Introduction to Web Development

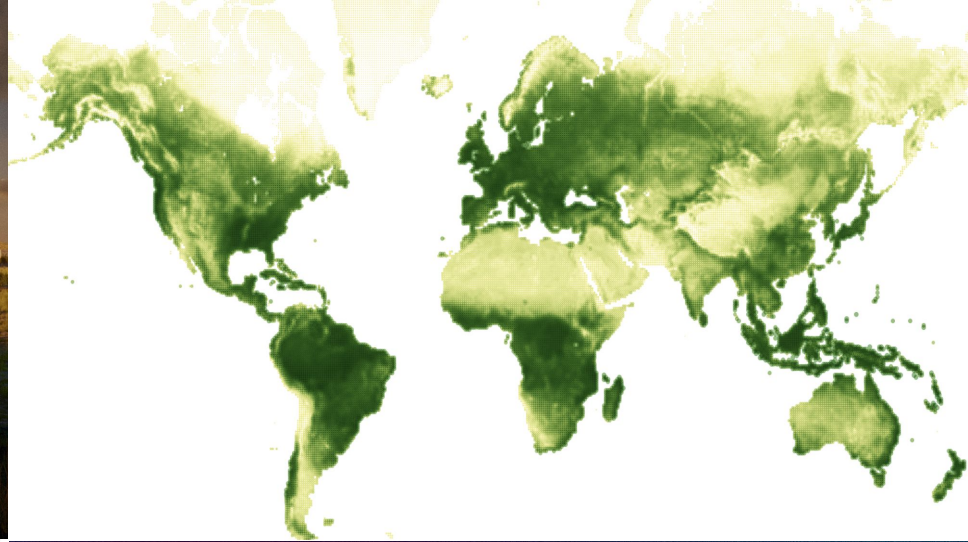
Session 1 & 2

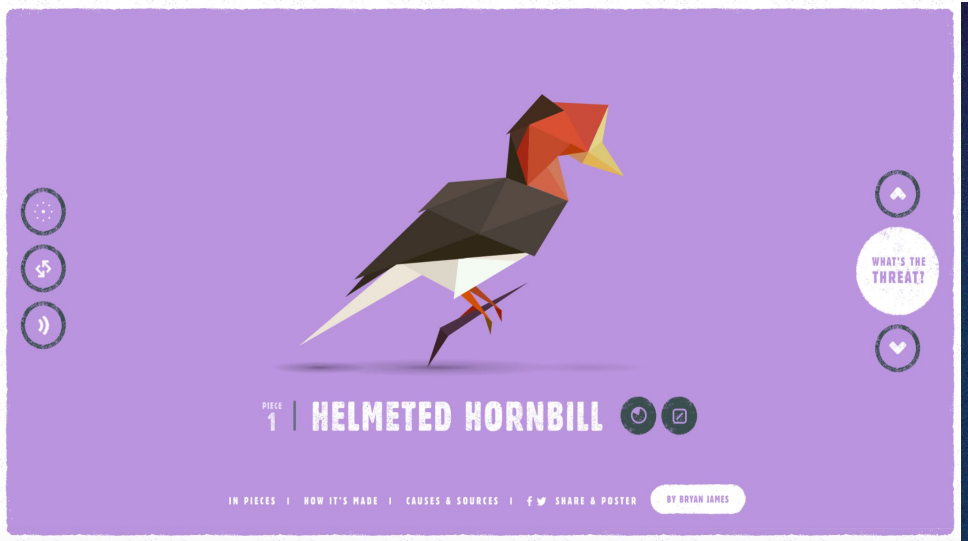
Session 1 & 2 Objectives

- CS / Programming knowledge check — How much do you know?
- IDE installation troubleshooting
- Explore applications of web development/design
- Introduction to HTML and CSS
- Think about long-term personal project ideas

