



# Fakultas Ilmu Komputer

***Learning*** is the process of acquiring new understanding, knowledge, behaviors, skills, values, attitudes, and preferences.

## JavaScript HTML DOM



# **Welcome to *the* Front-end Web Development *course***



## web crawler

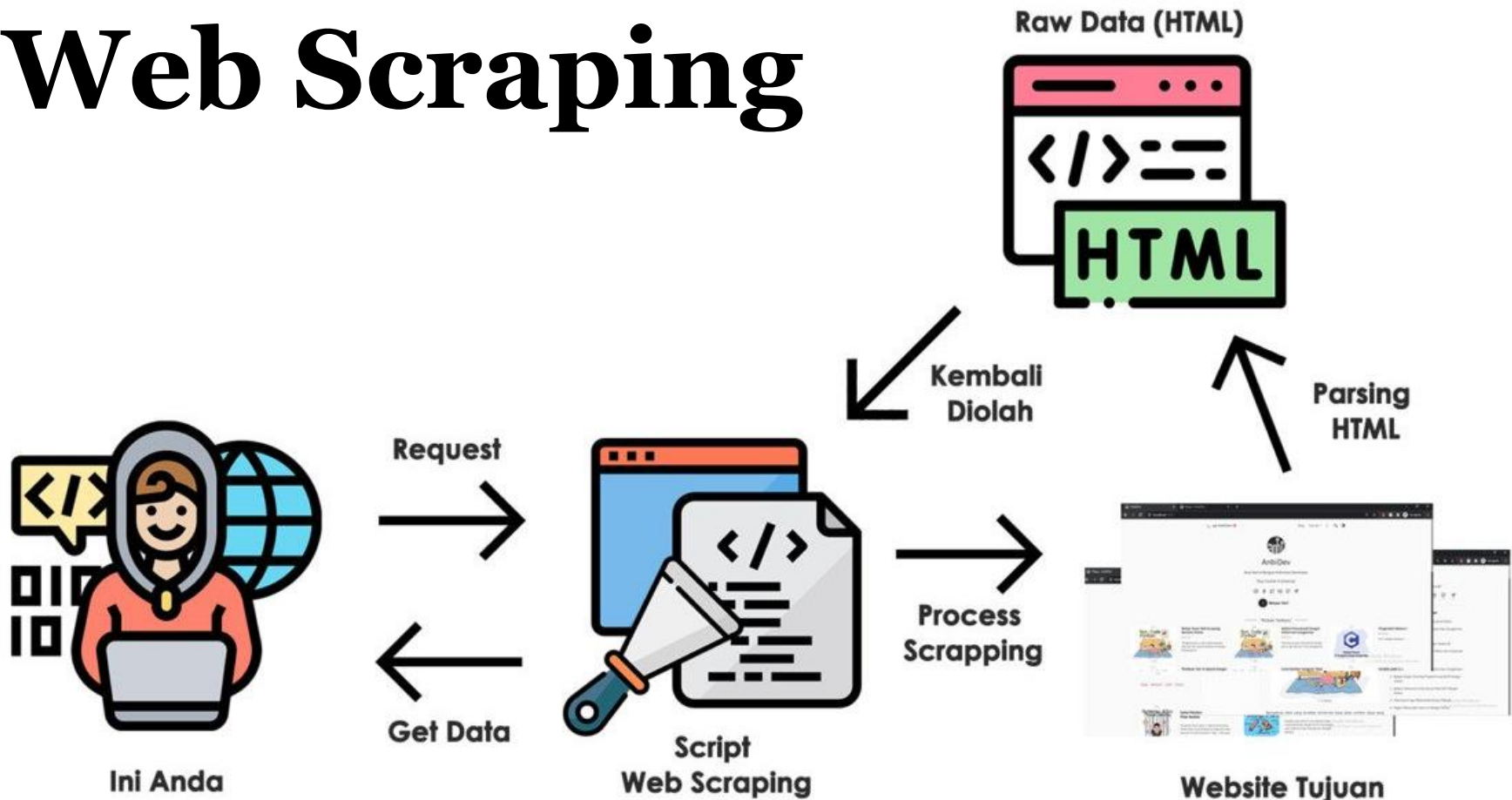


## web scraping



Web scraping is the process of using bots to extract content and data from a website. Unlike screen scraping, which only copies pixels displayed onscreen, web scraping extracts underlying HTML code and, with it, data stored in a database. The scraper can then replicate entire website content elsewhere.

# Langkah-langkah dalam Web Scrapping





**HOW TO  
DISABLE OR  
ENABLE COPY  
AND PASTE  
ON WEBSITE?**

**DO NOT  
COPY**

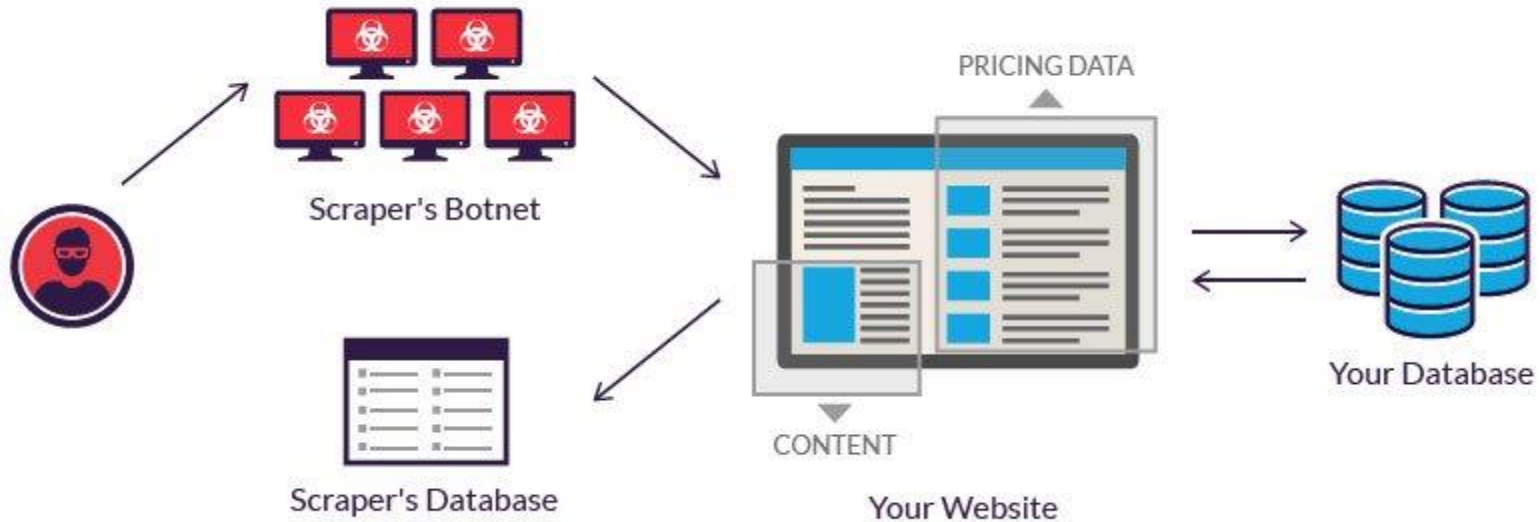
**DO NOT  
COPY**

**DO NOT  
COPY**

**DO NOT  
COPY**



## Botnet Scraper



Is scraping website legal?  
is this like the act of  
stealing?



