Intro to JavaScript

JS

Who is me?

Muhammad Ikhsan Effendy

- Senior IS student
- Research Assistant at Media-Tech Lab
- Software Engineer at Ikonsultan Inovatama
- Currently working on my final project on big data mining
- Loves cats & dogs

*picture of me eating outside my habitat

Before we go

- A code editor like VSCode installed on your computer.
- Node.js if you want to run JavaScript without a browser.
- An online compiler like Programiz if you don't want to install anything.
- You can also see the output from console.log in browser by Right Click › Inspect › Console
- For more tutorial, visit W3 Schools JavaScript Tutorial

Optional:

- If you have time before class, clone this repository or ask devstub staff for the repo.
- Read the materials and try to do the task by your own. Ask me if you encounter any problems. You can find my contact on the last slide. Good luck!

What is JavaScript?

JavaScript is a coding language that makes websites interactive.
Without JavaScript a website would look like a boring newspaper.

Why JavaScript?

- Add interactivity (e.g., button clicks, animations, pop-ups)
- Modify web content dynamically (e.g., changing text, images, styles)
- Respond to user actions (e.g., typing, scrolling, clicking)
- Communicate with servers (e.g., fetching new content without reloading the page)
- Power modern web applications like Facebook, Instagram, and Google Maps

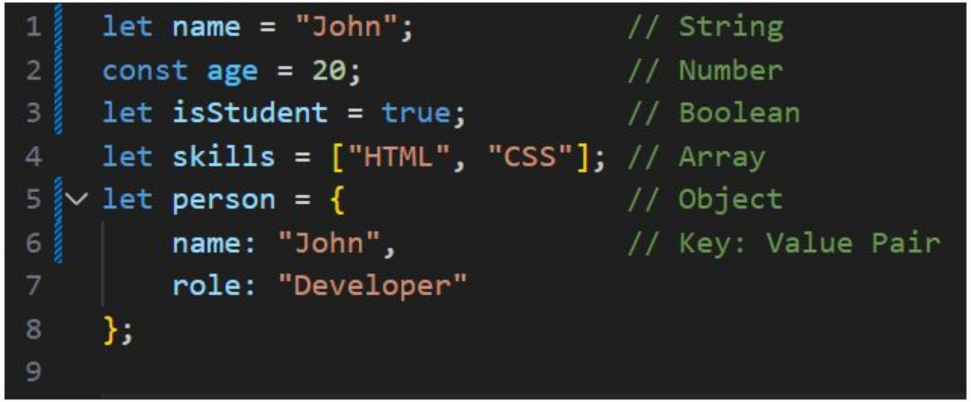
Variables & Data Types

Basic Data Types:

- String (store text or collection of characters)
- Number
- Boolean (true or false value)
- Array (collection of data)
- Object (structured data)

Variables:

- let - can be reassigned
- const - cannot be reassigned



```javascript
let name = "John";                        // String
const age = 20;                           // Number
let isStudent = true;                     // Boolean
let skills = ["HTML", "CSS"]; // Array
let person = {                            // Object
    name: "John",                         // Key: Value Pair
    role: "Developer"
};