Resources

- CCA Outline and Session Slides

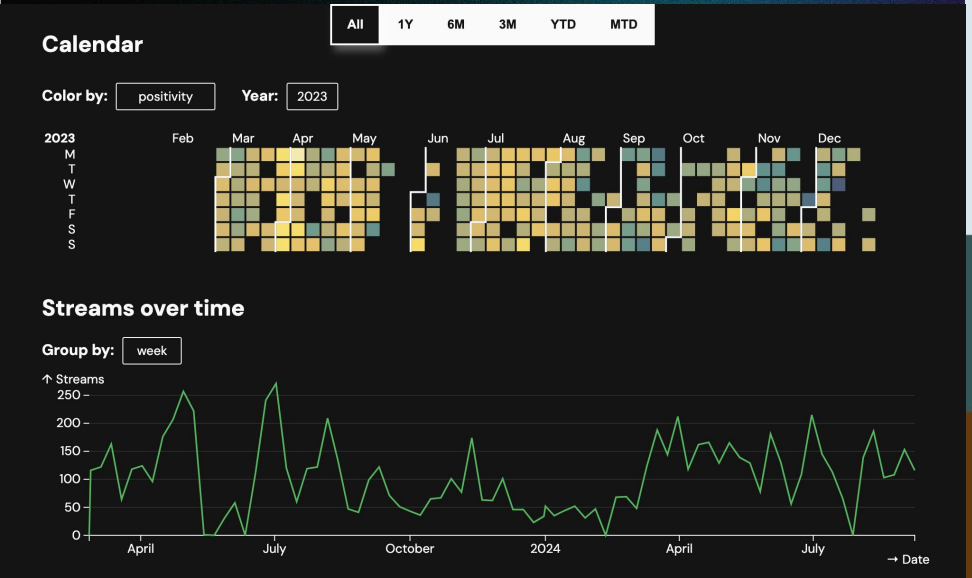


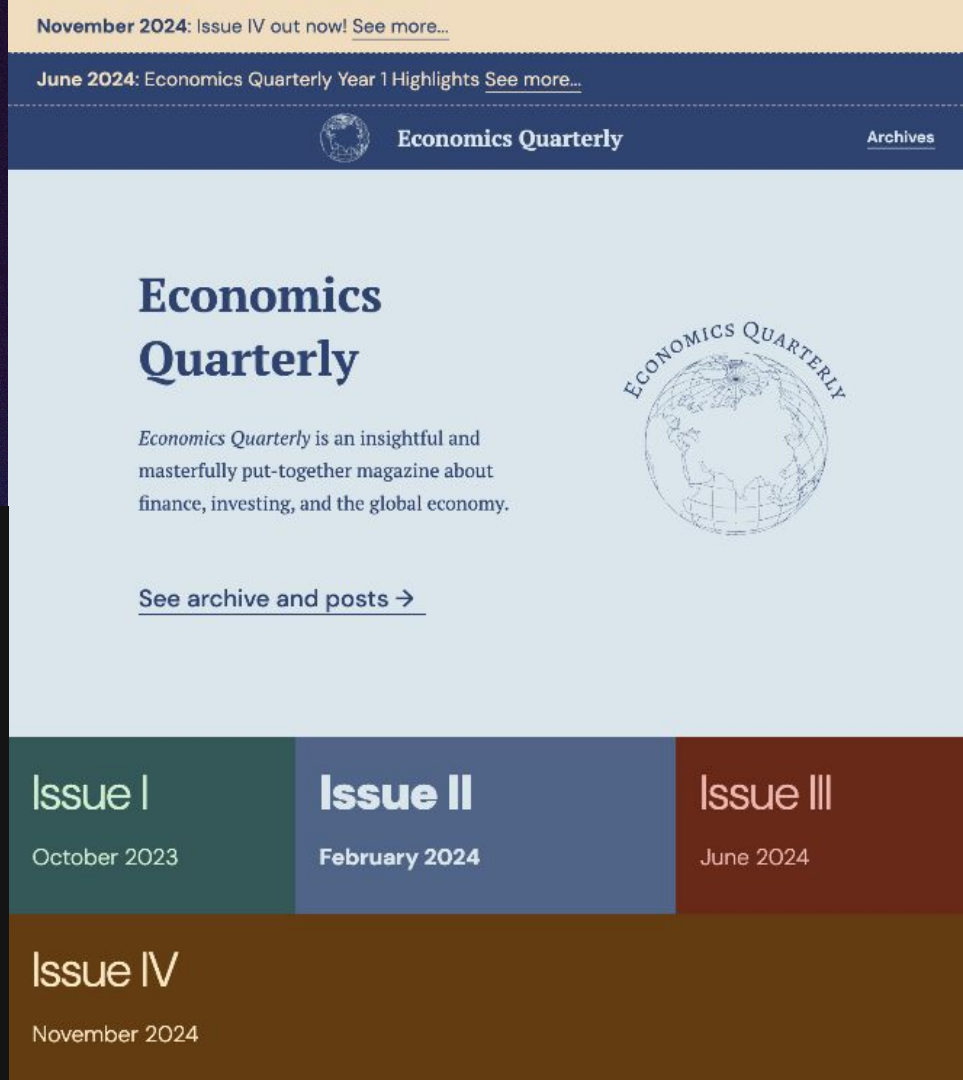


- Persepolis Reimagined
- Species in Pieces
- Breathing Earth

Some personal projects:

- Spotify Statistics
- Economics Quarterly





HTML, CSS, and JS

- HTML (Hypertext Markup Language) — structure
- CSS (Cascading Style Sheets) — styling
- JS (JavaScript) — functionality