# 1 - JavaScript Classes



# JavaScript Classes

- ❖ ECMAScript 2015, also known as ES6, introduced JavaScript Classes.
- ❖ JavaScript Classes are **templates** for JavaScript Objects.

## JavaScript Class Syntax

Use the keyword `class` to create a class.

Always add a method named `constructor()` :

### Syntax


```
class ClassName {  
  constructor() { ... }  
}
```

### Example

```
class Car {  
  constructor(name, year) {  
    this.name = name;  
    this.year = year;  
  }  
}
```

## Browser Support

The following table defines the first browser version with full support for Classes in JavaScript:

				
Chrome 49	Edge 12	Firefox 45	Safari 9	Opera 36
Mar, 2016	Jul, 2015	Mar, 2016	Oct, 2015	Mar, 2016

# Class Methods

```
<!DOCTYPE html>
<html>
<body>
<h2>JavaScript Class Method</h2>
<p>Pass a parameter into the "age()" method.</p>
<p id="demo"></p>

<script>
  class Car {
    constructor(name, year) {
      this.name = name;
      this.year = year;
    }
    age(x) {
      return x - this.year;
    }
  }

  let date = new Date();
  let year = date.getFullYear();

  let myCar = new Car("Ford", 2014);
  document.getElementById("demo").innerHTML=
  "My car is " + myCar.age(year) + " years old.";
</script>

</body>
</html>
```

## JavaScript Class Method

Pass a parameter into the "age()" method.

My car is 8 years old.

# JavaScript Errors

## Throw, and Try...Catch...Finally

The try statement defines a code block to run (to try).

The catch statement defines a code block to handle any error.

The finally statement defines a code block to run regardless of the result.

The throw statement defines a custom error.

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

<h2>JavaScript Error Handling</h2>

<p>How to use <b>catch</b> to display an error.</p>

<p id="demo"></p>

<script>
  try {
    adddler("Welcome guest!");
  }
  catch(err) {
    document.getElementById("demo").innerHTML = err.message;
  }
</script>

</body>
</html>
```

### JavaScript Error Handling

How to use catch to display an error.

Coba tebak apakah code  
di samping bisa di run?  
Ataukah terdapat error?  
Tebak errornya dimana?

The slide features a white background with green leaves and branches framing the top and bottom edges. The leaves are vibrant green and appear to be from a tree or shrub. The text is centered in the middle of the slide.

## 2 - JavaScript HTML DOM

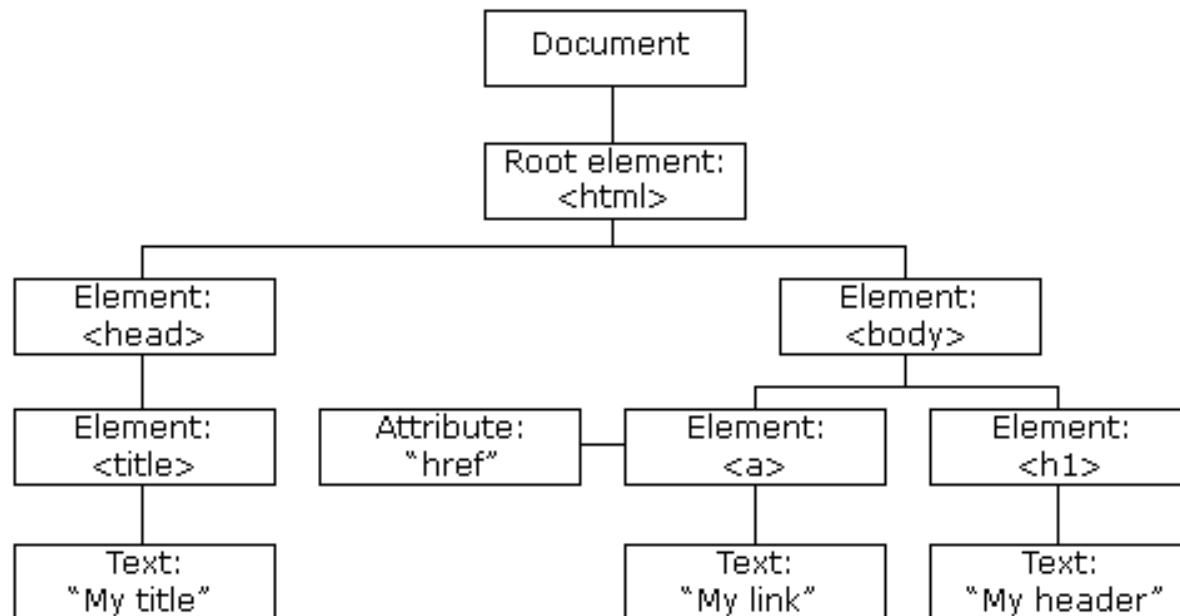
# JavaScript HTML DOM

**With the HTML DOM, JavaScript can access and change all the elements of an HTML document.**

## **The HTML DOM (Document Object Model)**

When a web page is loaded, the browser creates a **Document Object Model** of the page. The HTML DOM model is constructed as a tree of Objects:

The HTML DOM Tree of Objects





# Document Object Model (DOM)

With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- 1) JavaScript can change all the HTML elements in the page.
- 2) JavaScript can change all the HTML attributes in the page.
- 3) JavaScript can change all the CSS styles in the page.
- 4) JavaScript can remove existing HTML elements and attributes.
- 5) JavaScript can add new HTML elements and attributes.
- 6) JavaScript can react to all existing HTML events in the page.
- 7) JavaScript can create new HTML events in the page.

# What is the DOM?

- ❖ The DOM is a W3C (World Wide Web Consortium) standard.
- ❖ The DOM defines a standard for accessing documents:

"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."
- ❖ The W3C DOM standard is separated into 3 different parts:
  - ✓ Core DOM - standard model for all document types
  - ✓ XML DOM - standard model for XML documents
  - ✓ HTML DOM - standard model for HTML documents

# DOM

Document Object Model

document

Root element:  
<html>

Element:  
<head>

Element:  
<body>

Element:  
<title>

Text:  
"My title"

Element:  
<h1>

Text:  
"A heading"

Element:  
<a>

Attribute:  
href

Text:  
"Link text"

## What is the HTML DOM?

The HTML DOM is a standard object model and programming interface for HTML. It defines:

- ✓ The HTML elements as objects
- ✓ The properties of all HTML elements
- ✓ The methods to access all HTML elements
- ✓ The events for all HTML elements

**In other words:**

The HTML DOM is a standard for how to get, change, add, or delete HTML elements.

# JavaScript

(HTML DOM Methods)

# JavaScript - HTML DOM Methods

- ❖ HTML DOM **methods** are actions you can perform (on HTML Elements).
- ❖ HTML DOM **properties** are values (of HTML Elements) that you can set or change.



## THE DOM PROGRAMMING INTERFACE

- 1) The HTML DOM can be accessed with JavaScript (and with other programming languages).
- 2) In the DOM, all HTML elements are defined as **OBJECTS**.
- 3) The programming interface is the properties and methods of each object.
- 4) A property is a value that you can get or set (*like changing the content of an HTML element*).
- 5) A method is an action you can do (like add or deleting an HTML element).



# DOM Example

