# Web Development

Introduction to JavaScript

Khyari hamza

## What is JavaScript ?

- A **scripting** language used for **client-side** and **server-side** web development
- Adds **interactivity** and dynamic content to websites
- Can be used with HTML and CSS for a complete web development stack

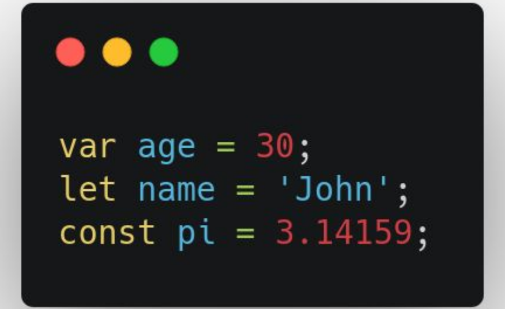
## Why learn JavaScript ?

- Widely used in web development
- Essential for creating interactive web applications
- Supported by all modern browsers
- In-demand skill for web developers

## JavaScript Basics

### Variables

- Containers for storing data values
- - Three ways to declare a variable:
    - **var** : Function-scoped (older method)
    - **let** : Block-scoped (introduced in ES6)
    - **const** : Block-scoped, can't be reass



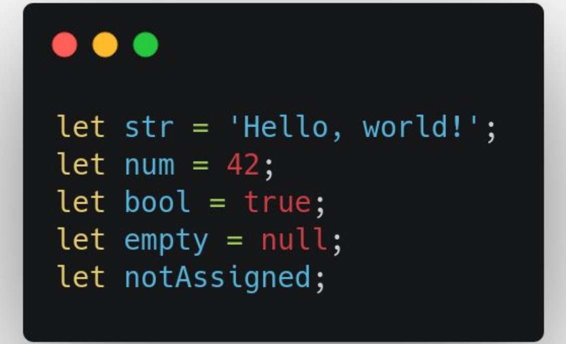
```
var age = 30;
let name = 'John';
const pi = 3.14159;
```

## JavaScript Basics

### Data Types

JavaScript has a few basic data types:

- Strings
- Numbers
- Booleans
- Null
- Undefined

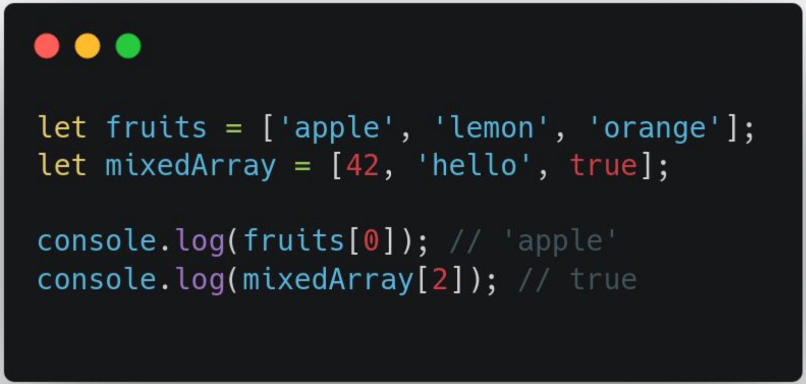


```
let str = 'Hello, world!';
let num = 42;
let bool = true;
let empty = null;
let notAssigned;
```

# JavaScript Basics

## Array

- Ordered collections of elements
- Can store elements of different data types

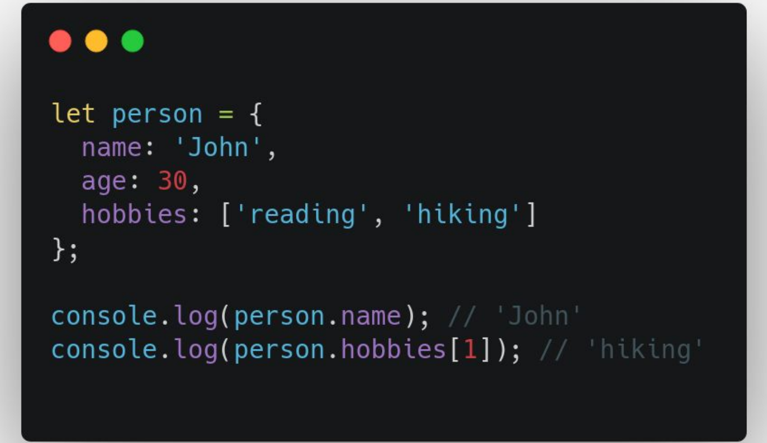


```javascript
let fruits = ['apple', 'lemon', 'orange'];
let mixedArray = [42, 'hello', true];

console.log(fruits[0]); // 'apple'
console.log(mixedArray[2]); // true
```