```

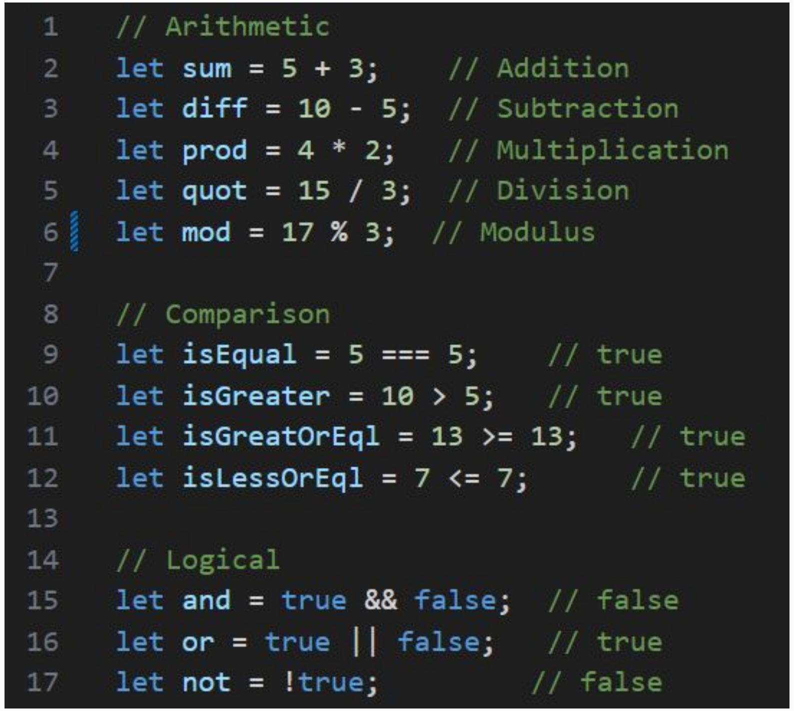
Source: data_type.js

Operators

- **Arithmetic:** Mathematical operations

Boolean Operators:

- **Comparison:** Compare two values
- **Logical:** Logical operation that returns boolean value
    - AND: return **true** if both value is **true**, otherwise **false**
    - OR: return **true** if one of two value is **true**
    - NOT: **reverse** the value to the opposite value



```
// Arithmetic
let sum = 5 + 3;     // Addition
let diff = 10 - 5;   // Subtraction
let prod = 4 * 2;    // Multiplication
let quot = 15 / 3;   // Division
let mod = 17 % 3;   // Modulus

// Comparison
let isEqual = 5 === 5;     // true
let isGreater = 10 > 5;    // true
let isGreatOrEql = 13 >= 13;    // true
let isLessOrEql = 7 <= 7;       // true

// Logical
let and = true && false;   // false
let or = true || false;    // true
let not = !true;          // false
```

Source: operators.js

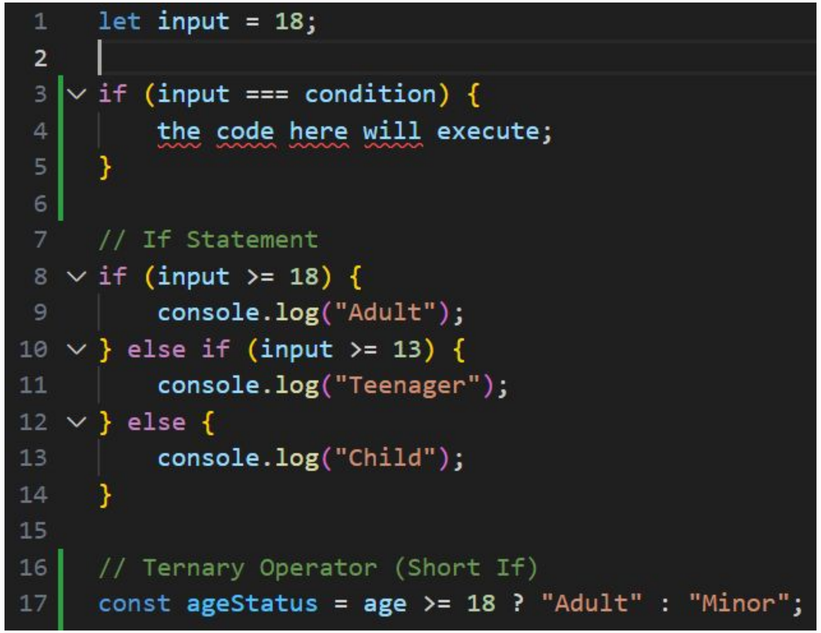
Conditional Statements

If Statements:

- Compare value of **input** with **condition**
- Can have multiple conditions (using else if)
- Can be nested

Ternary Operator:

- Shorthand for simple if/else
- Good for assigning value to variable



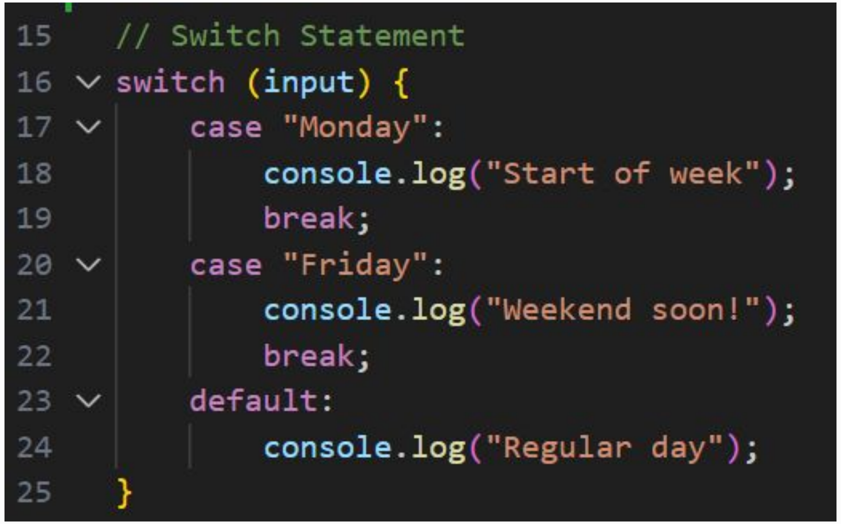
```
let input = 18;

if (input === condition) {
    the code here will execute;
}

// If Statement
if (input >= 18) {
    console.log("Adult");
} else if (input >= 13) {
    console.log("Teenager");
} else {
    console.log("Child");
}

// Ternary Operator (Short If)
const ageStatus = age >= 18 ? "Adult" : "Minor";
```

Source: conditionals.js

Conditional Statements (cont’d)

Switch Statement:

- Perfect for exact value matching
- Compare the value of **input** with the value of **case**
- Default case handles all other values that’s not specified before



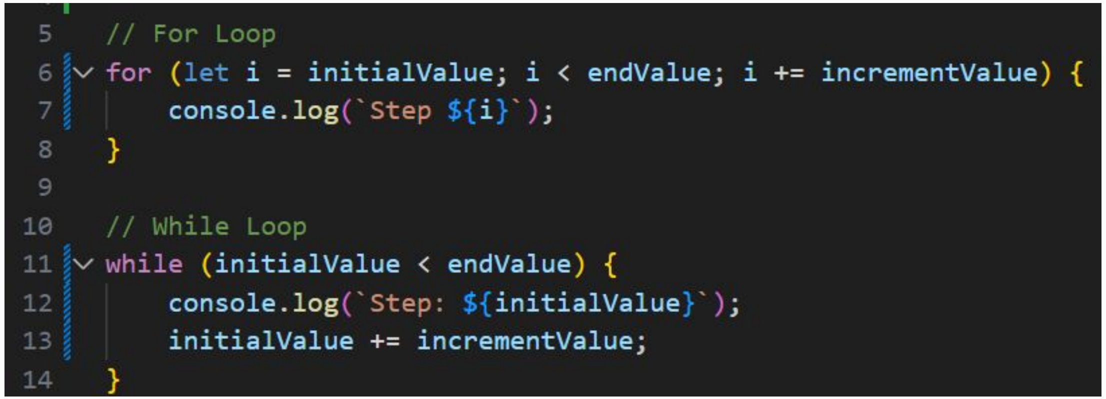
```javascript
// Switch Statement
switch (input) {
    case "Monday":
        console.log("Start of week");
        break;
    case "Friday":
        console.log("Weekend soon!");
        break;
    default:
        console.log("Regular day");
}
```

Source: conditionals.js

Loops

For Loop:

- Three parts: initial value, end/termination value, increment value
- Great if you know when to stop



```
// For Loop
for (let i = initialValue; i < endValue; i += incrementValue) {
    console.log(`Step ${i}`);
}

// While Loop
while (initialValue < endValue) {
    console.log(`Step: ${initialValue}`);
    initialValue += incrementValue;
}
```

Source: loops.js

While Loop:

- Runs while condition is true
- Good when you're not sure when to stop

Important! Make sure end value is achievable to avoid looping endlessly

Loops (cont'd)

