

# Front End Development

JavaScript

# JavaScript

**We will not thoroughly cover JavaScript syntax**

If you haven't used JavaScript before, you are expected to learn the basic syntax and concepts on your own (or stop by office hours)

Topics such as variables, conditionals, loops, functions, and data structures won't be explicitly covered in lecture

As with HTML/CSS, I recommend W3 Schools for a good tutorial/reference

<https://www.w3schools.com/js/default.asp>

# Front End JavaScript

We've seen the basic building blocks for web pages in HTML and CSS

This allows us to build what I call "poster sites" - The equivalent of making a poster that people can only view

In this course we want to make web apps that allow our users to interact with our content

To start this interaction, we need JavaScript (Specifically, ECMAScript)

# Front End JavaScript

HTML and CSS are not programming languages (well.. technically)

JavaScript is a programming language

Javascript allows us to add code to our page

When a user visits your site their browser will:

1. Download your JavaScript code
2. Run your code on their machine

What are the security implications of this?

# Front End JavaScript - Security

There are restrictions to what JavaScript can do in the browser:

- Cannot read/write files

- Cannot access other tabs or windows

- Cannot access data (ex. cookies, local storage) from other sites

- Vulnerabilities are still exposed/patched

With that, let's see our first script

# Front End JavaScript

```
var myDiv = document.getElementById("myDiv");  
myDiv.innerHTML = "Content added from JavaScript";
```

Here we call the `document.getElementById` method which returns the elements with the provided id

The element is an object with a key “innerHTML” whose value is the content (between the open and close tags) of the element

We'll save this code in a file named “script”

# Front End JavaScript

```
var myDiv = document.getElementById("myDiv");  
  
myDiv.innerHTML = "Content added from JavaScript";
```

```
<!DOCTYPE html>  
<html lang="en">  
  
  <head>  
    <meta charset="UTF-8">  
    <title>CSE312 - First Page</title>  
  </head>  
  
  <body>  
    <h1>First Web Page</h1>  
    <p>My content</p>  
    <div id="myDiv"></div>  
    <script src="script.js"></script>  
  </body>  
</html>
```

We “import” our javascript code by adding a script element at the bottom of the body element with a src (source) attribute containing our JavaScript filename

This runs our script once the body is loaded

# Front End JavaScript

The script runs when the body loads and sets the content of “myDiv” resulting in this page (CSS removed)

**First Web Page**

My content

Content added from JavaScript



# JavaScript - Events

We saw a few examples of running a JS script to change the page, but what about reacting to the user? That's what we really want after all

Let's explore browser events and write code that can react to them

See here for a list of events - [https://www.w3schools.com/js/js\\_events\\_examples.asp](https://www.w3schools.com/js/js_events_examples.asp)

# Front End JavaScript

```
<div class="green wide" id="myDiv" onmouseenter="makeBlue(this)"  
onmouseleave="makeGreen(this)"></div>
```

```
function makeBlue(elem) {  
    elem.style.setProperty("color", "#000077")  
}  
  
function makeGreen(elem) {  
    elem.style.setProperty("color", "#007700")  
}
```

We add a few more attributes to our div to run JavaScript functions on the mouse enter and mouse leave events

The value of these attributes is any valid JavaScript

To keep our project organized, we'll call functions that are defined in a separate file

# Front End JavaScript

```
<div class="green wide" id="myDiv" onmouseenter="makeBlue(this)"  
onmouseleave="makeGreen(this)"></div>
```

```
function makeBlue(elem) {  
    elem.style.setProperty("color", "#000077")  
}  
  
function makeGreen(elem) {  
    elem.style.setProperty("color", "#007700")  
}
```

We pass the element itself as an argument to the functions

In the function, we change one of the style properties of the element

We can run any valid JavaScript here so what you can do is only limited by what you can conceive and code

# Browser Extensions

- Extensions allow additional JavaScript code to be injected into a website
- Useful when we want to modify a site without opening the browser console and typing commands after every page load
- Extensions automate this process

# Browser Extensions

- Extension have some limitations
  - Cannot access variables or functions from the page
  - More specifically: Extensions and page source run in two different environments
- Extension can modify the DOM (HTML elements)

# Activity

## **Write a web page using HTML, CSS, and Javascript**

To meet the requirements you must read documentation (W3 Schools recommended) which will be a theme throughout the course. Your page must contain:

- An HTML file with with at least one list and one table
- An external CSS file that adds style to at least 3 of your HTML elements and uses at least one class
- An external JavaScript file that reacts to at least one event that can be triggered by the user