## JavaScript Basics

### Objects

- Collections of key-value pairs
- Keys are strings, values can be any data type

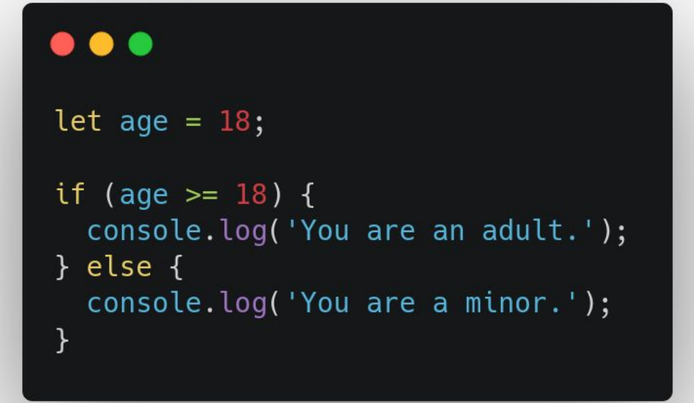


```
let person = {
  name: 'John',
  age: 30,
  hobbies: ['reading', 'hiking']
};

console.log(person.name); // 'John'
console.log(person.hobbies[1]); // 'hiking'
```

## Control Structures

### If Statements

- Used to make decisions in code
- Executes a block of code if a specified condition is true

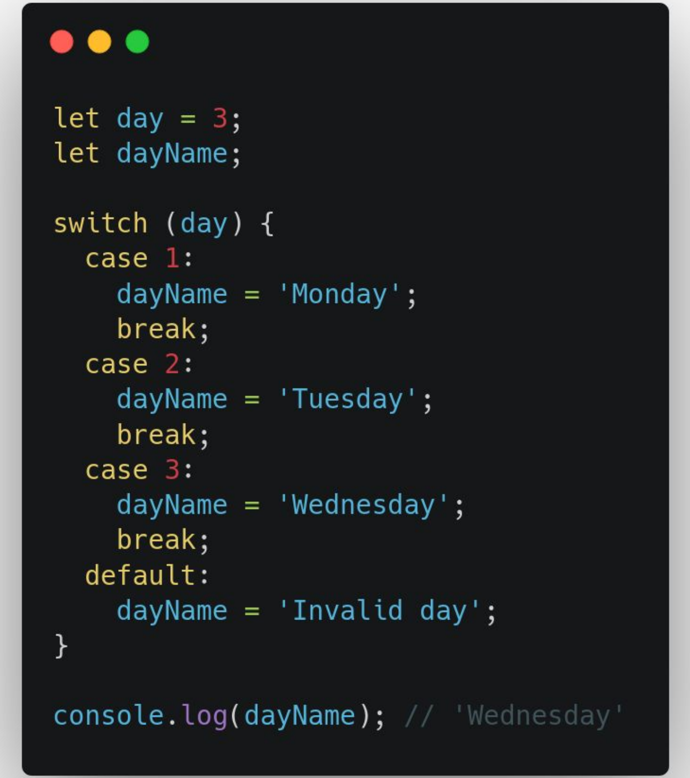


```
let age = 18;

if (age >= 18) {
  console.log('You are an adult.');
} else {
  console.log('You are a minor.');
}
```