```
<html>
<body>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>

</body>
</html>
```

**Perhatikan code dalam script!!!**  
`getElementById` is a method, while  
`innerHTML` is a property.

The `innerHTML` property can be used to get or change any HTML element, including `<html>` and `<body>`.

## The `getElementById` Method

The most common way to access an HTML element is to use the `id` of the element.

In the example above the `getElementById` method used `id="demo"` to find the element.

## The `innerHTML` Property

The easiest way to get the content of an element is by using the `innerHTML` property.

The `innerHTML` property is useful for getting or replacing the content of HTML elements.

# JavaScript HTML DOM Document

The HTML DOM document object is the *owner* of all other objects in your web page.

## The HTML DOM Document Object

- ❖ The document object represents your web page.
- ❖ If you want to access any element in an HTML page, you always start with accessing the document object.
- ❖ Below are some examples of how you can use the document object to access and manipulate HTML.

### Finding HTML Elements

Method	Description
<code>document.getElementById(<i>id</i>)</code>	Find an element by element id
<code>document.getElementsByTagName(<i>name</i>)</code>	Find elements by tag name
<code>document.getElementsByClassName(<i>name</i>)</code>	Find elements by class name

### Changing HTML Elements

Property	Description
<code>element.innerHTML = new html content</code>	Change the inner HTML of an element
<code>element.attribute = new value</code>	Change the attribute value of an HTML element
<code>element.style.property = new style</code>	Change the style of an HTML element
Method	Description
<code>element.setAttribute(<i>attribute</i>, <i>value</i>)</code>	Change the attribute value of an HTML element

# Adding & Deleting Elements

Method	Description
<code>document.createElement(<i>element</i>)</code>	Create an HTML element
<code>document.removeChild(<i>element</i>)</code>	Remove an HTML element
<code>document.appendChild(<i>element</i>)</code>	Add an HTML element
<code>document.replaceChild(<i>new</i>, <i>old</i>)</code>	Replace an HTML element
<code>document.write(<i>text</i>)</code>	Write into the HTML output stream



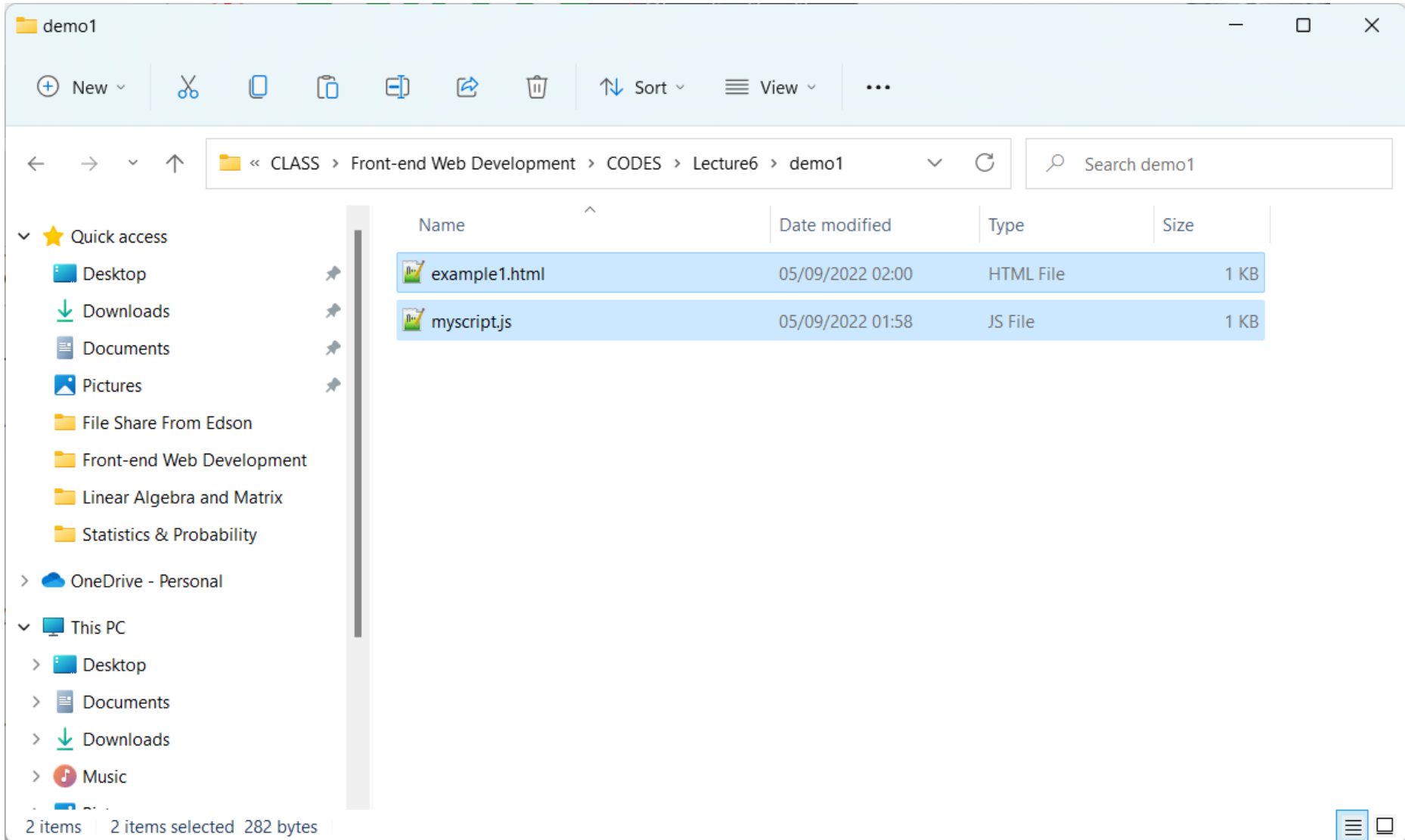
# Exercise *for* Students

# **Exercise #1**

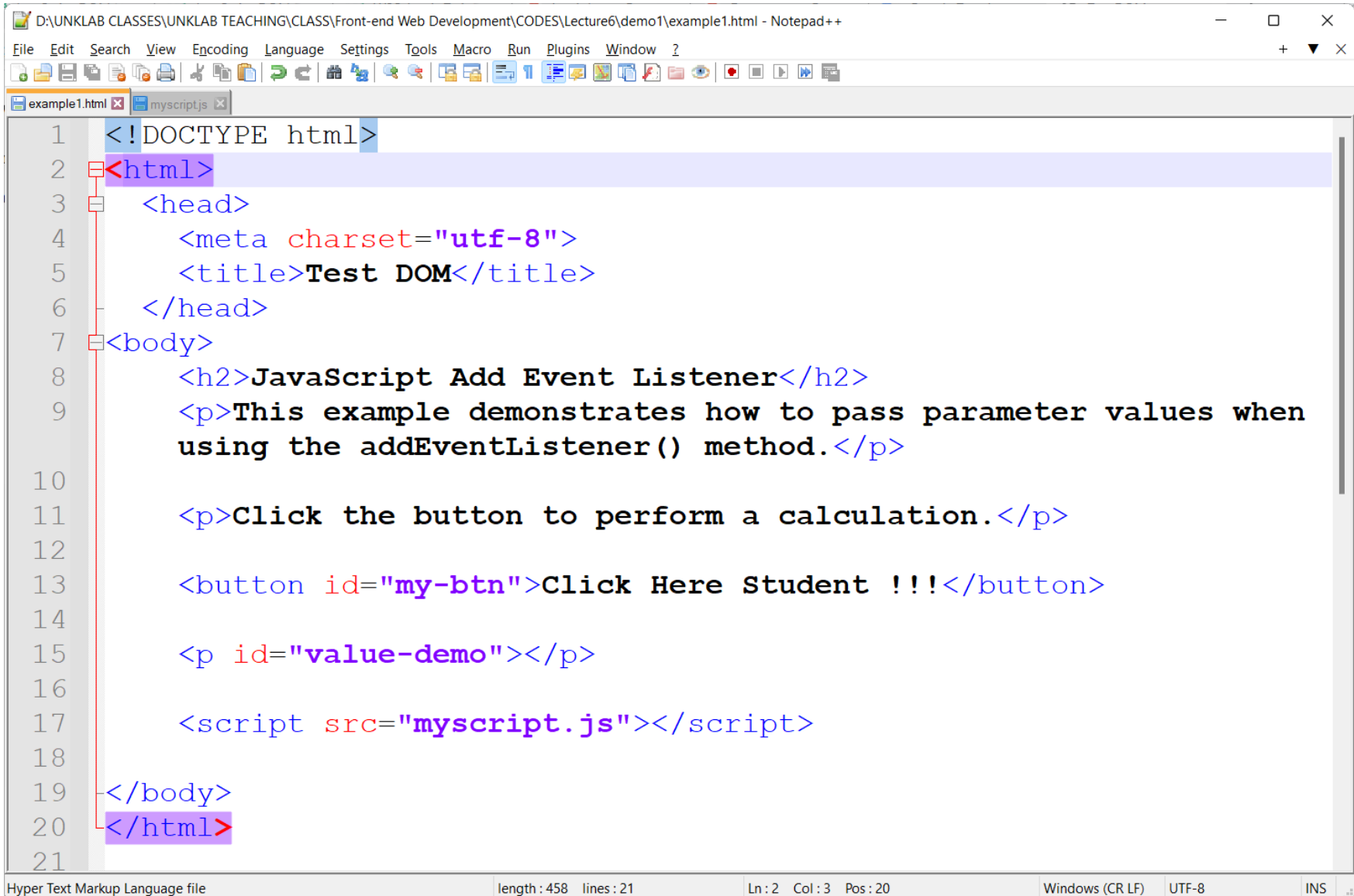
**JavaScript HTML DOM EventListener**



# Create New Folder “demo1”

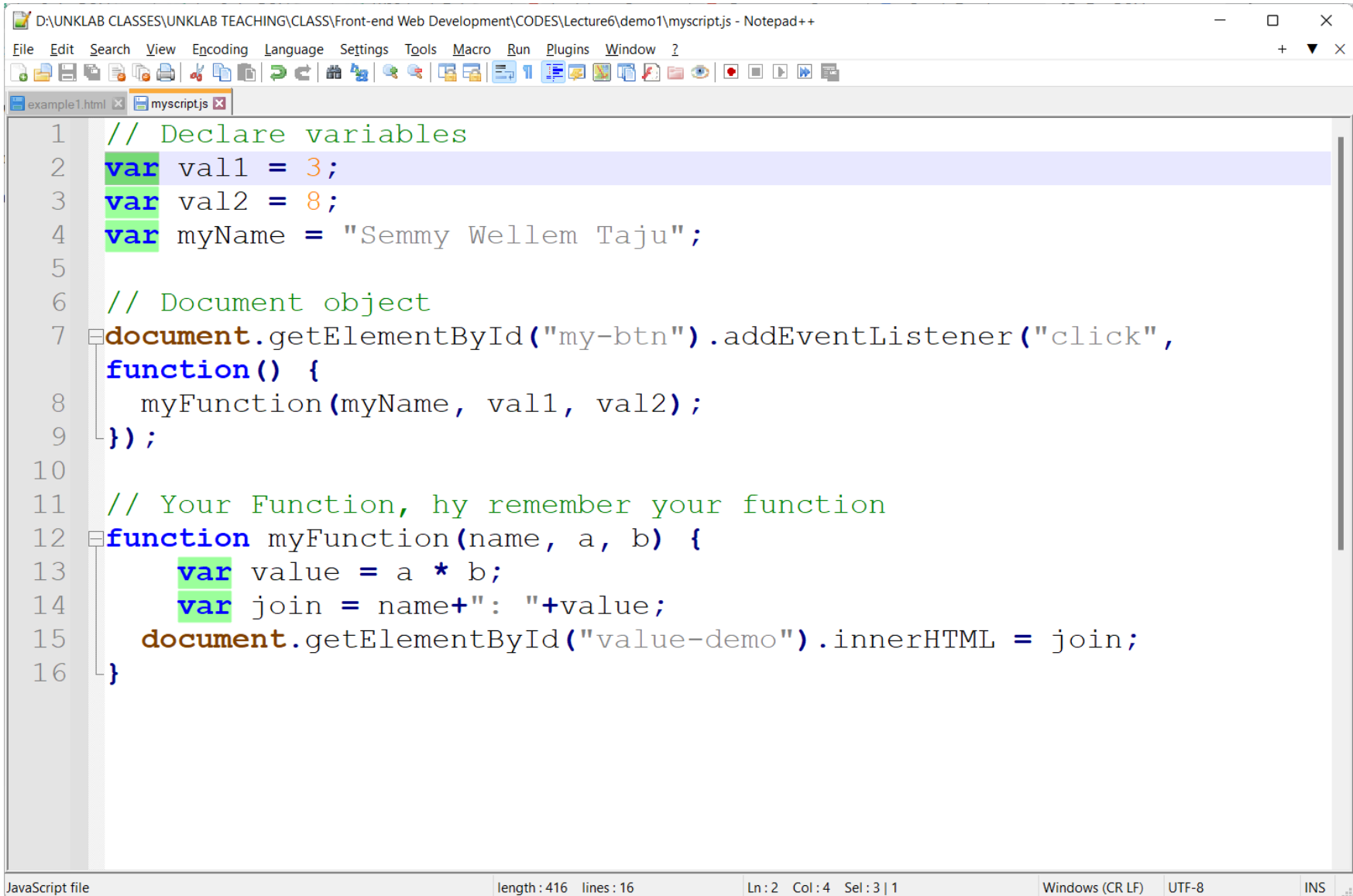


# Write HTML Code



