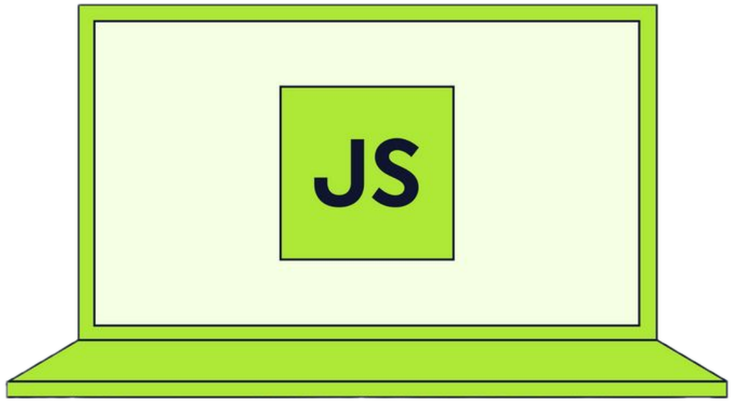
Newton School of Technology

The Complete Javascript Course



@newtonschool

Lecture 23: Dynamic Web Pages with JavaScript

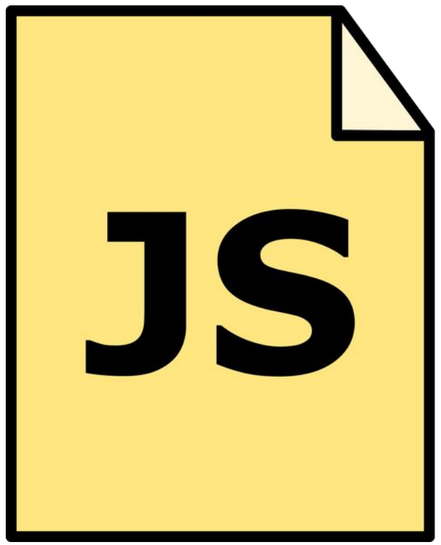
-Ajay Sharma

JS

Key Take away from this Lecture

Newton School of Technology

- Quick review on handling clicks
- Making things appear with JS
- Storing data using arrays
- Build your first mini project



Recap: How Button Clicks Work

Newton School of Technology

Inline Events

```
1 <button onclick="sayHello()">Click Me</button>
```

Inline event handler attribute

Recap: How Button Clicks Work

Newton School of Technology

Inline Events

```
1 <button onclick="sayHello()">Click Me</button>
```

Inline event handler attribute

Event: click (triggered when the element is clicked)

Event attribute: onclick

Event handler: fun() — the function that runs when the event occurs

Recap: How Button Clicks Work

Newton School of Technology

Inline Events

```
1 <button onclick="sayHello()">Click Me</button>
```

Why we call function here :

1. Inline event handlers act like you are writing code inside an attribute.
2. The browser expects an expression to execute, not a function reference (name of function).

Recap: How Button Clicks Work

Newton School of Technology

JavaScript property event handlers

```
let a = document.querySelector("#btn");
a.onclick = fun;
```

Why we call function here :

1. a.onclick expects a **function reference**(function name), not a **function call**.
2. If you write **fun()**, it will run **immediately** and assign its return value (usually undefined) to onclick.

Recap: How Button Clicks Work

Newton School of Technology

Inline Events

```html
<button onclick="sayHello()">Click Me</button>
```

Event Listeners—The Professional Way

```javascript
const button = document.getElementById("myButton");
button.addEventListener("click", function() {
        alert("Hello!");
});
```

Newton School of Technology

How to add multiple event listeners to a single button click.

One Button, Many Event Listeners

Newton School of Technology

When you attach multiple event listeners of the same type (for example, multiple "click" listeners) to the same HTML element using **addEventListener( )**,

```html
<html>
  <button id="myButton">Click Me!</button>
</html>
<script>
    const button = document.getElementById("myButton");
    button.addEventListener("click", function() { alert("First!");});
    button.addEventListener("click", function() { alert("Second!");});
    button.addEventListener("click", function() { alert("Third!");});
</script>
```

Result: One click, multiple actions!

- They are called in the order you registered (attached) them in nearly all modern browsers.
- Useful when handling logic separately.

Using multiple event listeners on one button click

Newton School of Technology

```html
<html>
  <button id="myButton">Click Me!</button>
  <p id="message"></p>
</html>
<script>
    const button = document.getElementById("myButton");
    const message = document.getElementById("message");
    // First listener: show alert
    button.addEventListener("click", function() {
      alert("You clicked the button!");
    });
    // Second listener: change button color and update text
    button.addEventListener("click", function() {
      button.style.backgroundColor = "lightgreen";
      message.innerText = "The button was clicked and color changed!";
    });
  </script>
```

- Clicking the button triggers both actions—first showing the alert, then changing the color.
- Order matters: Listeners run in the order you attached them

The 3 Magic Words 🪄

Newton School of Technology

To add something new to the page:

1. createElement( ) → make it
2. innerText → write on it
3. appendChild( ) → place it on the screen

Once you've learned all three concepts, you'll be able to understand this code

```
const p = document.createElement("p");
p.innerText = "Hello!";
document.body.appendChild(p);
```

Understanding createElement( )

Newton School of Technology

Purpose:

createElement() is used to make a new HTML element with JavaScript — like creating a building block you can add to your webpage dynamically.