## Control Structures

### Switch Statements

- Used to select one of many code blocks to be executed

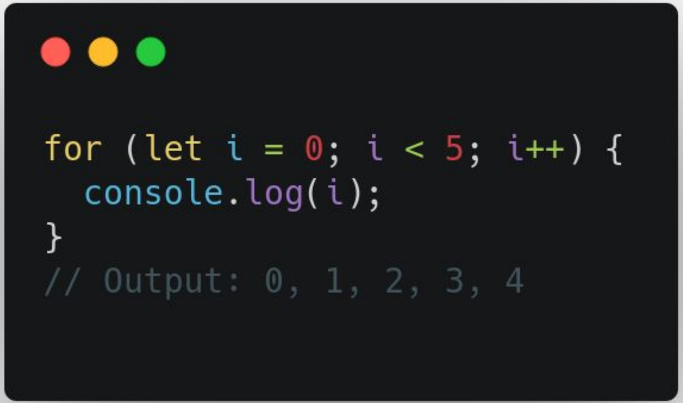


```
let day = 3;
let dayName;

switch (day) {
  case 1:
    dayName = 'Monday';
    break;
  case 2:
    dayName = 'Tuesday';
    break;
  case 3:
    dayName = 'Wednesday';
    break;
  default:
    dayName = 'Invalid day';
}

console.log(dayName); // 'Wednesday'
```

## Control Structures

### for loops

- Used to repeatedly execute a block of code a specific number of times

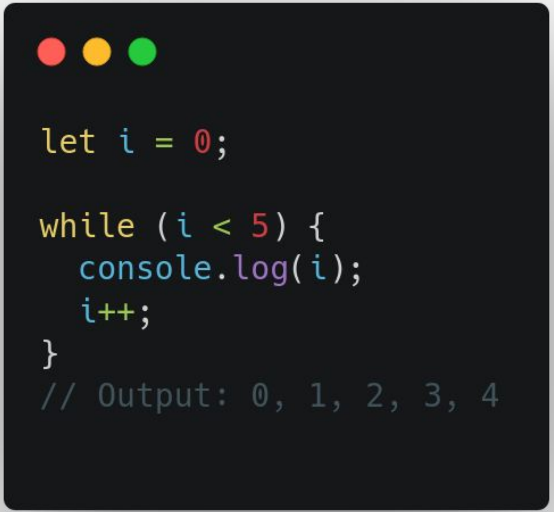


```
for (let i = 0; i < 5; i++) {
  console.log(i);
}
// Output: 0, 1, 2, 3, 4
```

# Control Structures

**While loops**

- Executes a block of code as long as a specified condition is true

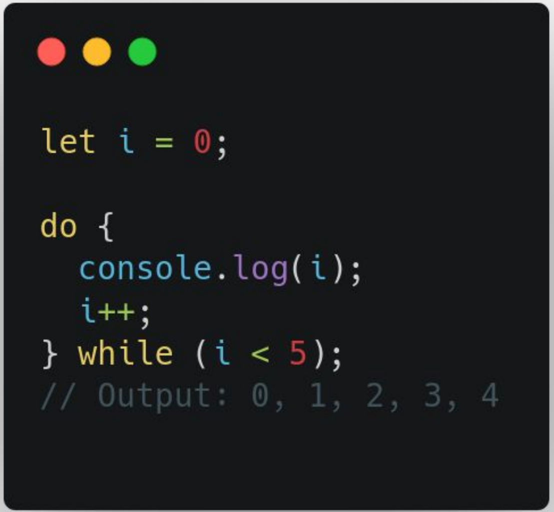


```
let i = 0;

while (i < 5) {
  console.log(i);
  i++;
}
// Output: 0, 1, 2, 3, 4
```