For...of Loop

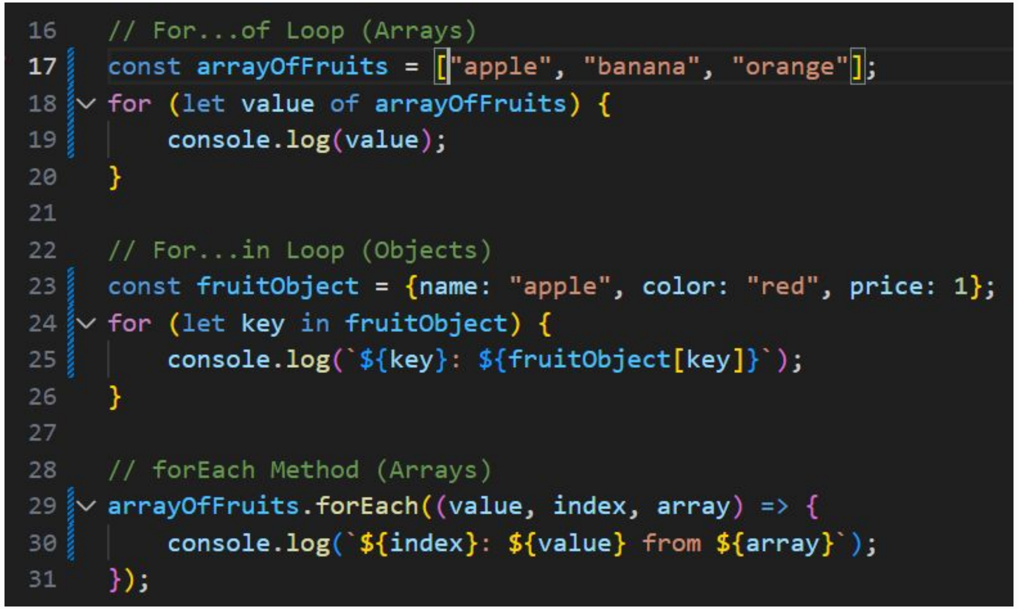
- Best for Arrays & Strings
- Loops through values directly (no need for index)

For...in Loop:

- Best for objects
- Loops through keys (property names) in an object.

forEach Method:

- Best for Arrays (when you need index value)
- Provides value, index, and array



```javascript
// For...of Loop (Arrays)
const arrayOfFruits = ["apple", "banana", "orange"];
for (let value of arrayOfFruits) {
    console.log(value);
}

// For...in Loop (Objects)
const fruitObject = {name: "apple", color: "red", price: 1};
for (let key in fruitObject) {
    console.log(`${key}: ${fruitObject[key]}`);
}

// forEach Method (Arrays)
arrayOfFruits.forEach((value, index, array) => {
    console.log(`${index}: ${value} from ${array}`);
});
```

Source: loops.js

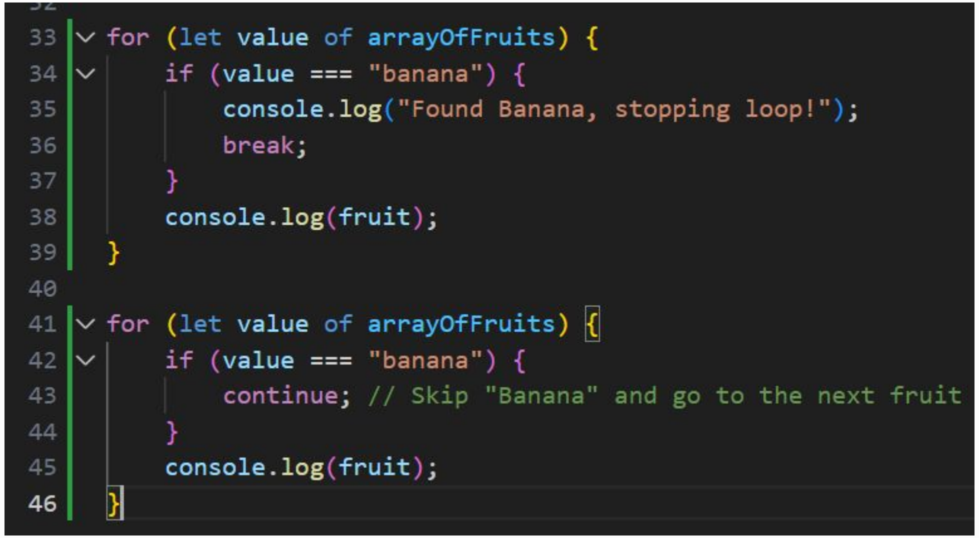
Loops - Breaking and Skipping Iteration

break Statement

- Stops the loop completely
- Used to exit a loop before it finishes.
- Commonly used when a condition is met.

continue Statement

- Skips the current iteration and moves to the next one.
- Useful when you want to ignore certain values.



```javascript
for (let value of arrayOfFruits) {
    if (value === "banana") {
        console.log("Found Banana, stopping loop!");
        break;
    }
    console.log(fruit);
}

for (let value of arrayOfFruits) {
    if (value === "banana") {
        continue; // Skip "Banana" and go to the next fruit
    }
    console.log(fruit);
}
```