Note: introduce Developer Tools

HTML — Elements and syntax

- All pages consist of **elements**, which are made of matching tags:

  ```
  <a href="" target="_blank">CONTENT</a>
  ```

  ```
  <img src="img.png" alt="description" />
  ```

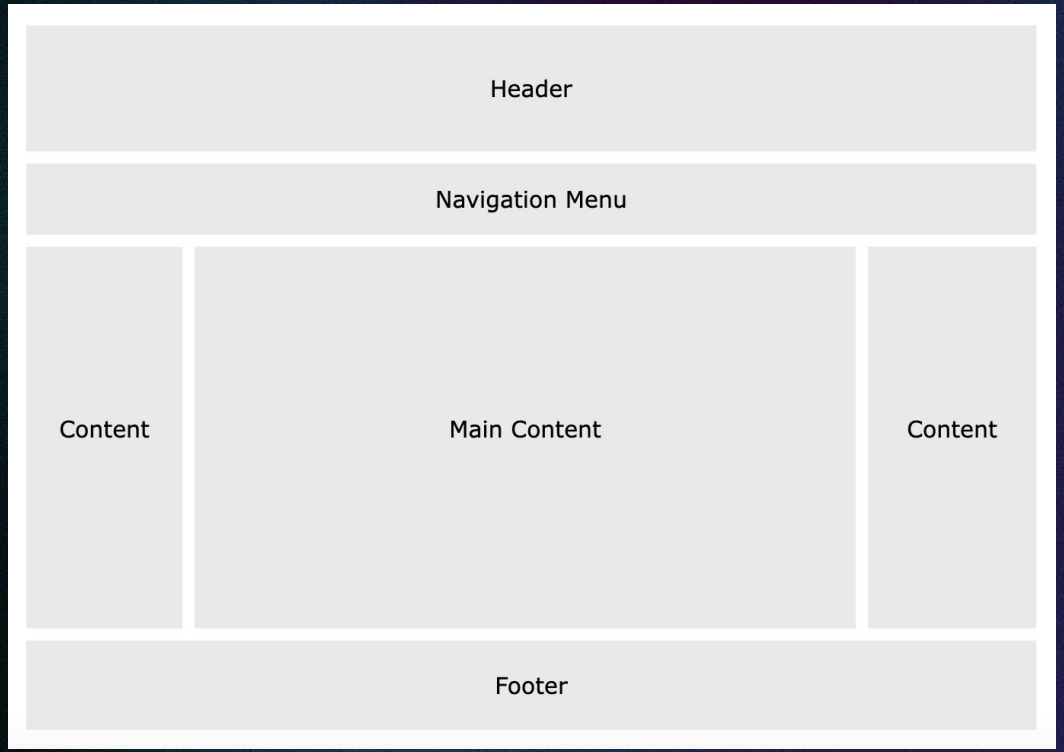
  which wrap around **content**. Element **attributes** are placed within the starting tag in the form `attribute_name="value"`.

  Example tags: h1-h6; p; div; span; button; a; label; table

HTML — Links

- Can link to other sections of page (using ids), relative and absolute links.

HTML — Website layout



HTML — Document structure

- **HTML document structure** (.html; <!DOCTYPE html>, <html>, <head>, and <body>)
- Side note: absolute and relative URLs

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is a web page.</p>
  </body>
</html>
```

HTML — Self-closing elements

- **Self-closing elements**: `<element />`
- Examples — br; embed; hr; img; input; link; meta;

HTML — h1-h6; p

- Headings: h1-h6 provide hierarchy (book analogy)
- Paragraph — bold, italic
- Automatic removal of extra whitespace
- Horizontal rule (hr) – breaks page into sections with a line
- Line breaks (br) for line breaks in element
- Display: block vs. span

CSS — Syntax

```
selector {
    property: value; /* <= declaration */
    another-property: value; /* <= declaration */
}
```

CSS Selectors

- Selectors:
  - simple selectors (tag name, id, class)
  - combinator selectors (based on specific relationship)
  - pseudo-class selectors (based on state)
  - pseudo-elements selectors (select/style part of element)
  - attribute selectors (based on attribute/attribute value)

CSS Examples

- `p {}`
  `#unique-element {}`
  `.common-element {}`
- `p.centered.red {}` vs. `div p.centered.red {}`
- `* {}`
- `h1, h2, h3 {}`

CSS — Location

- Where can CSS go? (external stylesheet, in <style> tag, inline)
- Precedence:

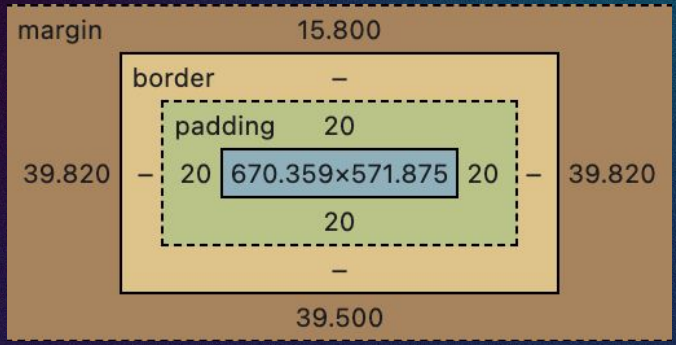
  Inline > External & internal (depend on order) > Browser default

CSS — Color styling, Borders

- Common properties:
  - color; background-color; opacity
  - background-image
  - border-color; border-style; border-width; border-radius
- RGB, RGBA, HSL and HEX colors, keyword colors

CSS — Margins and Padding

- margin-top; margin-bottom; margin-left; margin-right;
- margin: top right bottom left;
- margin: auto/inherit/0;
- Same for padding
- Units:
    - Absolute: cm; mm; in; px; pt; pc;
    - Relative: em; %; vw; vh;

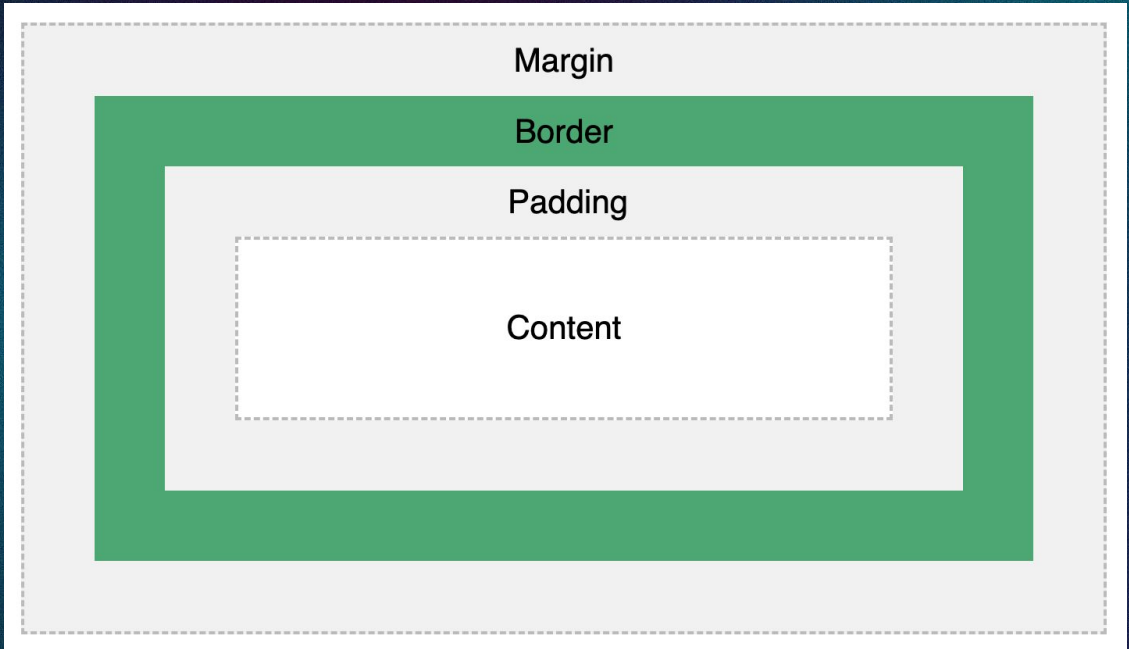


CSS — Box Model

box-sizing

`border-box` — includes border, padding when setting width/height

`content-box` (default) — sets width/height of content box



Before Session 2

- Try to get familiar with using your IDE. You can begin learning on your own (preview slides and check the suggested resources)
- Review slides.
- Think about possible personal project ideas (will spend final few sessions working on them) [see the outline for ideas]

Before Session 3

- Review slides.
- Learn more HTML and CSS (W3Schools), code something for yourself!

Introduction to Web Development

Session 3

Objectives

- Continuing with HTML and CSS

CSS: Text and typography

- Text styling properties: color; text-align; line-height;
- Typography (W3Schools reference):
  - ```
    font-family: "DM Sans", "Arial", sans-serif;
    font-weight: 600;
    font-style: italic;
    font-size: 18pt;
    ```
  - Google Fonts; variable fonts
    ```