## Control Structures

### Do-While loops

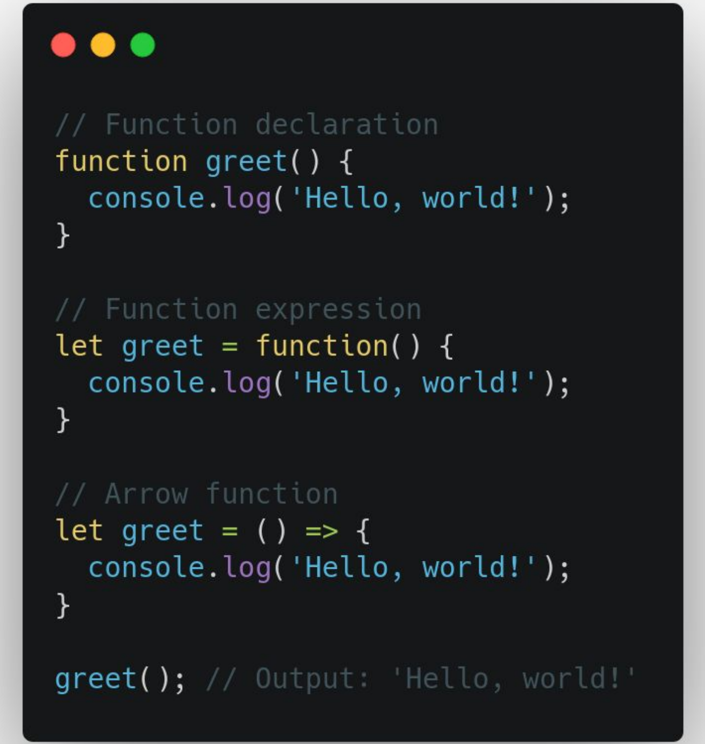
- Executes a block of code once, then repeats the loop as long as a specified condition is true



```
let i = 0;

do {
  console.log(i);
  i++;
} while (i < 5);
// Output: 0, 1, 2, 3, 4
```

## Functions

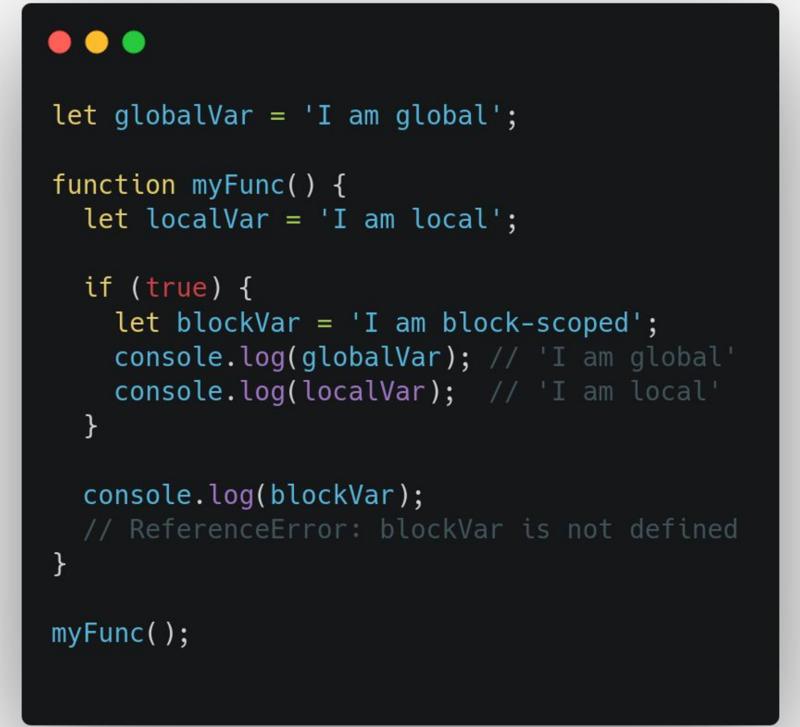
- Blocks of **reusable** code that perform a specific task
- Can be declared, assigned to variables, or defined as arrow functions



```
// Function declaration
function greet() {
  console.log('Hello, world!');
}

// Function expression
let greet = function() {
  console.log('Hello, world!');
}

// Arrow function
let greet = () => {
  console.log('Hello, world!');
}

greet(); // Output: 'Hello, world!'
```

## Scope

- Determines the visibility and lifetime of variables in code
- Variables can have global, local, or block scope