Source: loops.js

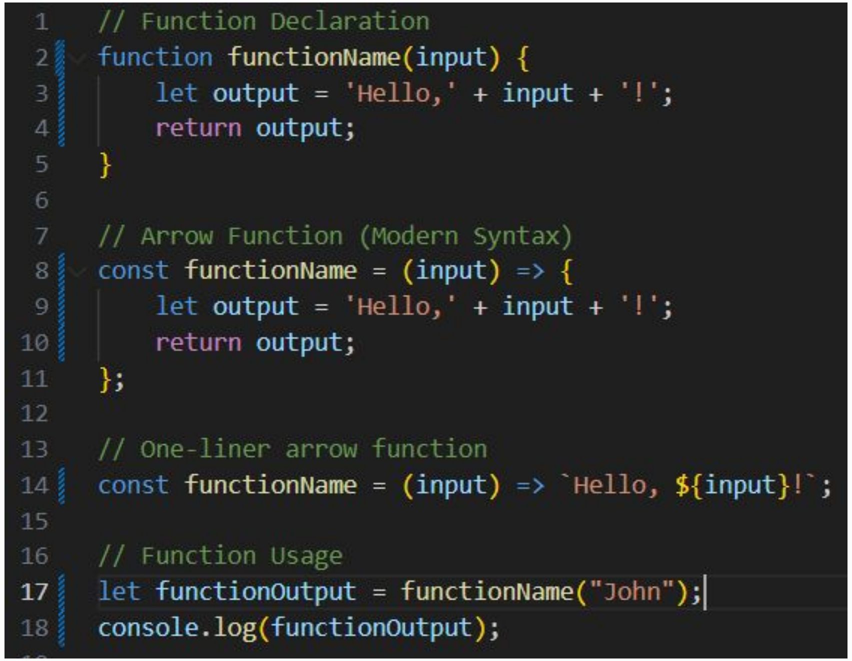
Functions

Components:

- Name: What we call the function
- Parameters: Input values (optional)
- Body: Code to execute
- Return value: Output (optional)

When to use each type:

- Arrow functions: Alternative to traditional
- One-liners: For simple operations



```javascript
// Function Declaration
function functionName(input) {
    let output = 'Hello,' + input + '!';
    return output;
}

// Arrow Function (Modern Syntax)
const functionName = (input) => {
    let output = 'Hello,' + input + '!';
    return output;
};

// One-liner arrow function
const functionName = (input) => `Hello, ${input}!`;

// Function Usage
let functionOutput = functionName("John");
console.log(functionOutput);
```

Source: functions.js

Document Object Model (DOM)

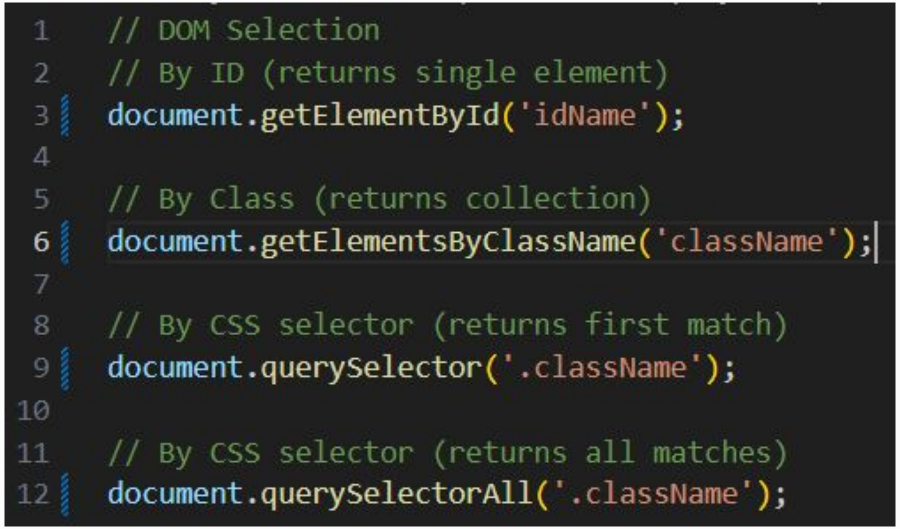
The structure of HTML elements.

