
Pre-Lab Exercises – Experiment 5

DOM Basics & Preparation for To-Do List Application

Part A: DOM Fundamentals (Conceptual)

Exercise 1: Understanding DOM

Define DOM:
The DOM (Document Object Model) is a programming interface that represents an HTML document as a tree structure of objects. It allows JavaScript to access, modify, add, or remove HTML elements dynamically.

Simple DOM Tree:

html

└── body

  ├── h1

  └── p

Exercise 2: DOM Selection Methods

Purpose of:

- **getElementById()**
  Selects a single HTML element using its unique id.
- **querySelector()**
  Selects the first element that matches a given CSS selector.
- **querySelectorAll()**
  Selects all elements that match a given CSS selector and returns a NodeList.

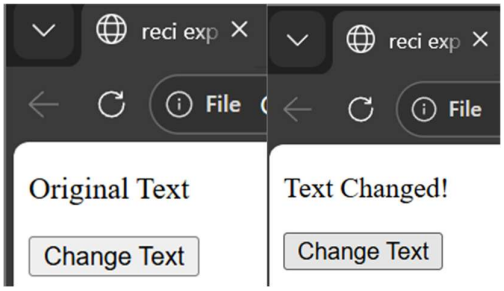
Part B: Hands-On DOM Practice

Exercise 3: Accessing Elements

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="para">Original Text</p>
<button onclick="changeText()">Change Text</button>

<script>
function changeText() {
  document.getElementById("para").innerText = "Text Changed!";
}
</script>

</body>
</html>
```

Output:

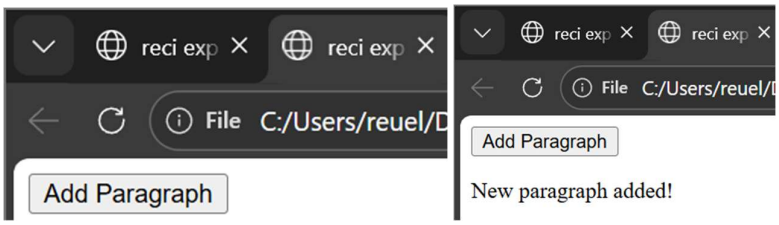


Exercise 4: Creating Elements Dynamically

Code:

```
<!DOCTYPE html>
<html>
<body>

<button onclick="addParagraph()">Add Paragraph</button>

<script>
function addParagraph() {
  let p = document.createElement("p");
  p.innerText = "New paragraph added!";
  document.body.appendChild(p);
}
</script>

</body>
</html>
```

Output:

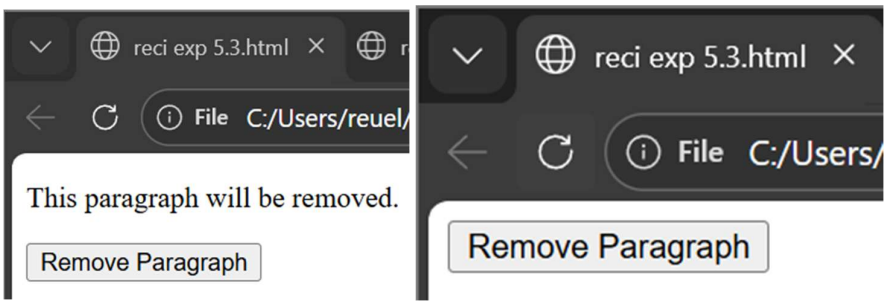


Exercise 5: Removing Elements

Code:

```
<!DOCTYPE html>
<html>
<body>

<p id="removeMe">This paragraph will be removed.</p>
<button onclick="removePara()">Remove Paragraph</button>

<script>
function removePara() {
  document.getElementById("removeMe").remove();
}
</script>

