FALL 2021 / ONLINE

INTERACTIVE DEVELOPMENT

Friday, October 8

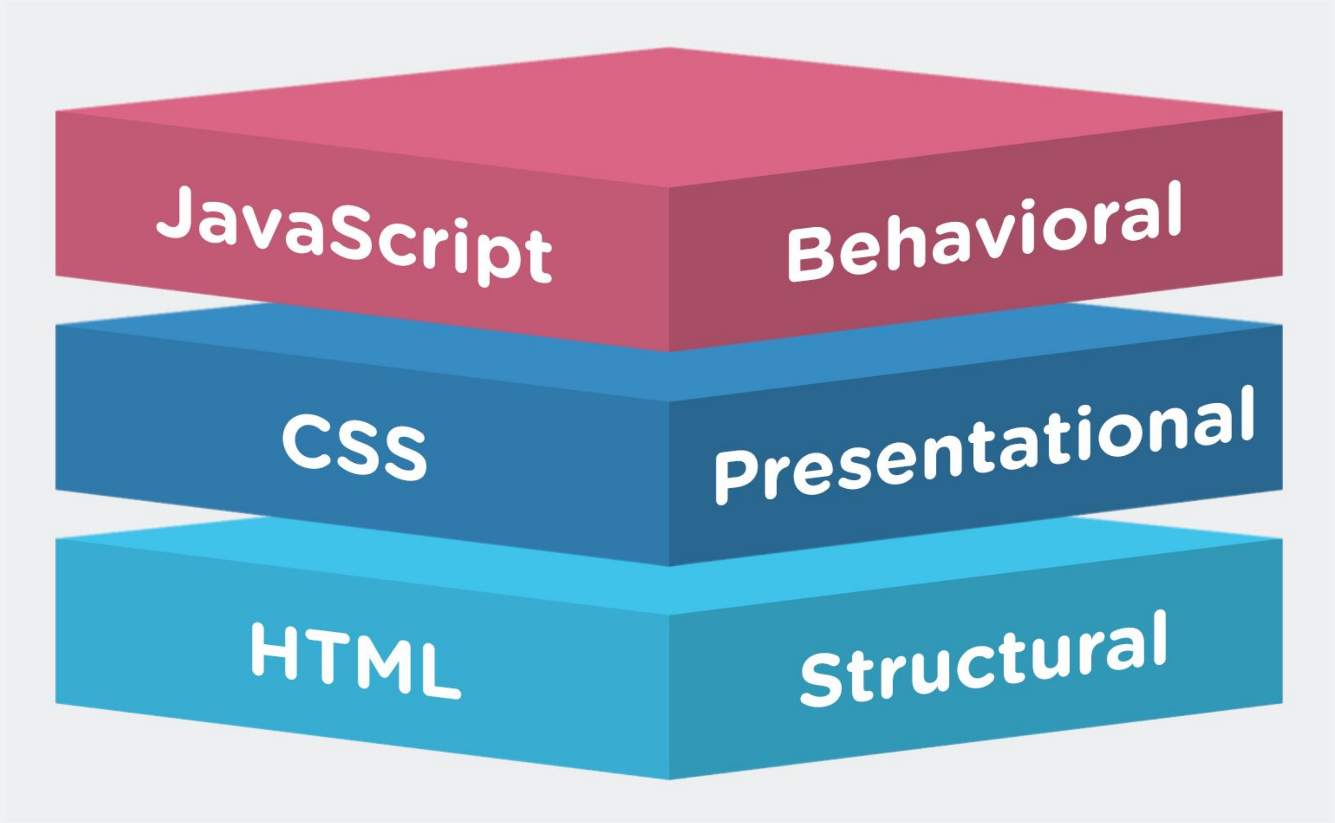
WARM-UP QUESTION:
FAVORITE FONT/TYPE FAMILY?

A04 RECAP

DEMOS + DISCUSSION

HOLD ON TIGHT!

WHAT IS JAVASCRIPT?



1 https://www.1training.org/blog/html-css-javascript-web-development

HOW DOES IT FIT INTO THIS CLASS?

A VERY QUICK HISTORY OF JAVASCRIPT

Brendan Eich created JavaScript in 1995 while he was at Netscape Communications Corporation, the creators of the legendary Netscape Navigator web browser. At the time, the Java coding language was rapidly gaining traction and Netscape Communications was working to make it available in Netscape Communicator.