Allows JavaScript to:

- Access HTML elements
- Modify content
- Change styles
- React to events

Selection Tips:

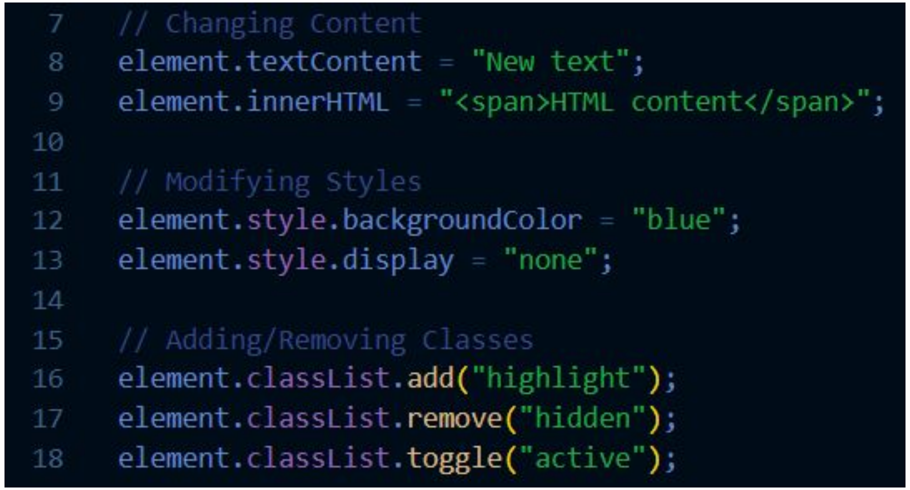
- IDs must be unique
- Classes can be reused
- querySelector is most flexible
- Always check if element exists before using



```javascript
// DOM Selection
// By ID (returns single element)
document.getElementById('idName');

// By Class (returns collection)
document.getElementsByClassName('className');

// By CSS selector (returns first match)
document.querySelector('.className');

// By CSS selector (returns all matches)
document.querySelectorAll('.className');
```

Source: dom.js

DOM Manipulation



```
// Changing Content
element.textContent = "New text";
element.innerHTML = "<span>HTML content</span>";

// Modifying Styles
element.style.backgroundColor = "blue";
element.style.display = "none";

// Adding/Removing Classes
element.classList.add("highlight");
element.classList.remove("hidden");
element.classList.toggle("active");
```

Source: dom.js

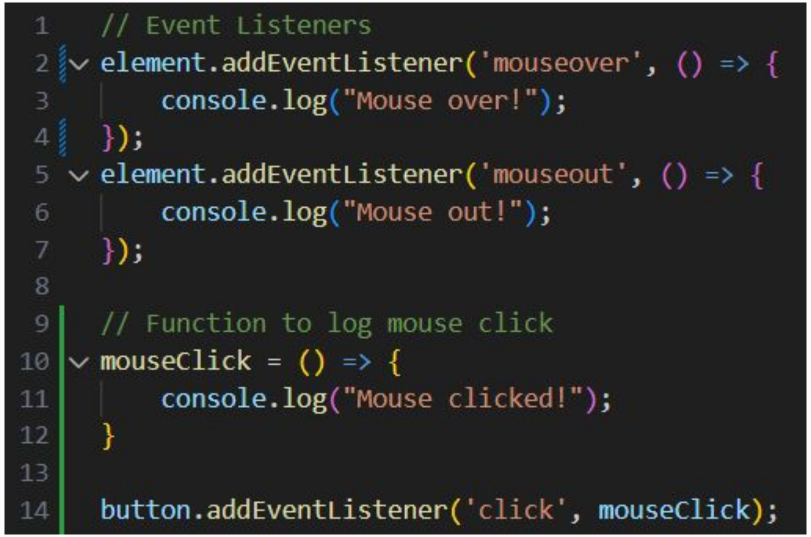
Best Practice:

- Use textContent for plain text
- Be cautious with innerHTML (XSS vulnerability)

Events Handling

Common Events:

- click: Mouse clicks
- mouseover/mouseout: Hover states
- keydown/keyup: Keyboard input
- submit: Form submission
- load: Page/resource loading
- resize: Window resizing
- scroll: Page scrolling



```javascript
// Event Listeners
element.addEventListener('mouseover', () => {
    console.log("Mouse over!");
});
element.addEventListener('mouseout', () => {
    console.log("Mouse out!");
});

// Function to log mouse click
mouseClick = () => {
    console.log("Mouse clicked!");
}

button.addEventListener('click', mouseClick);
```

Source: events.js

#That’s all for the material!

Do you have any questions?

Gives us feedback!

Follow our Instagram!

Further questions? Email me anytime at muhammad.effendy@my.sampoernauniversity.ac.id or through Microsoft Teams