CSS: Height and width

- Units
- max-width and max-height; min-width and min-height for setting maximum and minimum height of element.
- width/height: clamp(smallest, ideal, greatest);
- Relative width/height (e.g., auto or %) require parent elements to have specified absolute width/height

CSS: Display

- Block elements (div, h1-h6, p, section) span entire width of page; inline elements (a, img, span) take as much width as needed.
- display: none / inline / block / inline-block / flex / grid

CSS: Position

- position: static (default) / relative / absolute / fixed / sticky; and top/left/right/bottom attributes
  - relative: alters position relative to original position
  - fixed: element taken from page and position relative to view
  - absolute: positioned relative to nearest positioned parent element.
  - sticky: toggles between relative and fixed based on scroll

CSS: Alignment

- Alignment
  - Centering — margin: auto (requires set width, display: block); text-align;
  - Left/right — position absolute or float

Responsive design

- Recognize the importance for designing for multiple screen sizes. You have most likely been designing for desktop, however ~**63%** of web traffic comes from **mobile devices**.
- CSS Media queries allow for conditional styling based on viewport:

```
@media screen and (max-width: 800px) {}

@media screen and (min-width: 800px) {}
```

Flexbox and grid

- Flexbox (parent element): display; flex-direction; flex-wrap; justify-content; align-items; align-content; gap;
- Grid

Guided Project — Basic card component

- Start working on Guided Project

For next session

- Review slides. CSS Tricks
- W3Schools extra:
  - HTML: Links, Images, Lists, Head, Semantics, Layout
  - CSS: Links
- Try finish Guided Project — Basic card component.