```
D:\UNCLAB CLASSES\UNCLAB TEACHING\CLASS\Front-end Web Development\CODS\Lecture6\demo1\example1.html - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
example1.html myscript.js
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>Test DOM</title>
6   </head>
7   <body>
8     <h2>JavaScript Add Event Listener</h2>
9     <p>This example demonstrates how to pass parameter values when
10    using the addEventListener() method.</p>
11
12    <p>Click the button to perform a calculation.</p>
13
14    <button id="my-btn">Click Here Student !!!</button>
15
16    <p id="value-demo"></p>
17
18    <script src="myscript.js"></script>
19  </body>
20 </html>
21
Hyper Text Markup Language file    length : 458   lines : 21    Ln : 2   Col : 3   Pos : 20    Windows (CR LF)    UTF-8    INS
```

# Write JavaScript Code

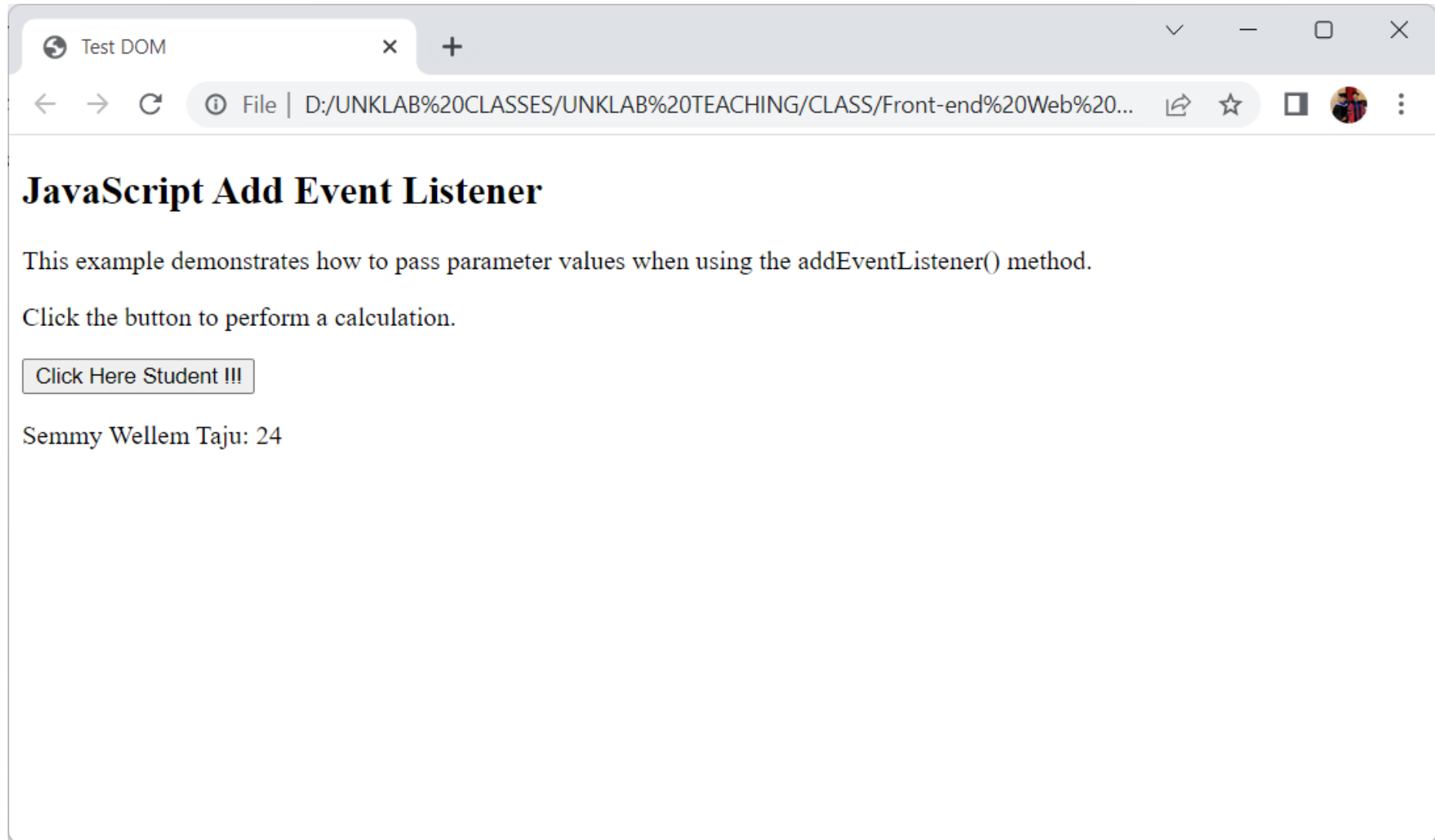


The image shows a Notepad++ window with the title bar "D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture6\demo1\myscript.js - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and Help. The toolbar contains various icons for file operations and editing. The active tab is "myscript.js". The code is as follows:

```
1 // Declare variables
2 var val1 = 3;
3 var val2 = 8;
4 var myName = "Semmy Wellem Taju";
5
6 // Document object
7 document.getElementById("my-btn").addEventListener("click",
  function() {
8     myFunction(myName, val1, val2);
9 });
10
11 // Your Function, hy remember your function
12 function myFunction(name, a, b) {
13     var value = a * b;
14     var join = name+": "+value;
15     document.getElementById("value-demo").innerHTML = join;
16 }
```

The status bar at the bottom displays "JavaScript file", "length: 416 lines: 16", "Ln: 2 Col: 4 Sel: 3 | 1", "Windows (CR LF)", "UTF-8", and "INS".

# Expected Output



## JavaScript Add Event Listener

This example demonstrates how to pass parameter values when using the `addEventListener()` method.

Click the button to perform a calculation.

Click Here Student !!!

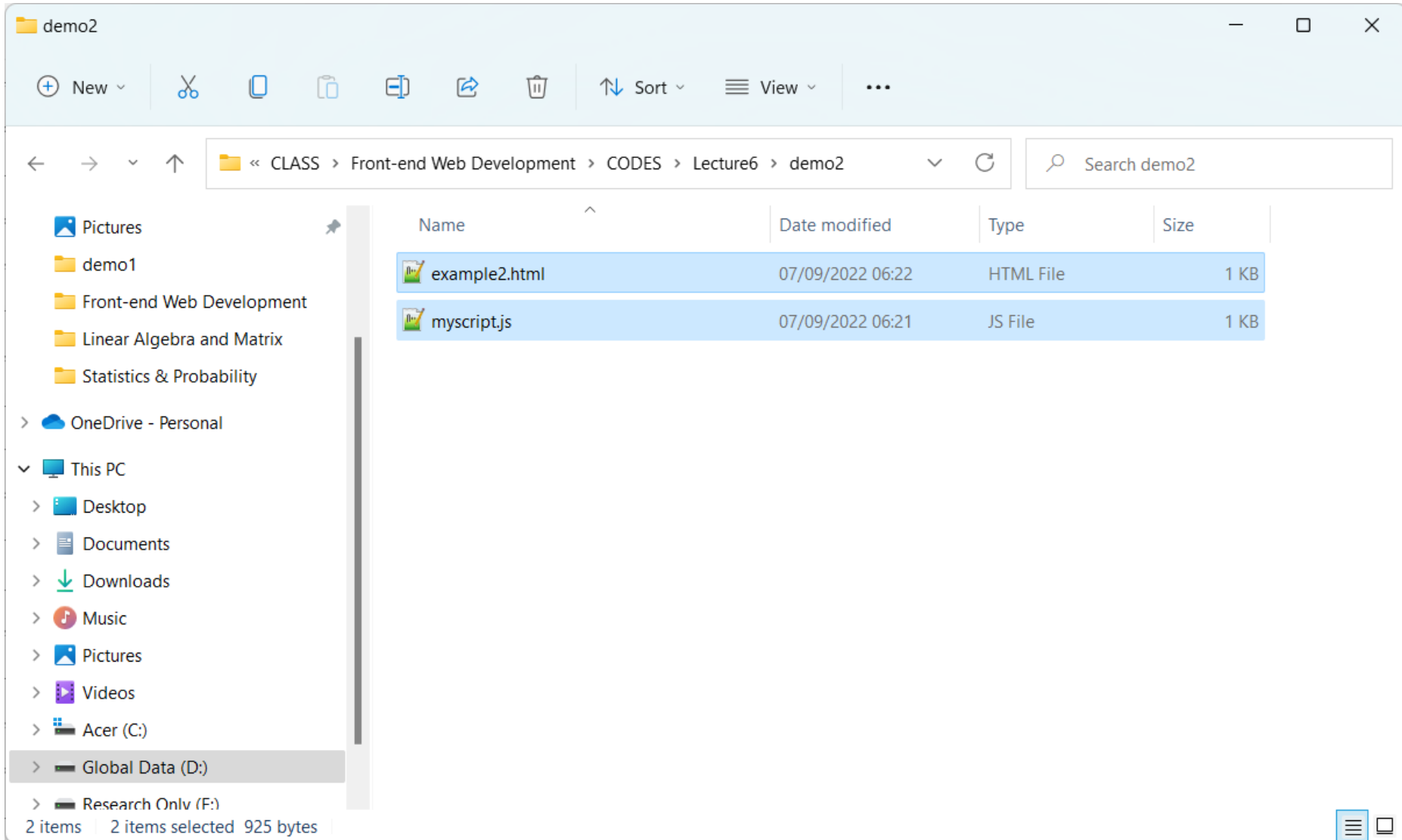
Semmy Wellem Taju: 24

# **Exercise #2**

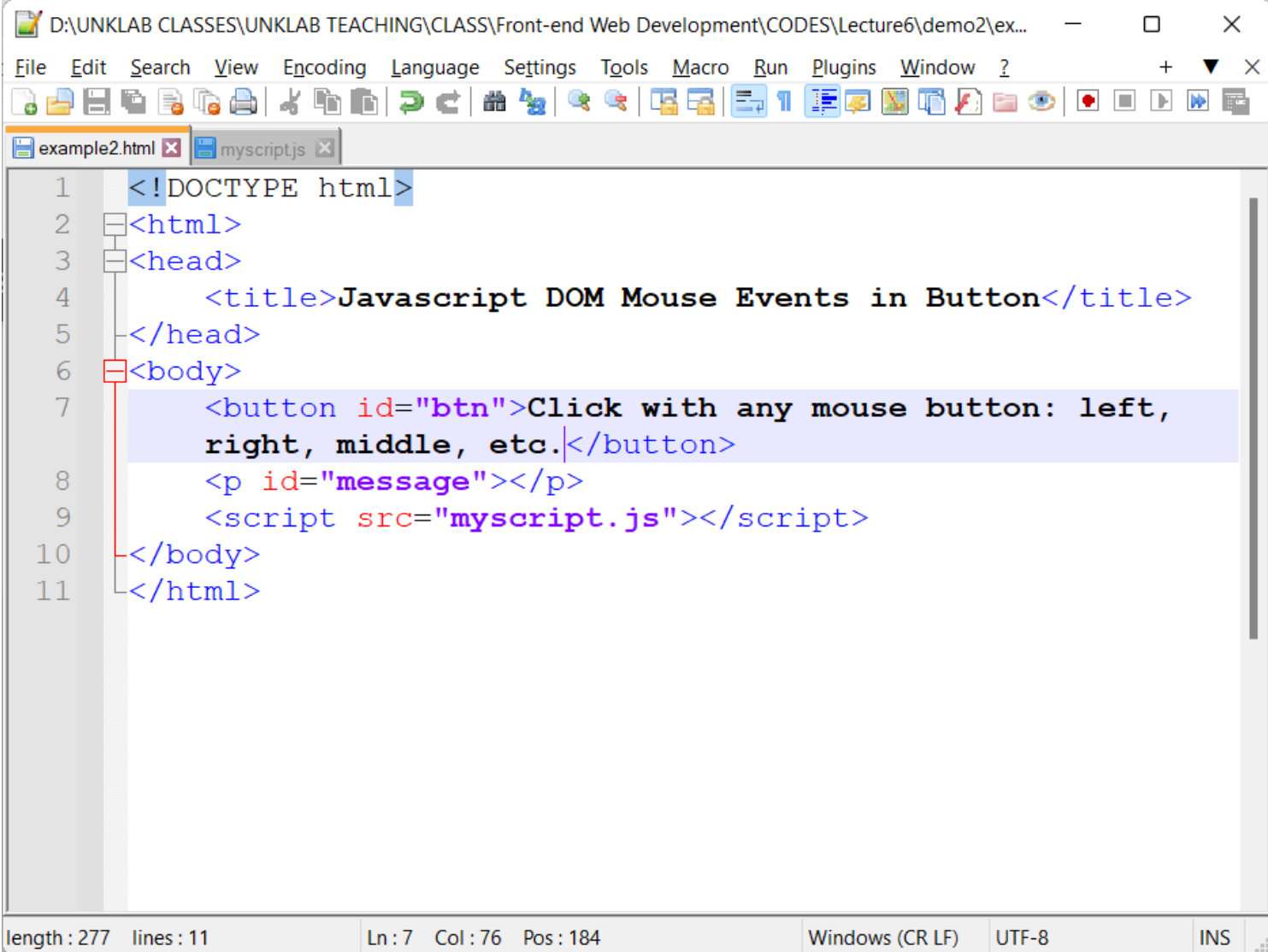
**HTML DOM MouseEvent  
(MouseEvent Properties and Methods)**



# Create New Folder “demo2”



# Write HTML Code

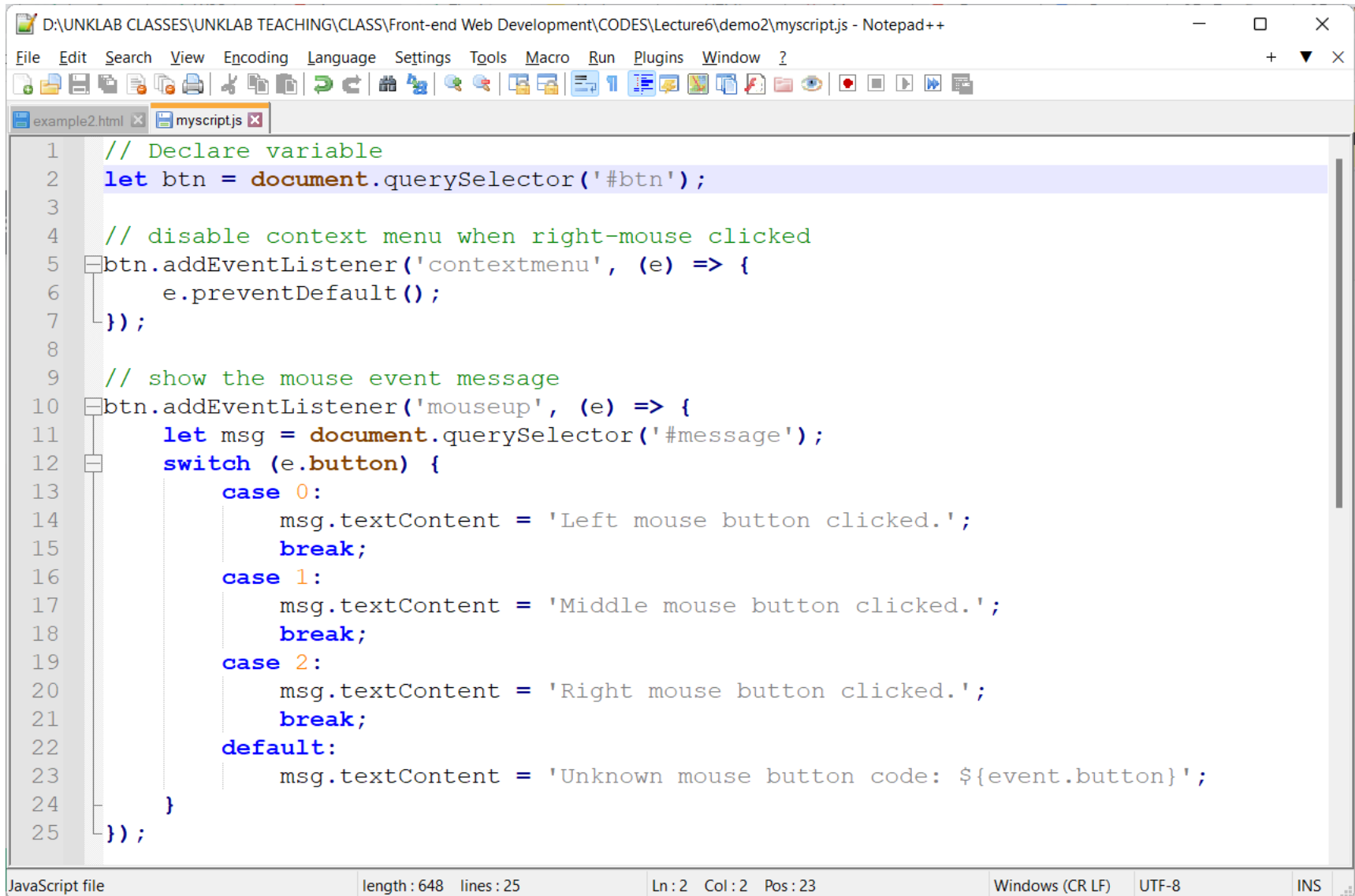


The screenshot shows a web development IDE with a menu bar (File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, ?) and a toolbar. Two tabs are open: 'example2.html' and 'myscript.js'. The 'example2.html' tab is active, displaying the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title>Javascript DOM Mouse Events in Button</title>
5 </head>
6 <body>
7     <button id="btn">Click with any mouse button: left,
8     right, middle, etc.</button>
9     <p id="message"></p>
10    <script src="myscript.js"></script>
11 </body>
12 </html>
```

