**Hands-on Exercises - JS**

1. To create a dynamic webpage that changes based on user interaction, you can use HTML, CSS, and JavaScript. Here’s a simple example of a webpage that changes the background color based on user input:

**HTML (index.html):**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Dynamic Webpage</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="content">

<h1>Change Background Color</h1>

<label for="colorPicker">Pick a color: </label>

<input type="color" id="colorPicker">

</div>

<script src="script.js"></script>

</body>

</html>

**CSS (styles.css):**

css

CopyEdit

body {

font-family: Arial, sans-serif;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

transition: background-color 0.5s ease;

}

.content {

text-align: center;

}

h1 {

color: #333;

}

**JavaScript (script.js):**

javascript

CopyEdit

const colorPicker = document.getElementById('colorPicker');

colorPicker.addEventListener('input', (event) => {

// Change the background color based on user input

document.body.style.backgroundColor = event.target.value;

});

**How it works:**

1. **HTML**: Contains a color input element where the user can pick a color.
2. **CSS**: Styles the page, and transition is used to smoothly change the background color.
3. **JavaScript**: Listens for the input event on the color picker. When the user selects a color, it updates the background color of the body.

**Result:**

When the user picks a color using the color picker, the background color of the webpage will change dynamically.

This is just one example of a dynamic interaction; you can expand this to include more features, like changing text, showing/hiding elements, or adding animations based on other types of user interaction.

**Practice**

1. **Case Study 1: Change the Background Color Based on User Selection**

**Scenario:**

The user can click a button to change the background color of the webpage.

**Code Example:**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Dynamic Webpage - Background Color Change</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

.color-btn {

padding: 10px 20px;

font-size: 16px;

cursor: pointer;

margin: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

}

.color-btn:hover {

background-color: #45a049;

}

</style>

</head>

<body>

<h1>Choose a Background Color</h1>

<button class="color-btn" onclick="changeColor('lightblue')">Light Blue</button>

<button class="color-btn" onclick="changeColor('lightgreen')">Light Green</button>

<button class="color-btn" onclick="changeColor('lightcoral')">Light Coral</button>

<script>

// Function to change the background color

</script>

</body>

</html>

**Explanation:**

* **HTML**: Provides the basic structure with buttons for different color choices.
* **CSS**: Styles the buttons to make them visually appealing.
* **JavaScript**: The changeColor() function updates the background color of the webpage when the user clicks one of the buttons. It modifies the style.backgroundColor of the body element.

**Case Study 3: Show a Greeting Based on Time of Day**

**Scenario:**

The webpage greets the user based on the current time when the page loads.

**Code Example:**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Dynamic Greeting</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

.greeting {

font-size: 24px;

font-weight: bold;

}

</style>

</head>

<body>

<h1 class="greeting" id="greeting"></h1>

<script>

// Function to determine the greeting based on the time of day

function setGreeting() {

}

// Call the function to set the greeting

setGreeting();

</script>

</body>

</html>

**Explanation:**

* **HTML**: Contains an empty <h1> tag with an ID of "greeting" where the greeting message will appear.
* **CSS**: Styles the greeting message to make it bold and larger.
* **JavaScript**: The setGreeting() function gets the current hour of the day and sets the greeting message accordingly (Good Morning, Good Afternoon, or Good Evening).

**Case Study 4: Toggle Content Visibility**

**Scenario:**

The user can toggle between showing and hiding content on the page.

**Code Example:**

html

CopyEdit

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Toggle Content Visibility</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 50px;

}

.toggle-btn {

padding: 10px 20px;

font-size: 16px;

cursor: pointer;

margin: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 5px;

}

.toggle-btn:hover {

background-color: #45a049;

}

.content {

display: none;

margin-top: 20px;

font-size: 18px;

}

</style>

</head>

<body>

<h1>Toggle Content Visibility</h1>

<button class="toggle-btn" onclick="toggleContent()">Show/Hide Content</button>

<div id="content" class="content">

<p>This is some hidden content. You can toggle its visibility using the button above.</p>

</div>

<script>

// Function to toggle the visibility of content

</script>

</body>

</html>

**Explanation:**

* **HTML**: Contains a button to toggle visibility and a <div> with the content to show/hide.
* **CSS**: Initially hides the content with display: none and styles the button.
* **JavaScript**: The toggleContent() function changes the display property of the content <div> from none to block, allowing it to toggle between visible and hidden.

**Key Concepts:**

* **DOM Manipulation**: In all cases, JavaScript interacts with the DOM (Document Object Model) to change the content or style of the page in response to user input.
* **User Interaction**: The page updates dynamically based on what the user clicks or the time of day, making the webpage interactive and responsive.

These examples demonstrate real-time case studies where the content of a webpage changes based on user actions. You can extend these concepts to more complex applications like user forms, live data updates, or interactive dashboards.