However, Java was too large and too complex to appeal to amateurs, scripters, and designers. In order to solve this problem, Netscape Communications contracted Brendan Eich to design a versatile programming language that could speed up web development and serve as a scripting companion for Java.

And yet, instead of being just a Scheme for Netscape Navigator, JavaScript went on to become something much bigger.

Brendan Eich went on to co-found of the Mozilla project, the Mozilla Foundation, and the Mozilla Corporation.

1 https://www.w3.org/People/howcome/p/cascade.html

GETTING STARTED

Just like with CSS, you have to *link* your JavaScript files within your HTML file to get it to render. This method is referred to as an external script. You will always plays these scripts at the BOTTOM of the page, directly ABOVE to `</body>` tag

```
    <script src="path/to/script.js">
```

You can also use a script tag/element to write JavaScript inline
(but don't do this because it can be a security risk)

```
    <script>
        function foo(){
            ...
        }
    </script>
```

MANIPULATING THE DOM

The Document Object Model (**DOM**) is the data representation of the objects that comprise the structure and content of a document on the web.

It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.

A web page is a document that can be either displayed in the browser window or as the HTML source. In both cases, it is the same document but the Document Object Model (DOM) representation allows it to be manipulated. As an object-oriented representation of the web page, it can be modified with a scripting language such as JavaScript.

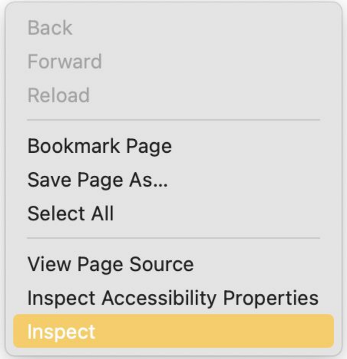
DEVTOOLS/WEB CONSOLE

All browser have some sort of developer toolkit to inspect the page.

- Chrome DevTools
- Firefox Web Console
- Safari Web Inspector
- Edge DevTools

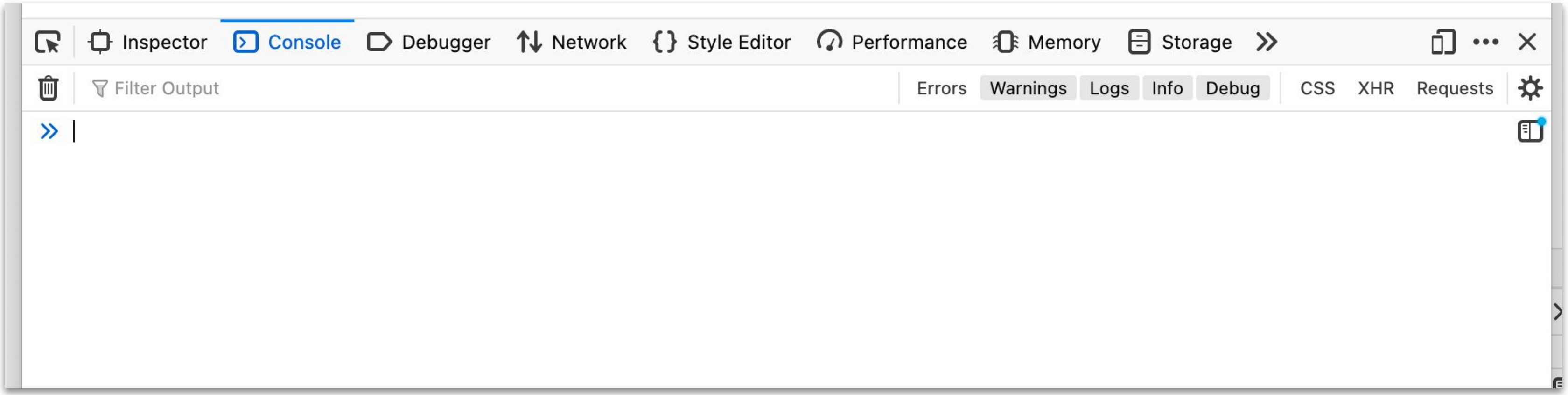
You can open these from a menu, right-clicking on a webpage and clicking “**Inspect**”.

Open it up and click the “Console” tab.



DEVTOOLS/WEB CONSOLE

Use the console in your browser (on a blank/new page) as a playground for the following walkthrough of code.



CODEPEN.IO

JS INTRO DEMO:

go to **https://maxx.link/javascript-demo**

JAVASCRIPT CRASH COURSE

VARIABLES

Variables are containers that store values. You start by declaring a variable with the variable type keyword followed by the name you give to the variable. There are three types:

- ~~`var` - Declares a variable, optionally initializing it to a value.~~
- `let` - Declares a block-scoped, local variable, optionally initializing it to a value.
- `const` - Declares a block-scoped, read-only named constant.

~~`var yourVar = false;`~~

```
let variable1 = “Maxx”

const myVariable = 2;
```

VARIABLES CAVEATS

A semicolon at the end of a line indicates where a statement ends. It is only required when you need to separate statements on a single line. *However, it's good practice to have semicolons at the end of each statement.*

You can name a variable nearly anything, but there are some restrictions.

JavaScript is case sensitive. This means `myVariable` is **NOT** the same as `myvariable`. If you have problems in your code, check the case!

VARIABLES TYPES

Note that variables may hold values that have different data types:

Type	Description	Example
String	A sequence of text, enclosed in quotes	let myVariable = 'Bob';
Number	An integer. No quotes!	let myVariable = 10;
Boolean	True/False value. No quotes!	let myVariable = true;
Array	This is a structure that allows you to store multiple values in a single reference.	let myVariable = [1,'Bob','Steve',10];
Object	This is a collection of properties	

BUT WHY VARIABLES?

OPERATORS

An operator is a mathematical symbol that produces a result based on two values (or variables).

Operator	Explanation	Symbol(s)
Addition	Add two numbers or combine strings	+
Subtraction, Multiplication, Division	Basic math.	- * /
Assignment	This assigns a value to a variable	=
Equality	This performs a test to see if two values are equal. It returns a true or false.	===
Not, Does-not-equal	This tests whether two values are not equal.	!, !==

CONSOLE DEMO

VARIABLES + OPERATORS

ASSIGNMENT SELECTORS

To set HTML elements as variables in JavaScript to select them. This is very similar to how CSS selectors work. There are multiple methods:

Document.getElementById()

```
    var element = document.getElementById(id);
```

Document.querySelector()

```
    let element = document.querySelector(selectors);
```

MANIPULATING CLASS NAMES

Classnames from elements can be queried using the Element.classList property.

Although the classList property itself is read-only, you can modify its associated DOMTokenList using the add(), remove(), replace(), and toggle() methods.

HTML:

```
<div id=”title” class=”red”>Hello World</div>
```

JavaScript:

```
const div = document.getElementById(“title”);
console.log(div.classList);
// expected: “title”
div.classList.toggle(“blue”);
console.log(div.classList);
// expected: “title blue”
```

CONDITIONALS

Conditionals are code structures used to test if an expression returns true or not. A very common form of conditionals is the `if ... else` statement.

```
let iceCream = 'chocolate';
if(iceCream === 'chocolate') {
  alert('Yay, I love chocolate ice cream!');
} else {
  alert('Awwww, but chocolate is my favorite...');
}
```

FUNCTIONS (BUILT IN)

Functions are a way of packaging functionality that you wish to reuse. Some functions are built into the browser:

```
    alert('hello!');
```

This function makes a pop-up box appear inside the browser window, but we need to give it a string as an argument to tell the function what message to display.

FUNCTIONS (CUSTOM)

You can also define your own functions.

```
function multiply(num1,num2) {
    let result = num1 * num2;
    return result;
}

multiply(4, 7);
```

*Note: The **return** statement tells the browser to return the result variable out of the function so it is available to use.*

EVENTS

Real interactivity on a website requires event handlers. These are code structures that listen for activity in the browser, and run code in response.

The most obvious example is handling the **click event**, which is fired by the browser when you click on something with your mouse.

```
document.querySelector('html').onclick = function() {
    alert('Ouch! Stop poking me!');
}
```

ASSIGNMENT 05: BAREBONES HTML SITE, SCRIPTED

- Continue with your previous poem site from A02/3/3.1
- Add a JavaScript file, link it to your HTML and add a function to toggle a dark/light class to the body element
- Add a button to your page that when clicked with make your site in “dark mode”
- Add a CSS class that styles your page accordingly
- Use GitHub Pages to host your project
- Check-in on Wednesday, Final due Friday, Sept 15, Beginning of class

Late submissions will be deducted -10% of the overall grade for each day the assignment is late unless arrangements are made with Maxx or Bryan.