Introduction to Web Development

Session 4

Objectives

- Continuing with HTML and CSS
- Introduction to JavaScript

CSS — Transitions

- Add hover interactions

```
a {
    transition: color 300ms;
    color: black;
}

a:hover {
    color: white;
}
```

CSS — Variables

- Applications: dark mode, transitions, color scheme/theme
-
```
:root {
    --color: rgb(20, 20, 20); --bg: rgb(250, 250, 250);
}
body {
    color: var(--color); background-color: var(--bg);
}
```

JS — Introduction

- Placed in `<script>` tags in `<head>`, `<body>` or in external file.
- C/C++/Java-like syntax
- Output methods: console.log(); window.alert()
- Variable declaration: (keywords `let`, `var`, or `const`, just use `let`)

JS — Variables, functions, loops, conditionals

- `let name = "";`

  `if () {} else if {} else {}`

  `switch () {}`

  `for () {}`

  `while() {}`

  `function name(parameter) {}`

JS — Arrays, sets, maps

- Arrays — access elements, length, push, splice, forEach
- Sets — stores only unique elements
- Maps — stores data in <key, value> pairs (where given a key, its corresponding value can be returned)

JS — References

- Go through Syntax, Statements, Variables, Data types

JS — Practice

(*Sum the digits in an integer*) Write a program that reads an integer between **0** and **1000** and adds all the digits in the integer. For example, if an integer is **932**, the sum of all its digits is **14**.

Hint: Use the **%** operator to extract digits, and use the **/** operator to remove the extracted digit. For instance, **932 % 10 = 2** and **932 / 10 = 93**.

****5.25** (*Compute* π) You can approximate π by using the following summation:

$$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \cdots + \frac{(-)^{i+1}}{2i-1}\right)$$

Write a program that displays the π value for **i = 10000**, **20000**, …, and **100000**.

JS — Practice

**5.17 (*Display pyramid*) Write a program that prompts the user to enter an integer from 1 to 15 and displays a pyramid, as presented in the following sample run:

```
Enter the number of lines: 7 ↵ Enter
                1
              2 1 2
            3 2 1 2 3
          4 3 2 1 2 3 4
        5 4 3 2 1 2 3 4 5
      6 5 4 3 2 1 2 3 4 5 6
    7 6 5 4 3 2 1 2 3 4 5 6 7
```

**5.29 (*Display calendars*) Write a program that prompts the user to enter the year and first day of the year and displays the calendar table for the year on the console. For

JS — Practice

- Create a program that finds the prime factorization of a user-inputted number.
- **Challenge**: write a program that finds all prime numbers up to 10,000,000,000. There are approximately 455,052,511 such prime numbers.
    - Cache the list in localStorage or indexedDB, and load the saved computational list each time the page is reloaded to avoid restarting when page reloads

For next session

- Work on Guided Project 2; practice JS with examples
- Additional information (optional) — see W3Schools references for: CSS Animations, !important, Combinators, z-index, Navigation Bar, Dropdowns, Image Gallery, Tooltips
- W3Schools — CSS Buttons, Gradients, Shadows; JS Functions, Conditionals, Loops (for, while, break); CSS Tricks

Introduction to Web Development

Session 5, 6

Objectives

- Continuing with JavaScript
- Work session on Guided Projects 2, 3

JS — Objects

- Reference
- Link to JSON

JS — Classes

JS — Search and sort

- Array search and sort

JS — Practice

- Using: 1) A custom loop, and 2) built-in JS functions, sort [2, -8, 16, 0, 4, 9, 2] in descending order.
- Download 2019 World Happiness Report data. Write code to read the .csv file into an array of country objects with their corresponding scores in each sub-category specified in the data. **Extension**: download the data from 2015-2019, read and merge all data into a single array of country objects.

JS — Practice

- Implement linear search, binary search, bubble sort, selection sort, quick sort, insertion sort, merge sort
- Challenge: write a sudoku solver

JS — DOM interaction and event handling

- `document.querySelector; document.getElementById; document.getElementsByClassName`
- `element.addEventListener("event", function);`

For next session

- W3Schools: JS arrow function, search, sort, iteration, classes, DOM interaction
- Finish Guided Projects 2 and 3
- Finalize idea for final project

Introduction to Web Development

Session 7, 8

Objectives

- Final project introduction and work sessions

For next session

- Work on final project

Introduction to Web Development

Session 9

Objectives

- Finish project
- Share on GitHub — deploy with GitHub Pages
- Project sharing/mini-presentation
- Feedback form

Share Project

- Open an issue in Guided Projects and paste your repository link.
- Briefly prepare 2-minute presentation for your project (show what it does, what did you learn/challenges?)

Deployment (Repository Settings)