BORDERS:NONE
JavaScript
March 4, 2025

Today’s Objectives

- Review and strengthen core JavaScript concepts
- Understand how JavaScript interacts with HTML
- Explore variables, data types, DOM interaction, arrays, and objects
- Apply concepts through practical code examples

What is the DOM?

- Document Object Model
- A programming interface for web documents
- Represents the page as a tree of objects
- Allows JavaScript to interact with the HTML content

JavaScript and HTML Connection

- JavaScript makes web pages dynamic and interactive
- Executed by the browser's JavaScript engine
- Can manipulate HTML elements in real-time
- Responds to user events (clicks, key presses, etc.)

Including JavaScript in HTML

```
<!-- Internal JavaScript -->
<script>
  // JavaScript code here
  document.getElementById("example").innerText = "Hello!";
</script>

<!-- External JavaScript -->
<script src="script.js"></script>
```

Script Placement

Scripts in <head>: Load before the body content

- May cause delays if scripts are large
- Good for initialization code

Scripts before </body>: Load after the page content

- Faster initial page rendering
- Recommended for most scripts

Variables: What are they?

- Named containers that store data
- Can be updated and reused throughout code
- Must be declared before use
- Have different scopes based on declaration method

Understanding Constants

```
// Constants cannot be reassigned after declaration
const PI = 3.14159;
const USERNAME = "admin";
const SETTINGS = { theme: "dark", notifications: true };

// These would cause errors:
// PI = 3.14;        // Error: Assignment to constant variable
// USERNAME = "user"; // Error: Assignment to constant variable

// However, for objects and arrays, the contents can be modified:
SETTINGS.theme = "light";  // This works! (changing a property)
SETTINGS.language = "en";  // This works! (adding a property)

// Just can't reassign the entire object:
// SETTINGS = { theme: "blue" };  // Error!
```

Variable Declaration

```
// Three ways to declare variables
var oldWay = "I'm an older variable";
let newWay = "I'm the modern approach";
const neverChanges = "I cannot be reassigned";

// Reassignment
oldWay = "Changed!";    // Allowed
newWay = "Changed!";    // Allowed
// neverChanges = "Changed!";  // Error!
```

```
let my_variable;
const my_constant;
let some_variable;

my_variable = 10;
some_variable = 15; // Error
```

Variable Scope: **var** vs **let**

```
// Function scope with var
function example() {
  var x = 10;
  if (true) {
    var x = 20;  // Same variable!
    console.log(x);  // 20
  }
  console.log(x);  // 20
}

// Block scope with let
function example2() {
  let y = 10;
  if (true) {
    let y = 20;  // Different variable!
    console.log(y);  // 20
  }
  console.log(y);  // 10
}
```

Basic Data Types

- **String**: Text within quotes (`"Hello"`, `'World'`)
- **Number**: Integers and decimals (`42`, `3.14`)
- **Boolean**: `true` or `false`
- **Null**: Intentional absence of value
- **Undefined**: Variable declared but not assigned

I created a bucket (variable) named my_new_bucket, and I put something in it.

1. Undefined
2. String
3. We don’t know → Ask more information

Type Coercion

```
// JavaScript automatically converts types when needed
let num = 5;
let str = "10";

console.log(num + str);      // "510" (number converted to string)
console.log(str - num);      // 5 (string converted to number)
console.log("5" == 5);       // true (values are equal after conversion)
console.log("5" === 5);      // false (different types)
```

Checking Variable Types

```
let name = "John";
let age = 30;
let isActive = true;
let person = { name: "John" };
let numbers = [1, 2, 3];

console.log(typeof name);      // "string"
console.log(typeof age);       // "number"
console.log(typeof isActive);  // "boolean"
console.log(typeof person);    // "object"
console.log(typeof numbers);   // "object" (arrays are objects)
```

Finding Elements in the DOM

```
// By ID (returns a single element)
let element = document.getElementById("example");

// By CSS selector (returns first matching element)
let element2 = document.querySelector("#example");
let paragraph = document.querySelector("p");
let special = document.querySelector(".special");

// Getting multiple elements
let allParagraphs = document.querySelectorAll("p");
```

```
let my_data = document.querySelector(“#example”)
```

<input id=”example” class=”colorful” target=”somewhere” value=”STUFF” />

Modifying Content

```
// Get the element
let paragraph = document.getElementById("example");

// Change text content
paragraph.innerText = "New text content";  // Text only

// Change HTML content
paragraph.innerHTML = "Text with <strong>bold</strong>";  // Can include HTML
```

Modifying Attributes and Styles

```
// Get the element
let image = document.getElementById("myImage");

// Change attributes
image.src = "new-image.jpg";
image.alt = "New description";

// Change CSS styles
paragraph.style.color = "blue";
paragraph.style.fontSize = "18px";
paragraph.style.backgroundColor = "yellow";
```

Arrays: Ordered Collections

```
// Creating arrays
let fruits = ["apple", "banana", "orange"];
let mixed = [1, "two", true, null];

// Accessing elements (zero-indexed)
console.log(fruits[0]);  // "apple"
console.log(fruits[2]);  // "orange"

// Getting array length
console.log(fruits.length);  // 3
```

Array Methods

```
let colors = ["red", "green"];

// Adding elements
colors.push("blue");          // Adds to end: ["red", "green", "blue"]
colors.unshift("yellow");     // Adds to beginning: ["yellow", "red", "green", "bl

// Removing elements
let lastColor = colors.pop(); // Removes from end, returns "blue"
let firstColor = colors.shift(); // Removes from beginning, returns "yellow"

// Other useful methods
colors.join(", ");            // "red, green"
colors.includes("red");       // true
```

Objects: Key-value pairs

```
// Creating an object
let person = {
  name: "John",
  age: 30,
  isStudent: false,
  address: {
    city: "New York",
    zip: "10001"
  }
};
```

Accessing Object Properties

```
let person = {
  name: "John",
  age: 30,
  "favorite color": "blue"
};

// Dot notation
console.log(person.name);  // "John"
console.log(person.age);   // 30

// Bracket notation
console.log(person["name"]);           // "John"
console.log(person["favorite color"]); // "blue" (required for keys with spaces)

// With variables
let key = "age";
console.log(person[key]);  // 30
```

Code Walkthrough

- Break down last week's code step by step
- Relate it back to the concepts we've learned
- Focus on:
  - DOM selection and manipulation
  - Variable types and objects
  - Function behavior
  - Arrays and methods

Next Steps

- Complete the skills review
- Submit your answers via Slack DM to Kiley or Mehmet
- Review additional resources on our GitHub
- Prepare questions for next week
    - Submit via Github!