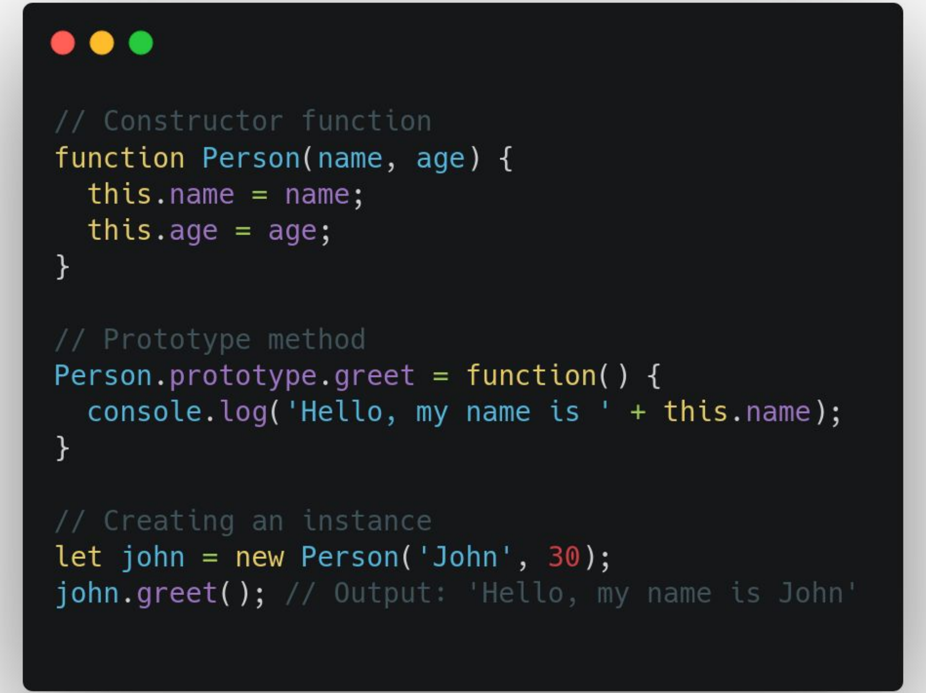


```javascript
let globalVar = 'I am global';

function myFunc() {
  let localVar = 'I am local';

  if (true) {
    let blockVar = 'I am block-scoped';
    console.log(globalVar); // 'I am global'
    console.log(localVar);  // 'I am local'
  }

  console.log(blockVar);
  // ReferenceError: blockVar is not defined
}

myFunc();
```

## Object-oriented JS

JavaScript supports object-oriented programming concepts, including:

- Objects and properties
- Methods
- Constructors and prototypes
- ES6 classes
- Inheritance



```javascript
// Constructor function
function Person(name, age) {
  this.name = name;
  this.age = age;
}

// Prototype method
Person.prototype.greet = function() {
  console.log('Hello, my name is ' + this.name);
}

// Creating an instance
let john = new Person('John', 30);
john.greet(); // Output: 'Hello, my name is John'
```