Syntax:

```
const element = document.createElement("tagName");
```

creating a paragraph tag

```
const para = document.createElement("p");
//Create a <p> element
```

Understanding innerText

Newton School of Technology

Purpose:

innerText is used to set or get the text inside an HTML element using JavaScript.

Syntax:

```
element.innerText = "Your new text here";
```

To set text inside p tag

```
const para = document.createElement("p");  // Step 1: Create a <p> element
para.innerText = "I was created with JS!"; // Step 2: Add content
```

Understanding appendChild( )

Newton School of Technology

Purpose:

appendChild() is used to add (or "attach") a new element to the end of an existing parent element in the DOM.

Syntax:

```
parentElement.appendChild(childElement);
```

Result on the Page

```
const para = document.createElement("p");    // Step 1: Create a <p> element
para.innerText = "This is a new paragraph."; // Step 2: Add some text
document.body.appendChild(para);             // Step 3: Add to the body
```

🔁 Remember: Create → Write → Stick!

Newton School of Technology

Coding Time

Newton School of Technology

Manipulating Arrays

Websites Need “Memory” = Arrays 🧠

Newton School of Technology

The browser shows things, BUT we also need to remember them.

We can use the Array to store multiple value in under one name.

Arrays = lists you store and use later

```
1 const names = [ ];
2 names.push("Alice");
3 names.push("Bob");
4 console.log(names); // ["Alice", "Bob"]
```

Arrays in Action: Store User Input

Newton School of Technology

```
const tasks = [];
button.addEventListener("click", () => {
  const task = input.value;
  tasks.push(task);
});
```

🔎 **Where we can use Arrays?**

- Count items
- Filter, delete, search
- Data to Save later

Arrays in Action: Store User Input

Newton School of Technology

```html
<html>
  <input id="taskInput" type="text" placeholder="Enter a task">
  <button id="addTask">Add Task</button>
</html>
<script>
const tasks = [];
button.addEventListener("click", () => {
  const task = input.value;
  tasks.push(task);
});
</script>
```

🔎 Where we can use Arrays?

- Count items
- Filter, delete, search
- Data to Save later

Newton School of Technology

Coding Time

Mini Project: “Name Collector”

Newton School of Technology

Create a small web app where users can type a name into an input box:

- User types a name and clicks “Add Name”
- The name is added to a list on the page
- Each name stored in an array
- Input field clears automatically
- Total count updates

```
<html>
    <input id="nameInput" />
    <button id="addName">Add</button>
    <ul id="nameList"></ul>
    <p id="counter">Total: 0</p>
</html>
```

Mini Project: “Name Collector” –Solution

Newton School of Technology

```html
<html>
    <input id="nameInput" />
    <button id="addName">Add</button>
    <ul id="nameList"></ul>
    <p id="counter">Total: 0</p>
</html>
<script>
  const names = [];
  document.getElementById("addName").addEventListener("click", () => {
  const input = document.getElementById("nameInput");
  const name = input.value;
  names.push(name);
  const li = document.createElement("li");
  li.innerText = name;
  document.getElementById("nameList").appendChild(li);
  document.getElementById("counter").innerText = "Total: " + names.length;
  input.value = "";
});
</script>
```

What You Learned Today

Newton School of Technology

You can now:

- Create webpage content instantly.
- Respond to events like clicks.
- Remember things using arrays.
- Use createElement, appendChild, and addEventListener.
- Build your first dynamic project.
- Use multiple listeners on one element!

Newton School of Technology

Quiz Time

References

Newton School of Technology

Dynamic DOM Creation & Manipulation

https://dev.to/hcoco1/javascript-dynamic-list-the-dom-manipulation-10c5

https://www.tutorialspoint.com/how-to-dynamically-create-new-elements-in-javascript

DOM Methods: createElement, appendChild, prepend

https://www.w3schools.com/jsref/met_node_appendchild.asp

https://savvy.co.il/en/blog/complete-javascript-guide/javascript-appendchild/

Arrays & JavaScript Memory Management

https://www.w3schools.com/js/js_arrays.asp

https://www.freecodecamp.org/news/javascript-array-tutorial-array-methods-in-js/

Event Handling with addEventListener( )

https://www.w3schools.com/js/js_htmldom_eventlistener.asp

https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Scripting/Events

Multiple Event Listeners on One Element

https://stackoverflow.com/questions/11845678/adding-multiple-event-listeners-to-one-element

Feedback

Newton School of Technology

Now it's time to give your valuable feedback

Newton School of Technology

Thanks for watching!