**JavaScript In HTML**

**Linking External JavaScript to Your HTML**

This lesson demonstrates the foundational process of incorporating JavaScript into an HTML document by linking to an external JavaScript file. This separation of concerns is a best practice in web development, keeping your structure (HTML) distinct from your behaviour (JavaScript). We'll use a simple HTML page containing a button as our starting point.

Our initial HTML file, index.html, looks like this:

<!DOCTYPE html>

<html>

<head>

<title>Buy me a coffee</title>

</head>

<body>

<button id="connectButton">Connect</button>

</body>

</html>

Our goal is to eventually add interactive logic to the connectButton element using JavaScript. While you *can* embed JavaScript directly within <script> tags inside your HTML, managing behaviour in separate files is generally preferred for organization and maintainability.

**Creating the External JavaScript File**

First, we need to create the file that will hold our JavaScript code.

1. Create a new file in the same directory as your index.html file.
2. Name this file index-js.js.

*Note on Naming:* The name index-js.js might seem slightly redundant. This specific naming convention is used here because we anticipate transitioning to TypeScript (.ts files) later. This naming helps distinguish the plain JavaScript version during that transition.

**Linking the JavaScript File to HTML**

Now, we need to tell the index.html file to load and execute the code contained within index-js.js. We do this using the <script> tag within the HTML's <body>. It's common practice to place this tag just before the closing </body> tag. This ensures that the HTML elements are loaded and available in the Document Object Model (DOM) before the script attempts to interact with them.

Add the following line inside the <body> of your index.html file, right before the </body> tag:

<script src="./index-js.js" type="module"></script>

</body>

</html>

Let's break down this tag:

* <script>**:** The standard HTML element for embedding or referencing executable scripts.
* src="./index-js.js"**:** The src (source) attribute specifies the path to the external script file. The ./ indicates that the file (index-js.js) is located in the same directory as the current HTML file (index.html).
* type="module"**:** This attribute indicates that the script should be treated as a modern JavaScript module. The specific implications of this aren't critical for this initial step, so don't worry about the details for now; just include it as shown.

When the browser parses the HTML and encounters this <script src="..."> line, it fetches the content of index-js.js and executes the JavaScript code found within it, effectively injecting that code's behaviour into the context of the HTML page.

**Verifying the Connection**

To confirm that our external JavaScript file is correctly linked and being executed, we can add a simple command to it that logs a message to the browser's developer console.

1. Open the index-js.js file.
2. Add the following line of code:

console.log("hi");

1. Save the index-js.js file.
2. Open the index.html file in your web browser.
3. Open your browser's developer tools (often by right-clicking on the page and selecting "Inspect" or "Inspect Element") and navigate to the "Console" tab.
4. Refresh the page if necessary.

You should see the text "hi" printed in the console, likely followed by a reference to the file and line number where the command originated (e.g., index-js.js:1). This output confirms that the browser successfully loaded and executed the code from your external JavaScript file.

If you add more console.log("hi"); lines to index-js.js, save, and refresh the browser, you will see multiple "hi" messages logged, further demonstrating the execution flow.

**Conclusion**

You have successfully linked an external JavaScript file (index-js.js) to your HTML document (index.html). This setup establishes the connection needed to start adding dynamic behaviour and interactivity to your web page elements, such as making the connectButton perform an action when clicked, which will be covered in subsequent steps. Using external files helps keep your project organized and makes your code easier to manage as it grows.