## DOM Manipulation

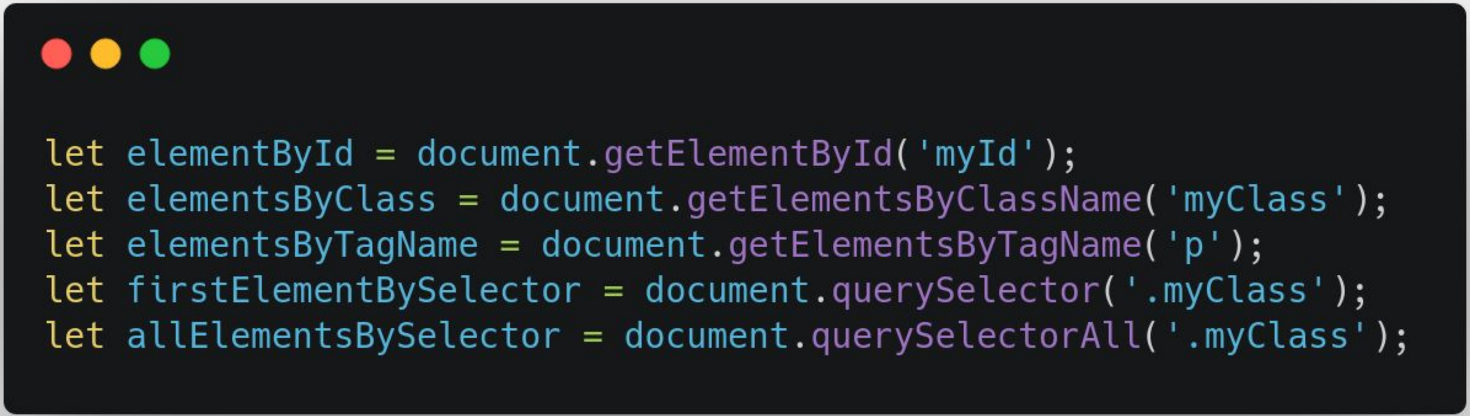
**What is DOM ?**

- Document Object Model (DOM) represents the structure of a web page
- Hierarchical tree-like structure
- JavaScript can be used to interact with the DOM to manipulate HTML and CSS

# DOM Manipulation

## DOM Selectors

- Used to select HTML elements on a web page
- Examples of common selectors:

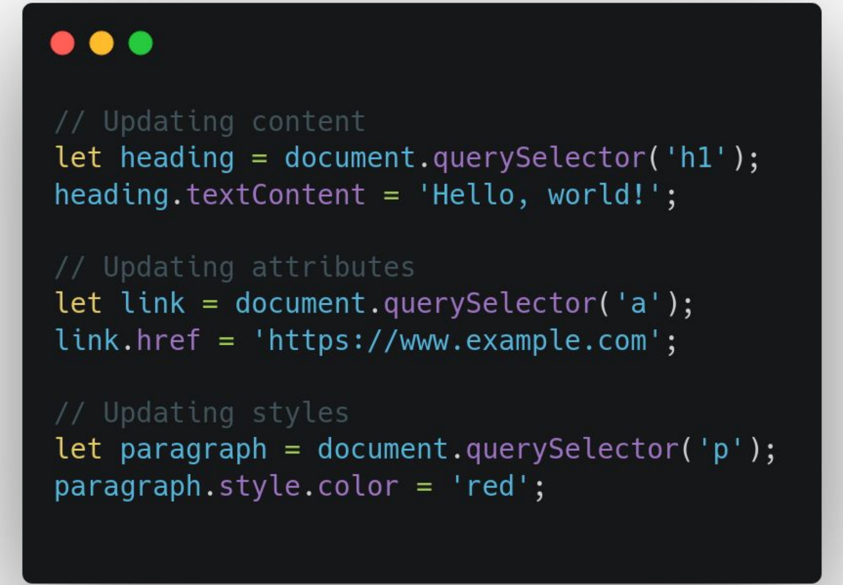


```javascript
let elementById = document.getElementById('myId');
let elementsByClass = document.getElementsByClassName('myClass');
let elementsByTagName = document.getElementsByTagName('p');
let firstElementBySelector = document.querySelector('.myClass');
let allElementsBySelector = document.querySelectorAll('.myClass');
```

# DOM Manipulation

**Updating Elements**

JavaScript can be used to modify the content, attributes, and styles of HTML elements

