(...)            |
| enter / merge / exit | Still available; selection.join() is an option |

The illustrated histogram uses D3 v4. Use D3 v7 for HW3.

A numeric thresholds(k) is a suggestion. Exact bin counts need explicit boundaries.

# Periodic API updates

- A timer requests data or refreshes API results at a chosen interval.
- Each update adds a record to the webpage without reloading it.
- Provider data time can stay unchanged between requests.
- A price comparison needs a clearly identified baseline.

The update interval and available data depend on the API plan.

# HW3 overview

| Question   | Data                | Practice                         |
|------------|---------------------|----------------------------------|
| Q1 weather | Seattle weather     | Static charts and D3 controls    |
| Q1 stocks  | Three chosen stocks | API prices and periodic updates  |
| Q2         | NVDA, TSLA and SPY  | Daily stock-data story           |
| Q3         | Your own dataset    | Written story and evidence views |

Q2 uses common daily-data rules. Each student fetches data and computes results.

# References

- [https://www.w3schools.com/js/js\\_htmlDOM.asp](https://www.w3schools.com/js/js_htmlDOM.asp)
- Vitaly Shmatikov CS 345 Introduction to JavaScript
- Sarbjit Kaur Introduction to HTML
- <https://d3js.org/>
- <https://developer.mozilla.org/en-US/docs/web/SVG>
- <https://livebook.manning.com/book/d3js-in-action-second-edition/chapter-7/70>
- <https://d3-graph-gallery.com/index.html>
- [https://www.w3schools.com/graphics/svg\\_intro.asp](https://www.w3schools.com/graphics/svg_intro.asp)
- [https://www.w3schools.com/html/html\\_intro.asp](https://www.w3schools.com/html/html_intro.asp)
- [https://www.w3schools.com/js/js\\_intro.asp](https://www.w3schools.com/js/js_intro.asp)
- [https://www.w3schools.com/css/css\\_intro.asp](https://www.w3schools.com/css/css_intro.asp)
- [https://www.w3schools.com/ai/ai\\_d3js.asp](https://www.w3schools.com/ai/ai_d3js.asp)

THANK YOU !!