The status bar at the bottom indicates: length : 277 lines : 11, Ln : 7 Col : 76 Pos : 184, Windows (CR LF), UTF-8, and INS.

# Write JavaScript Code



```
D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture6\demo2\myscript.js - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
example2.html x mysriptjs x
1 // Declare variable
2 let btn = document.querySelector('#btn');
3
4 // disable context menu when right-mouse clicked
5 btn.addEventListener('contextmenu', (e) => {
6     e.preventDefault();
7 });
8
9 // show the mouse event message
10 btn.addEventListener('mouseup', (e) => {
11     let msg = document.querySelector('#message');
12     switch (e.button) {
13         case 0:
14             msg.textContent = 'Left mouse button clicked.';
15             break;
16         case 1:
17             msg.textContent = 'Middle mouse button clicked.';
18             break;
19         case 2:
20             msg.textContent = 'Right mouse button clicked.';
21             break;
22         default:
23             msg.textContent = 'Unknown mouse button code: ${event.button}';
24     }
25 });
```

JavaScript file      length : 648   lines : 25      Ln : 2   Col : 2   Pos : 23      Windows (CR LF)   UTF-8   INS

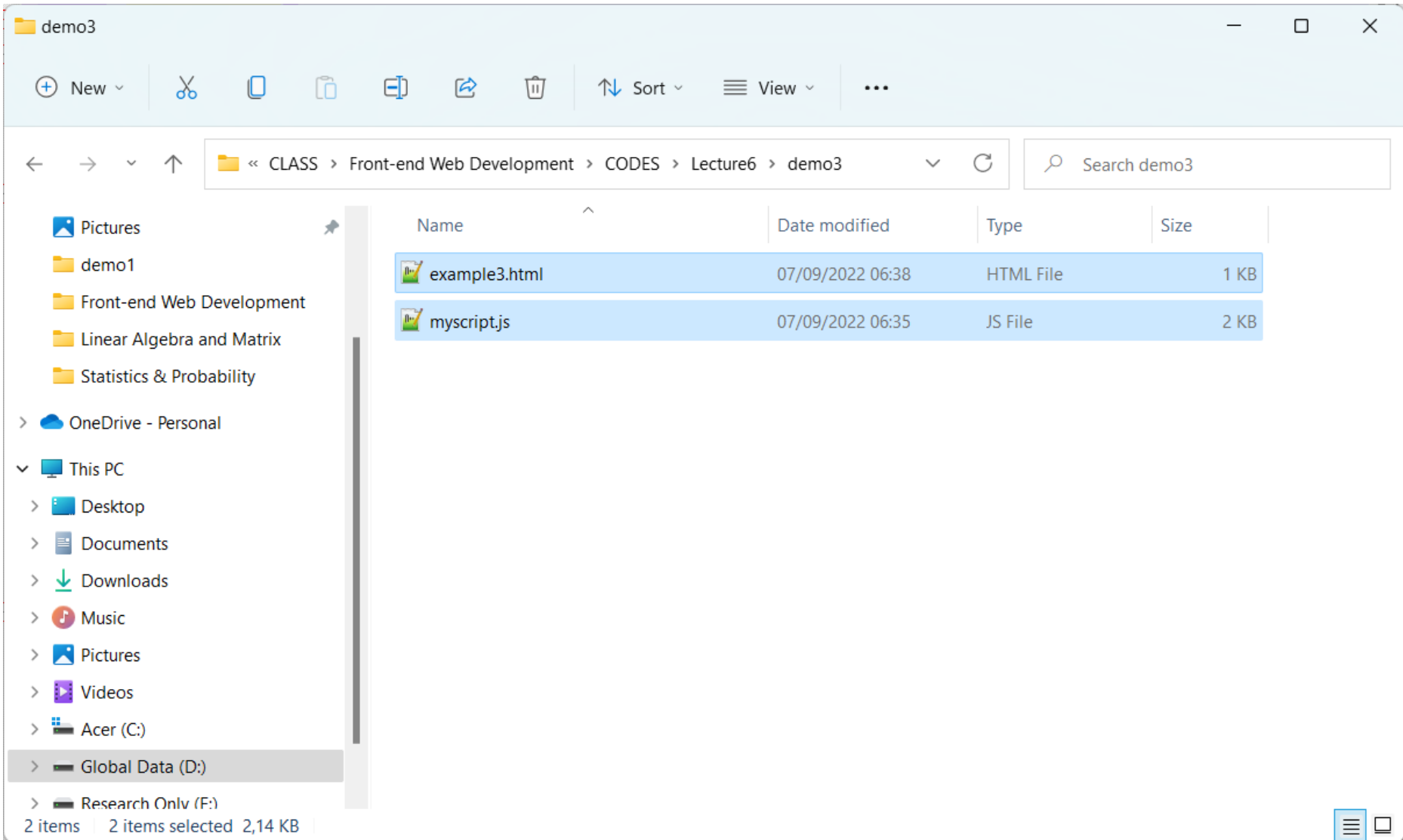
# Expected Output



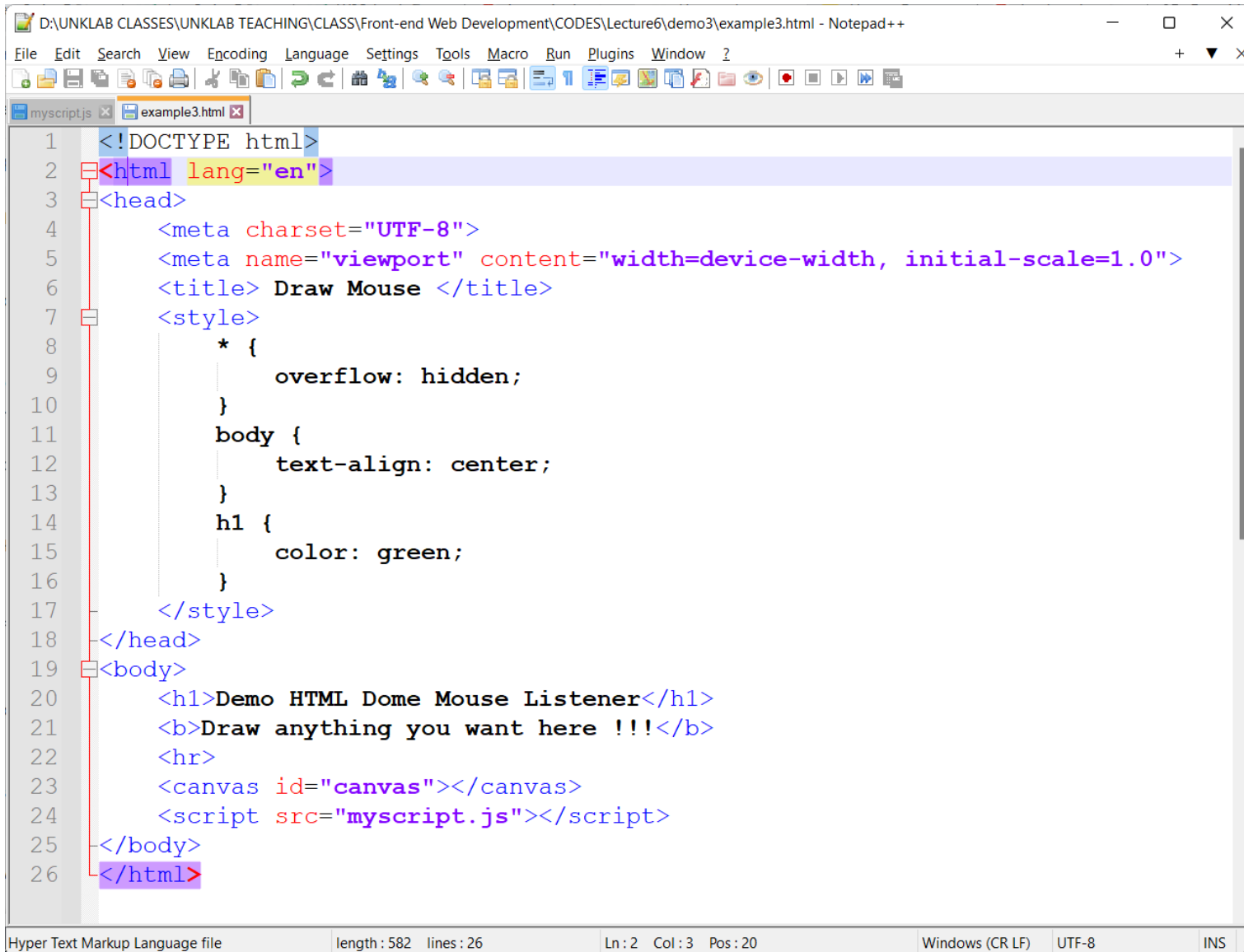
# **Exercise #3**

**JavaScript DOM Draw with mouse *in* HTML**

# Create New Folder “demo3”



# Write HTML Code



The screenshot shows a Notepad++ window with the title bar "D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture6\demo3\example3.html - Notepad++". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, and Window. The toolbar contains various icons for file operations and editing. The editor has two tabs: "myscript.js" and "example3.html". The "example3.html" tab is active, showing the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title> Draw Mouse </title>
7     <style>
8         * {
9             overflow: hidden;
10        }
11        body {
12            text-align: center;
13        }
14        h1 {
15            color: green;
16        }
17    </style>
18 </head>
19 <body>
20     <h1>Demo HTML Dome Mouse Listener</h1>
21     <b>Draw anything you want here !!!</b>
22     <hr>
23     <canvas id="canvas"></canvas>
24     <script src="myscript.js"></script>
25 </body>
26 </html>
```

The status bar at the bottom displays: "Hyper Text Markup Language file", "length : 582", "lines : 26", "Ln : 2", "Col : 3", "Pos : 20", "Windows (CR LF)", "UTF-8", and "INS".

# Write JavaScript Code

```
D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture6\demo3\myscript.js - Notep...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscript.js example3.html
1 // window element
2 window.addEventListener('load', ()=>{
3
4     resize(); // Resizes the canvas once the window loads
5     document.addEventListener('mousedown', startPainting);
6     document.addEventListener('mouseup', stopPainting);
7     document.addEventListener('mousemove', sketch);
8     window.addEventListener('resize', resize);
9 });
10
11 // declare variables
12 const canvas = document.querySelector('#canvas');
13 const ctx = canvas.getContext('2d'); // canvas for 2 dimensional operations
14 let coord = {x:0, y:0}; // Stores the initial position of the cursor
15 let paint = false; // trigger drawing
16
17 // Resizes the canvas to the available size of the window.
18 function resize(){
19     ctx.canvas.width = window.innerWidth;
20     ctx.canvas.height = window.innerHeight;
21 }
22
23 // Updates the coordinates of the cursor
24 function getPosition(event){
25     coord.x = event.clientX - canvas.offsetLeft;
26     coord.y = event.clientY - canvas.offsetTop;
27 }
28
29 // start and stop drawing
30 function startPainting(event){
31     paint = true;
32     getPosition(event);
33 }
34 function stopPainting(){
35     paint = false;
36 }
37
38 // Draws the line.
39 ctx.stroke();
40
41
42
43
44
45
46
47
48
49
50
51
52
53
JavaScript file length: 1.617 lines: 53 Ln: 1 Col: 1 Pos: 1 Windows (CR LF) UTF-8
```

```
D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture6\demo3\myscript.js - Notep...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscript.js example3.html
19 ctx.canvas.width = window.innerWidth;
20 ctx.canvas.height = window.innerHeight;
21 }
22
23 // Updates the coordinates of the cursor
24 function getPosition(event){
25     coord.x = event.clientX - canvas.offsetLeft;
26     coord.y = event.clientY - canvas.offsetTop;
27 }
28
29 // start and stop drawing
30 function startPainting(event){
31     paint = true;
32     getPosition(event);
33 }
34 function stopPainting(){
35     paint = false;
36 }
37
38 function sketch(event){
39     if (!paint) return;
40     ctx.beginPath();
41     ctx.lineWidth = 5;
42     // Sets round shape.
43     ctx.lineCap = 'round';
44     ctx.strokeStyle = 'green';
45     // The cursor to start drawing moves to this coordinate
46     ctx.moveTo(coord.x, coord.y);
47     // The position of the cursor gets updated as we move the mouse around.
48     getPosition(event);
49     // A line is traced from start coordinate to this coordinate
50     ctx.lineTo(coord.x, coord.y);
51     // Draws the line.
52     ctx.stroke();
53 }
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
JavaScript file length: 1.617 lines: 53 Ln: 53 Col: 2 Sel: 551 | 20 Windows (CR LF) UTF-8 INS
```



# Expected Output



# END PRESENTATION

Thank you for your attention

Instructor: S – W – T

THANK YOU