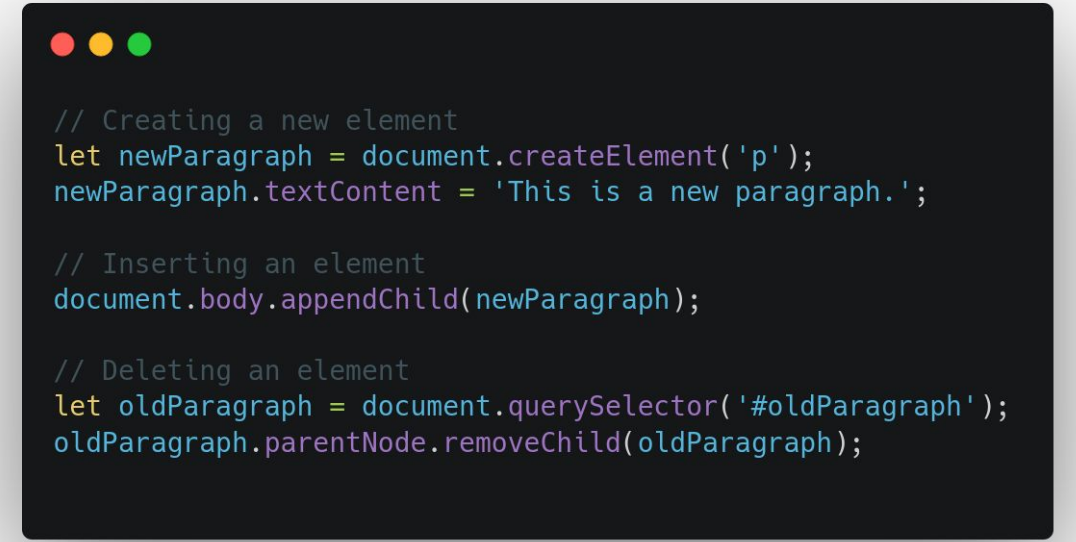


```javascript
// Updating content
let heading = document.querySelector('h1');
heading.textContent = 'Hello, world!';

// Updating attributes
let link = document.querySelector('a');
link.href = 'https://www.example.com';

// Updating styles
let paragraph = document.querySelector('p');
paragraph.style.color = 'red';
```

## DOM Manipulation

**Adding & Deleting Elements**

JavaScript can be used to create, insert, and delete HTML elements

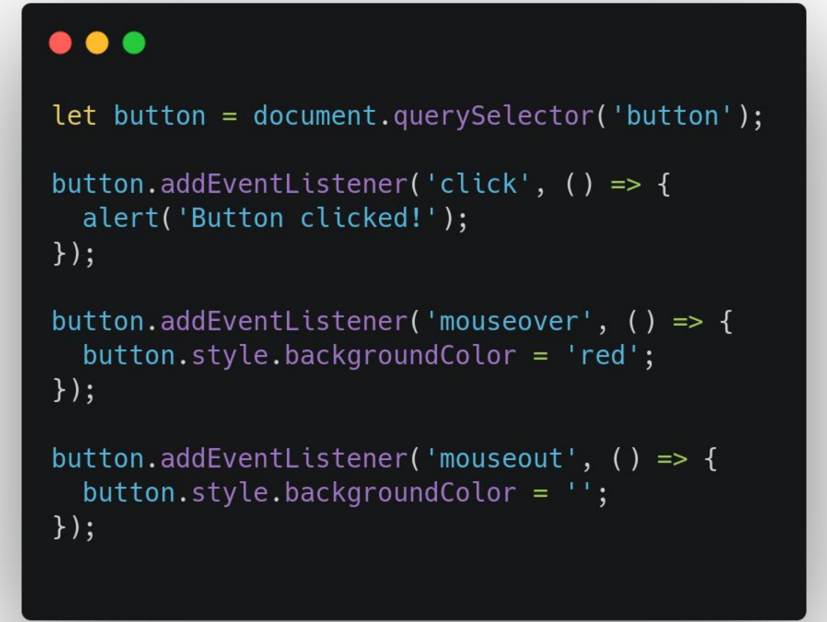


```
// Creating a new element
let newParagraph = document.createElement('p');
newParagraph.textContent = 'This is a new paragraph.';

// Inserting an element
document.body.appendChild(newParagraph);

// Deleting an element
let oldParagraph = document.querySelector('#oldParagraph');
oldParagraph.parentNode.removeChild(oldParagraph);
```

# DOM Manipulation

## Event Listeners

JavaScript can be used to respond to user interactions by attaching event listeners to elements



```
let button = document.querySelector('button');

button.addEventListener('click', () => {
  alert('Button clicked!');
});

button.addEventListener('mouseover', () => {
  button.style.backgroundColor = 'red';
});

button.addEventListener('mouseout', () => {
  button.style.backgroundColor = '';
});
```