</body>
</html>
```

Output:



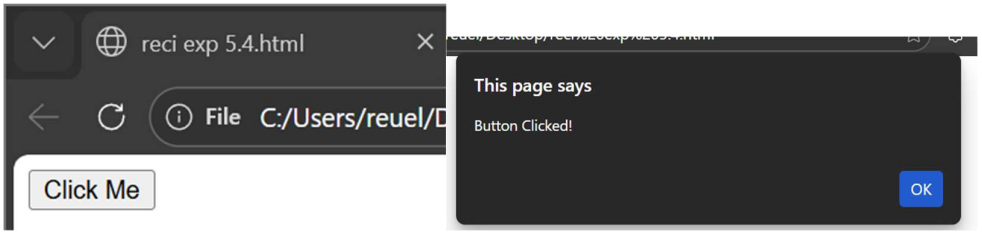
Part C: Event Handling Practice

Exercise 6: Event Listener

Code:

```
<!DOCTYPE html>
<html>
<body>

<button id="btn">Click Me</button>

<script>
document.getElementById("btn").addEventListener("click", function() {
  alert("Button Clicked!");
});
</script>

</body>
</html>
```

Output:

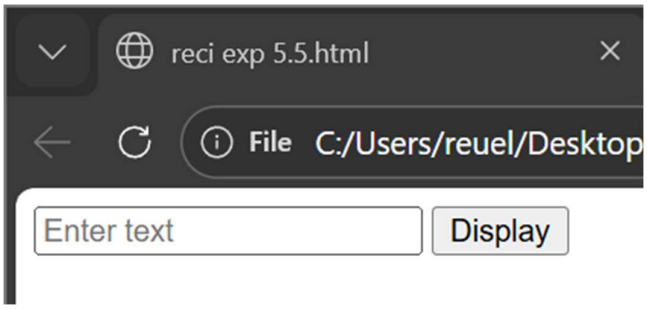


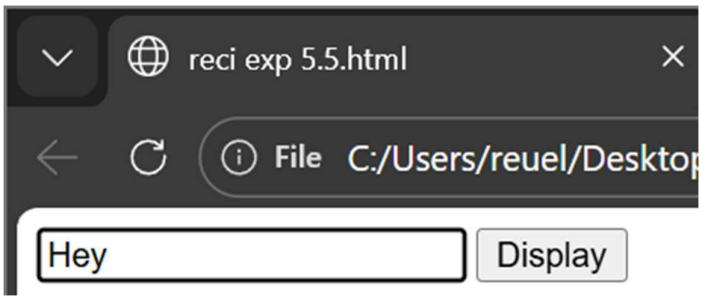
Exercise 7: Input Handling

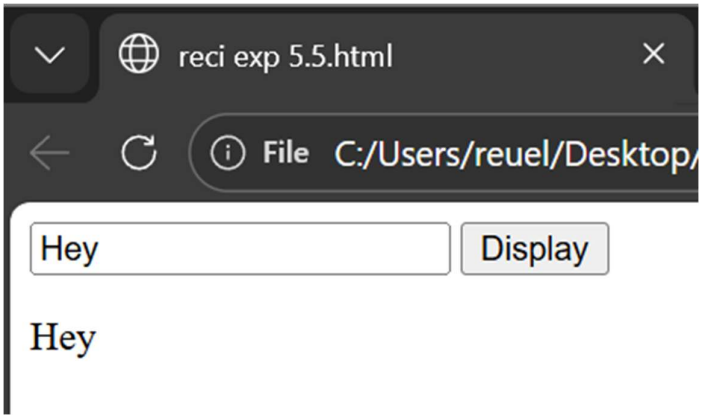
Code:

```
<!DOCTYPE html>
<html>
<body>

<input type="text" id="userInput" placeholder="Enter text">
<button onclick="showText()">Display</button>
<p id="output"></p>

<script>
function showText() {
  let text = document.getElementById("userInput").value;
  document.getElementById("output").innerText = text;
}
</script>

</body>
</html>
```

Output:







Part D: Logic Preparation for To-Do List

Exercise 8: Simple Task Addition Logic

Code:

```
<!DOCTYPE html>
<html>
<body>
<input type="text" id="taskInput" placeholder="Enter task">
<button onclick="addTask()">Add Task</button>

<ul id="taskList"></ul>

<script>
function addTask() {
  let task = document.getElementById("taskInput").value;
  let li = document.createElement("li");
  li.innerText = task;
  document.getElementById("taskList").appendChild(li);
}
</script>

</body>
</html>
```

Output: