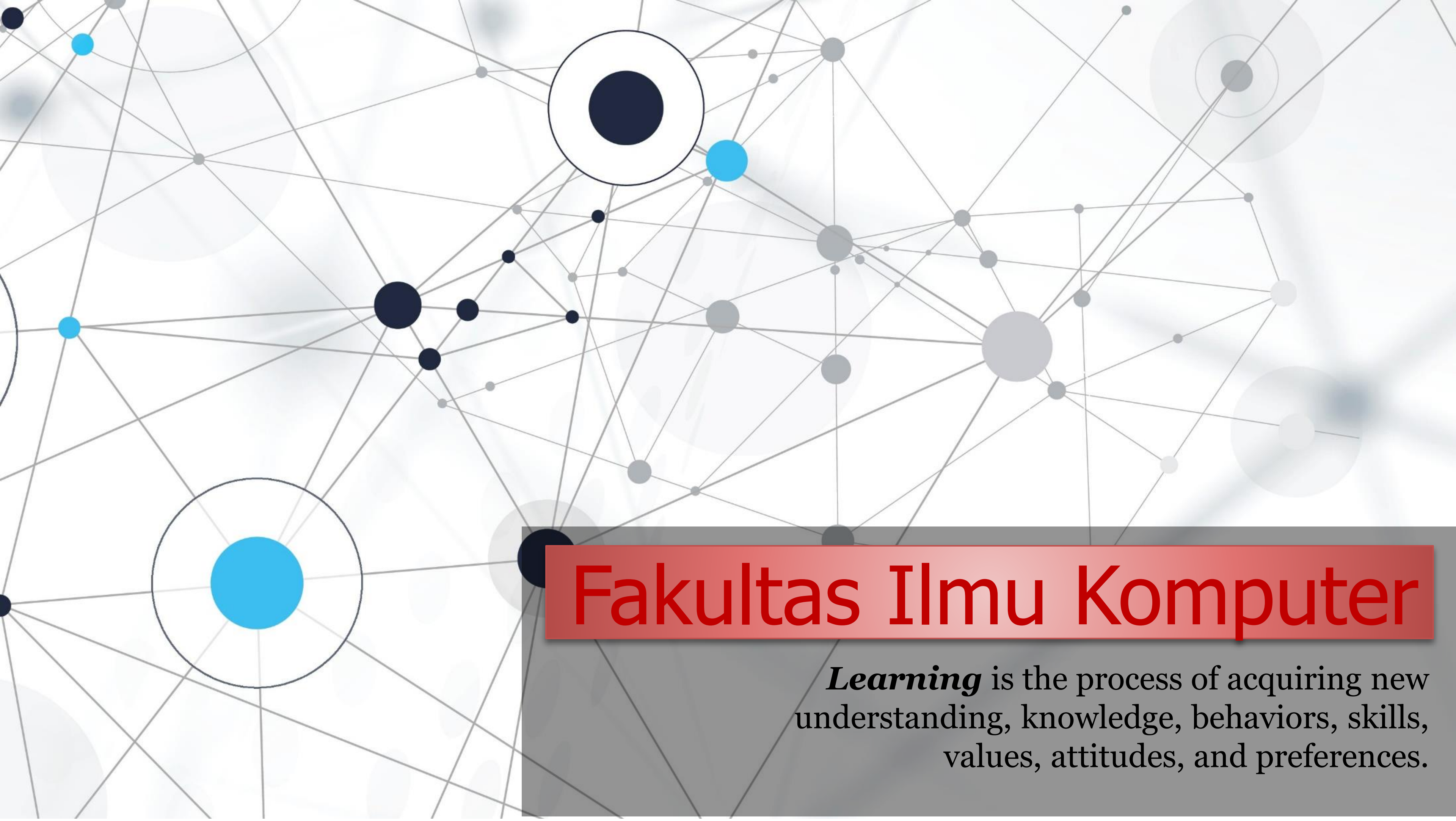


Fakultas Ilmu Komputer

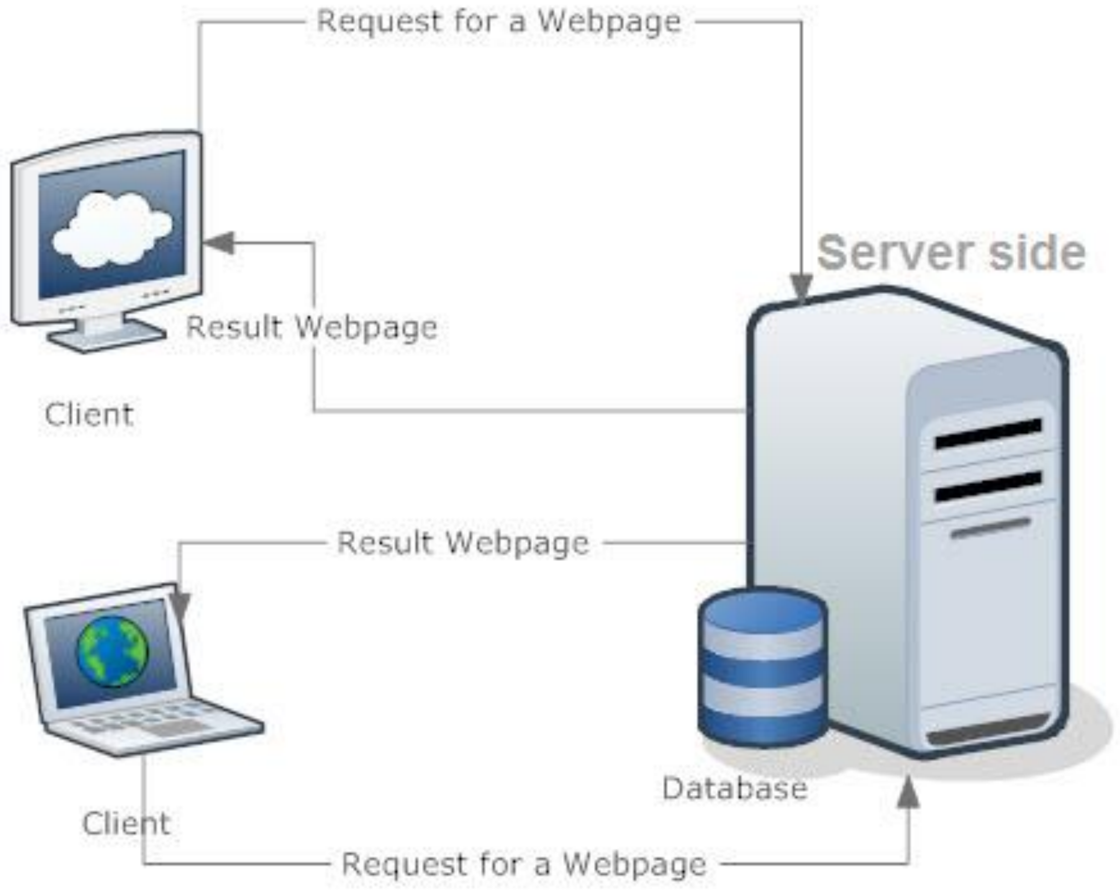
Learning is the process of acquiring new understanding, knowledge, behaviors, skills, values, attitudes, and preferences.

JavaScript Foundation *for* ReactJS

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

1 - Client Side Scripting

Client Side Scripting



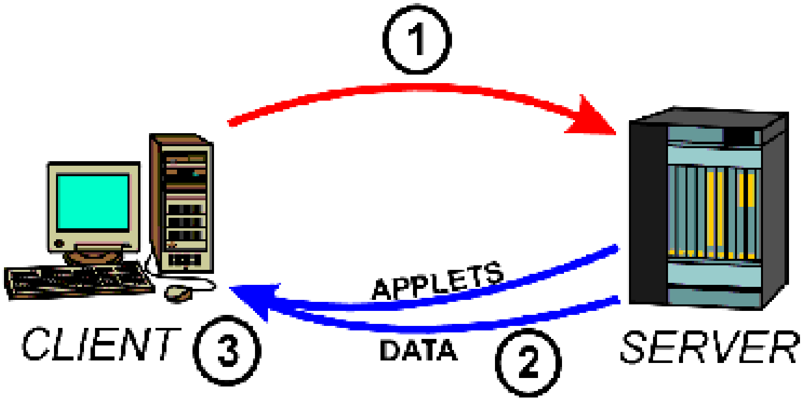
Client-side Scripting adalah bahasa pemrograman web yang pengolahan datanya dilakukan oleh komputer pengguna/pengunjung.

Jadi, ketika *users* berkunjung ke sebuah website, maka computer pengguna akan **mendownload data/script yang bersifat client-side di web** tersebut.

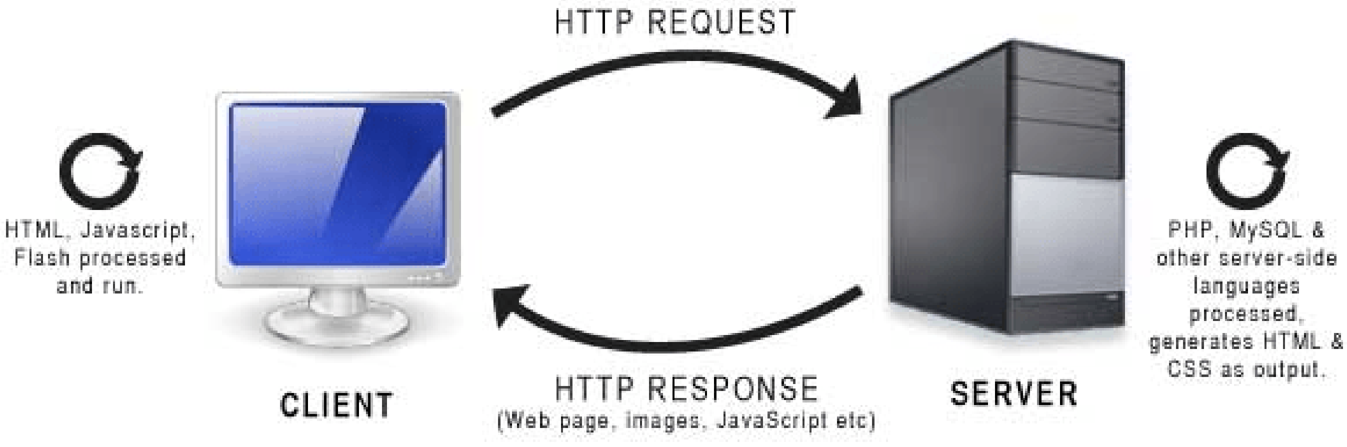
Contoh client side scripting adalah CSS, JavaScript, JQuery, HTML, XML dan HTML.

Contoh **server side scripting** adalah: JSP (Java Server Pages), ASP (Active Server Pages), PHP (Hypertext Preprocessor), Server Side Includes (SSI), Lasso, dan ColdFusion.

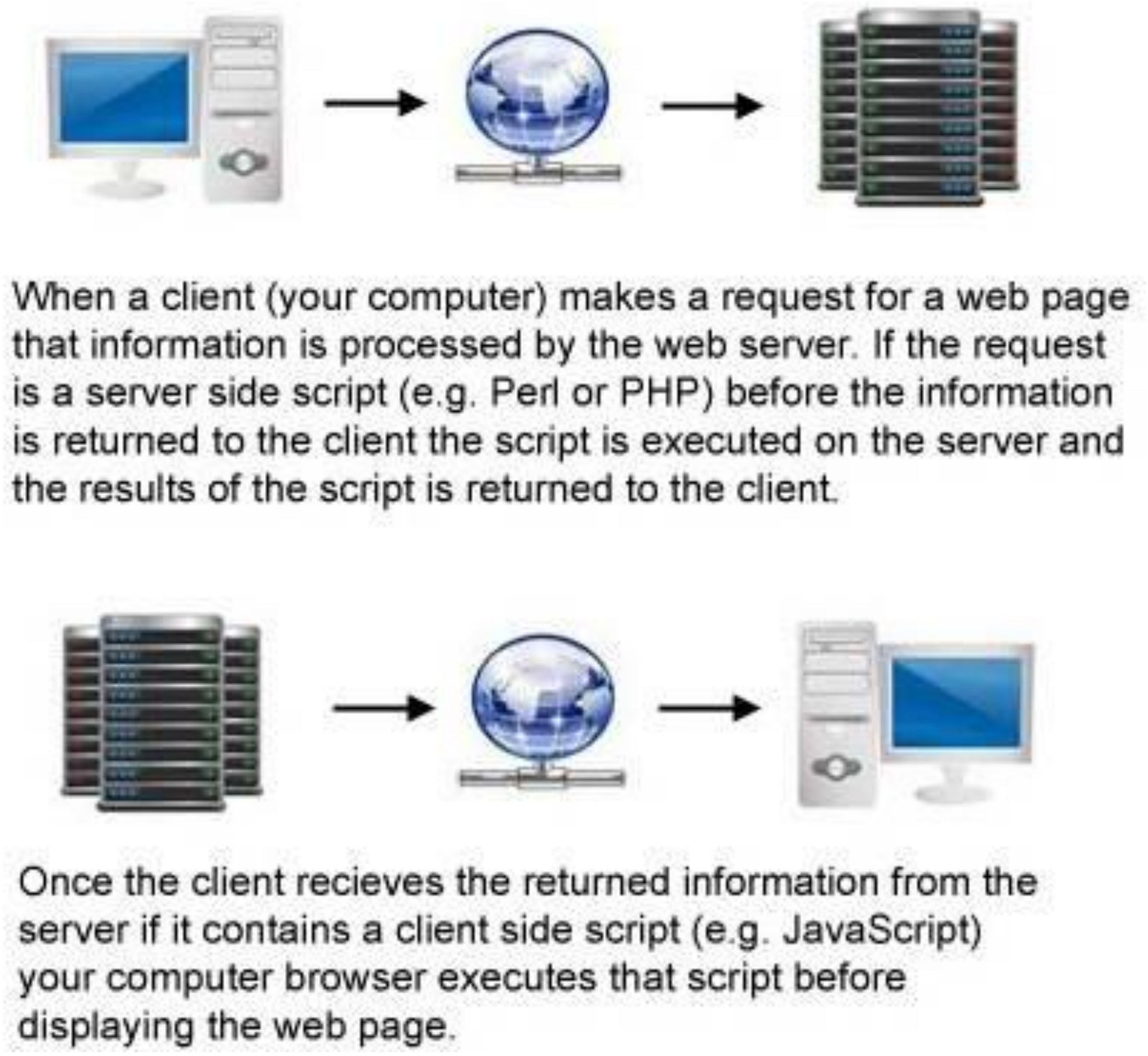
Client-side Configuration



1. Client sends request to server
2. Server processes request and returns information as needed
3. Data is processed by client's computer


HTTP REQUEST
HTML, Javascript, Flash processed and run.
CLIENT
HTTP RESPONSE
(Web page, images, JavaScript etc)
SERVER
PHP, MySQL & other server-side languages processed, generates HTML & CSS as output.

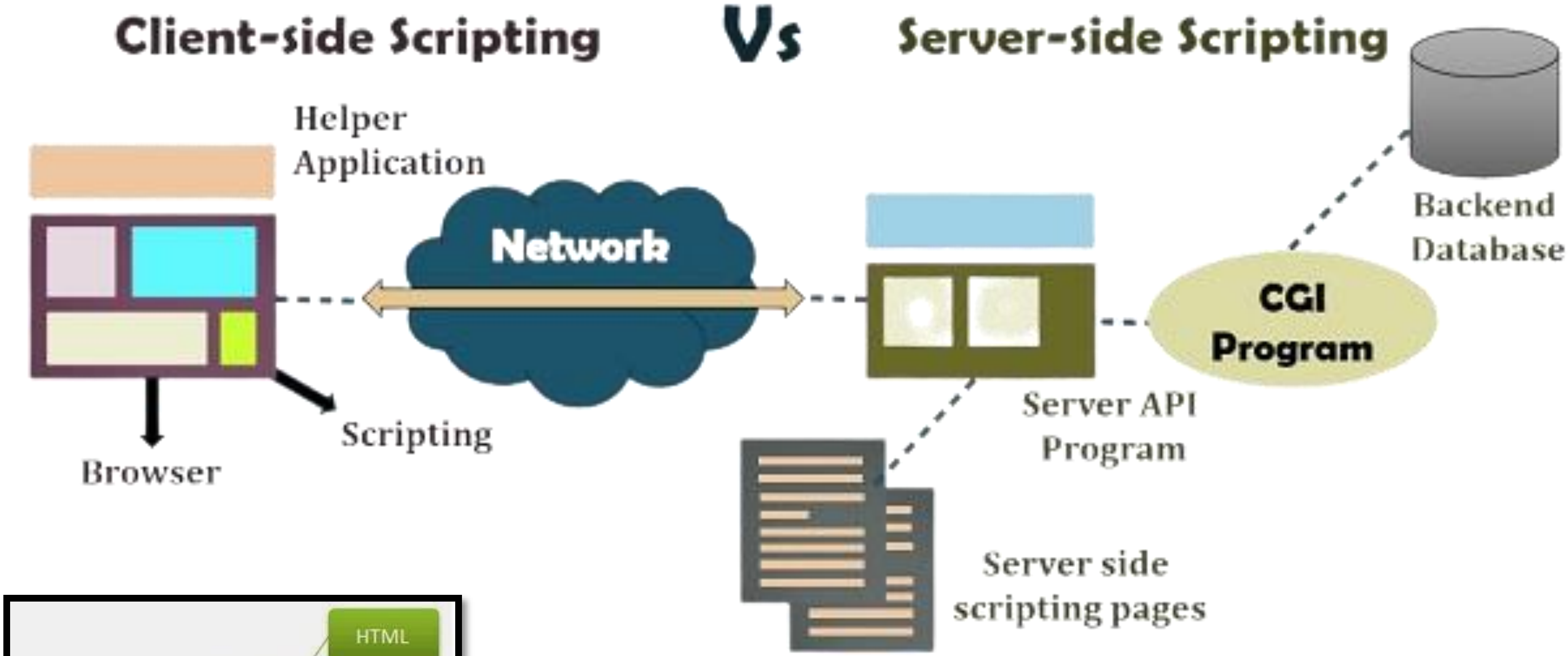
Client Side *vs* Server Side

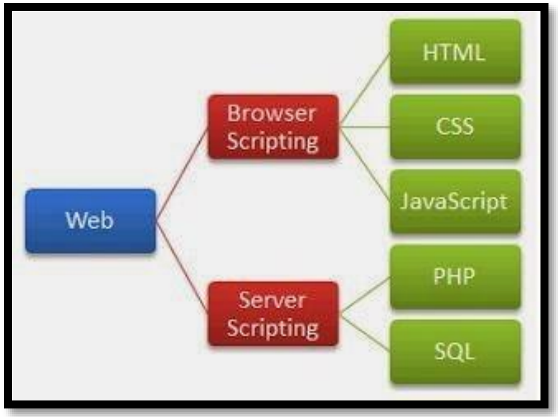


When a client (your computer) makes a request for a web page that information is processed by the web server. If the request is a server side script (e.g. Perl or PHP) before the information is returned to the client the script is executed on the server and the results of the script is returned to the client.

Once the client recieves the returned information from the server if it contains a client side script (e.g. JavaScript) your computer browser executes that script before displaying the web page.

Client Side *vs* Server Side Scripts

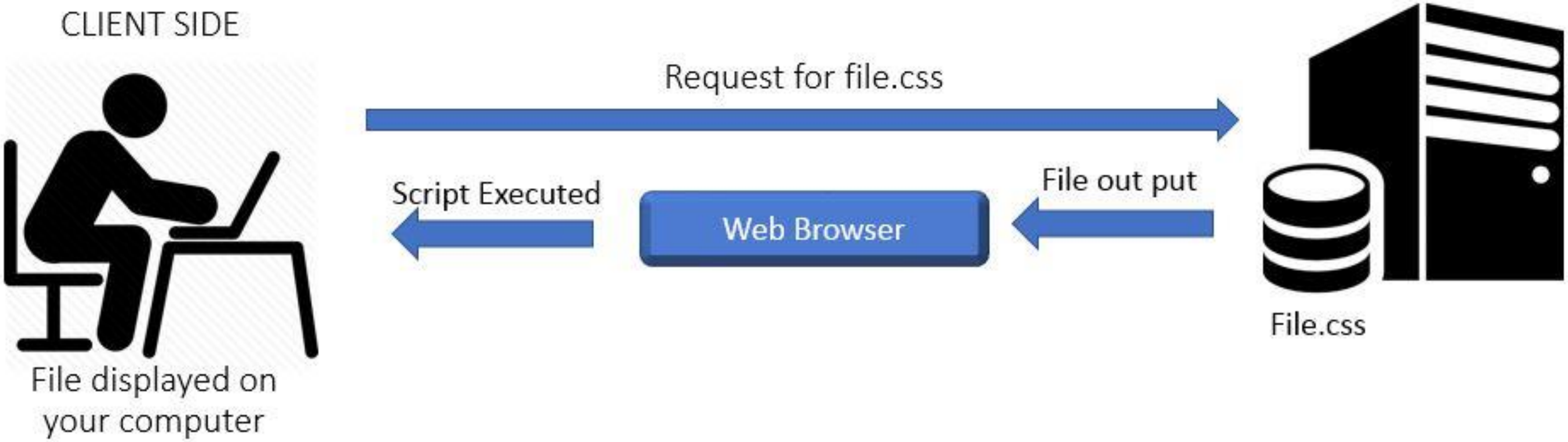


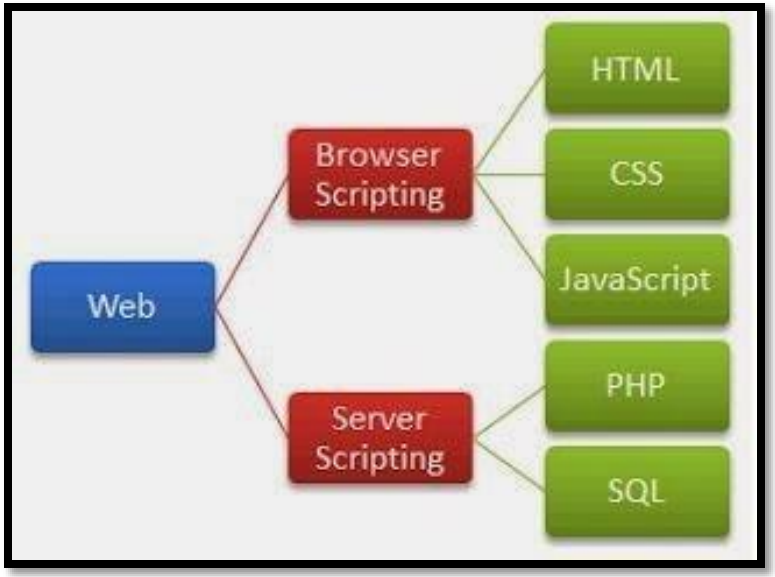


What is CGI Web page?

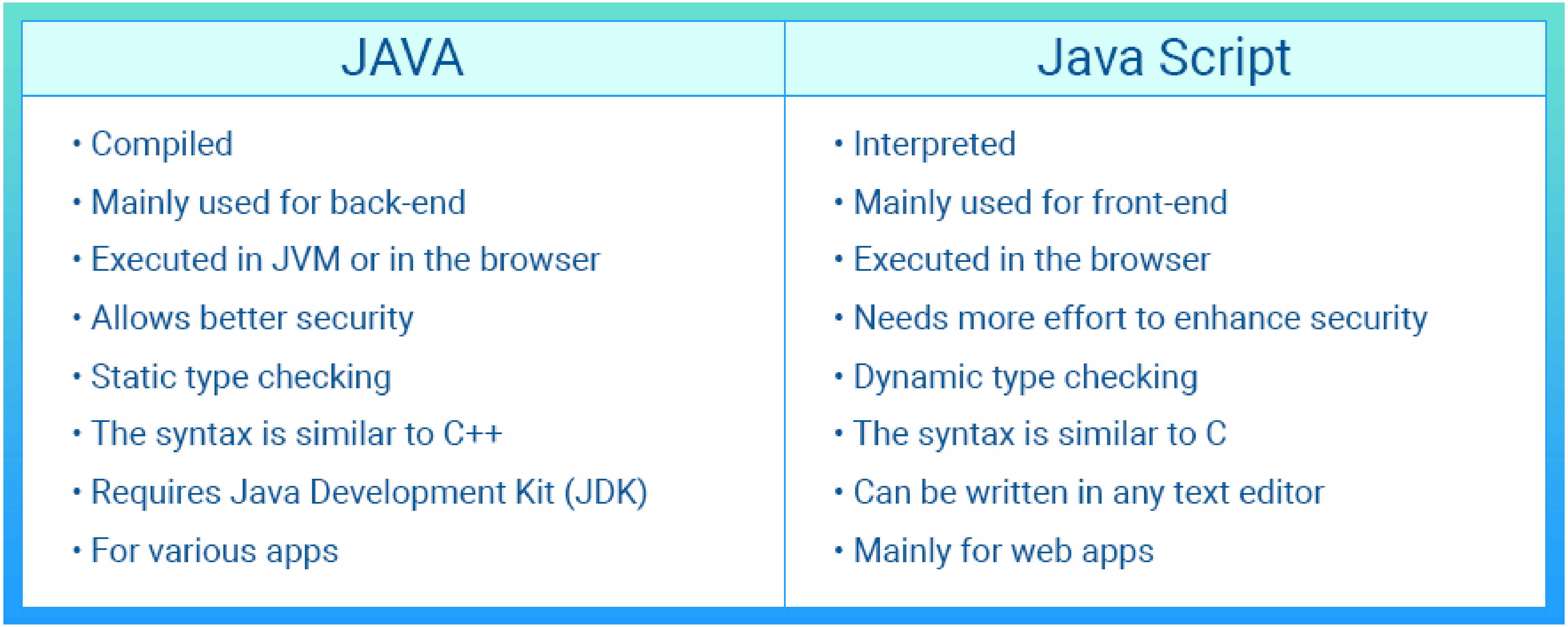
The common gateway interface (CGI) is a standard way for a Web server to pass a Web user's request to an application program and to receive data back to forward to the *user*.

Process of Client Side Scripting





JavaScript *vs* Java



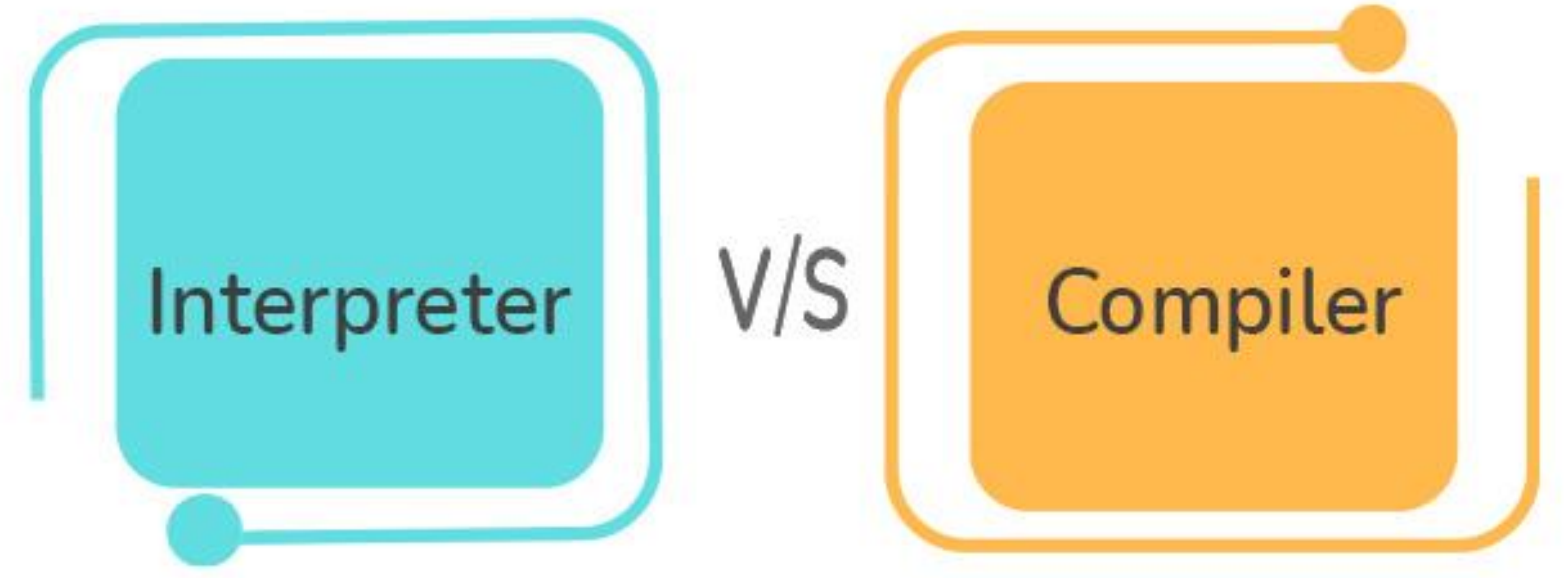
JAVA	Java Script
• Compiled	• Interpreted
• Mainly used for back-end	• Mainly used for front-end
• Executed in JVM or in the browser	• Executed in the browser
• Allows better security	• Needs more effort to enhance security
• Static type checking	• Dynamic type checking
• The syntax is similar to C++	• The syntax is similar to C
• Requires Java Development Kit (JDK)	• Can be written in any text editor
• For various apps	• Mainly for web apps

❖ JavaScript and Java are completely different languages, both in concept and design.

❖ JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997.

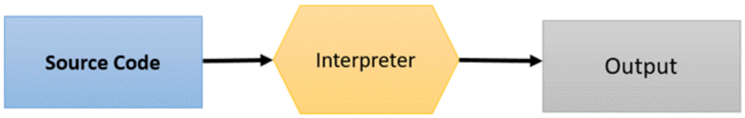
❖ ECMA-262 is the official name of the standard. ECMAScript is the official name of the language.

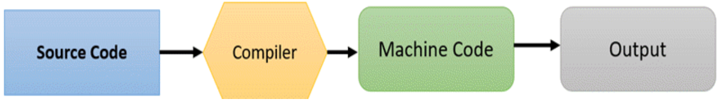
❖ A *compiler* translates the entire source code in a single run. An *interpreter* translates the entire source code line by line.



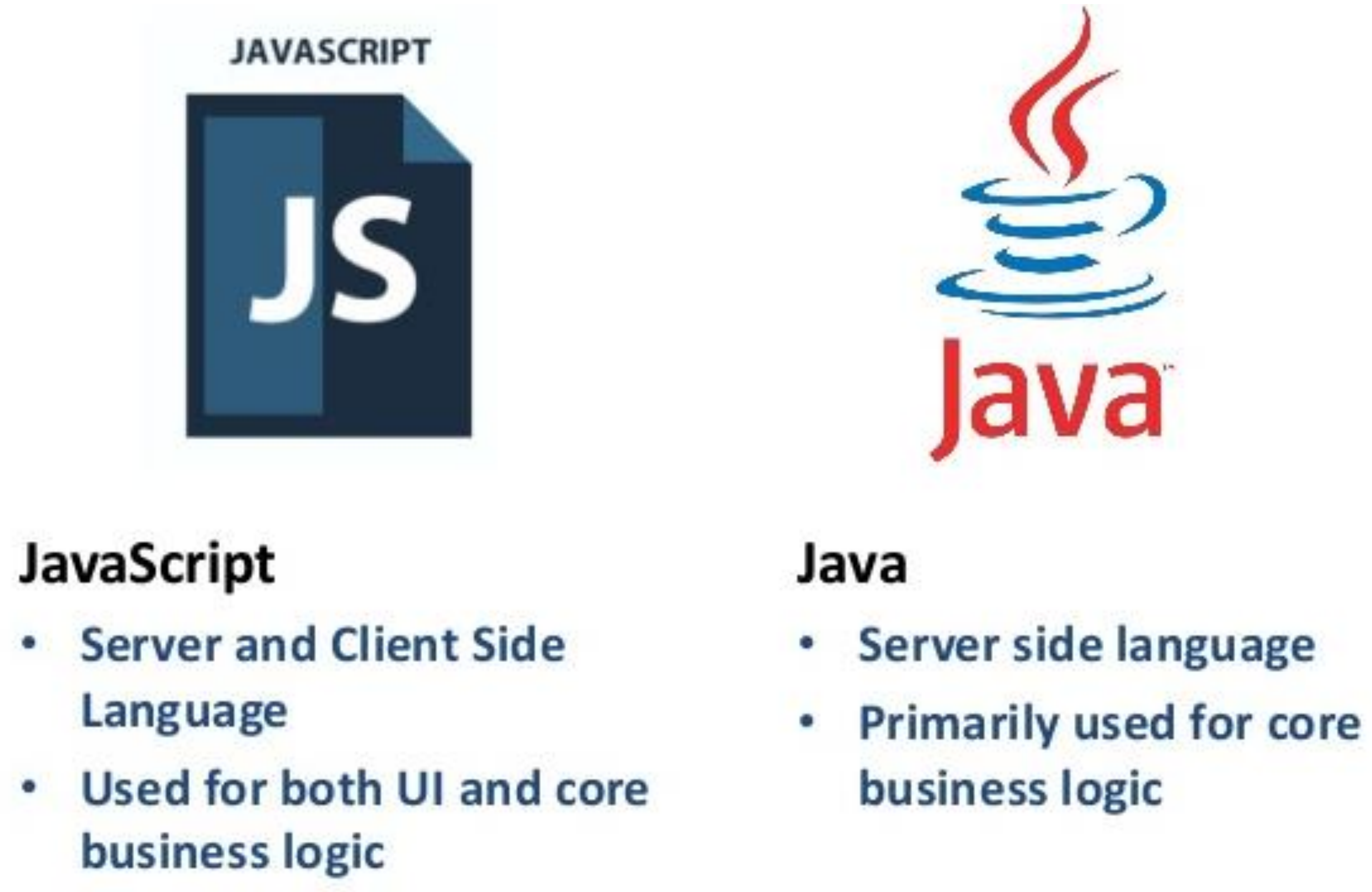
Interpreter mengkonversi *source code* menjadi *machine code* secara langsung ketika program dijalankan.

Pada compiler, *source code* akan dikonversi menjadi *machine code* sebelum program tersebut dijalankan.


Source Code → Interpreter → Output


Source Code → Compiler → Machine Code → Output

JavaScript *vs* Java


JAVASCRIPT
JS
Java

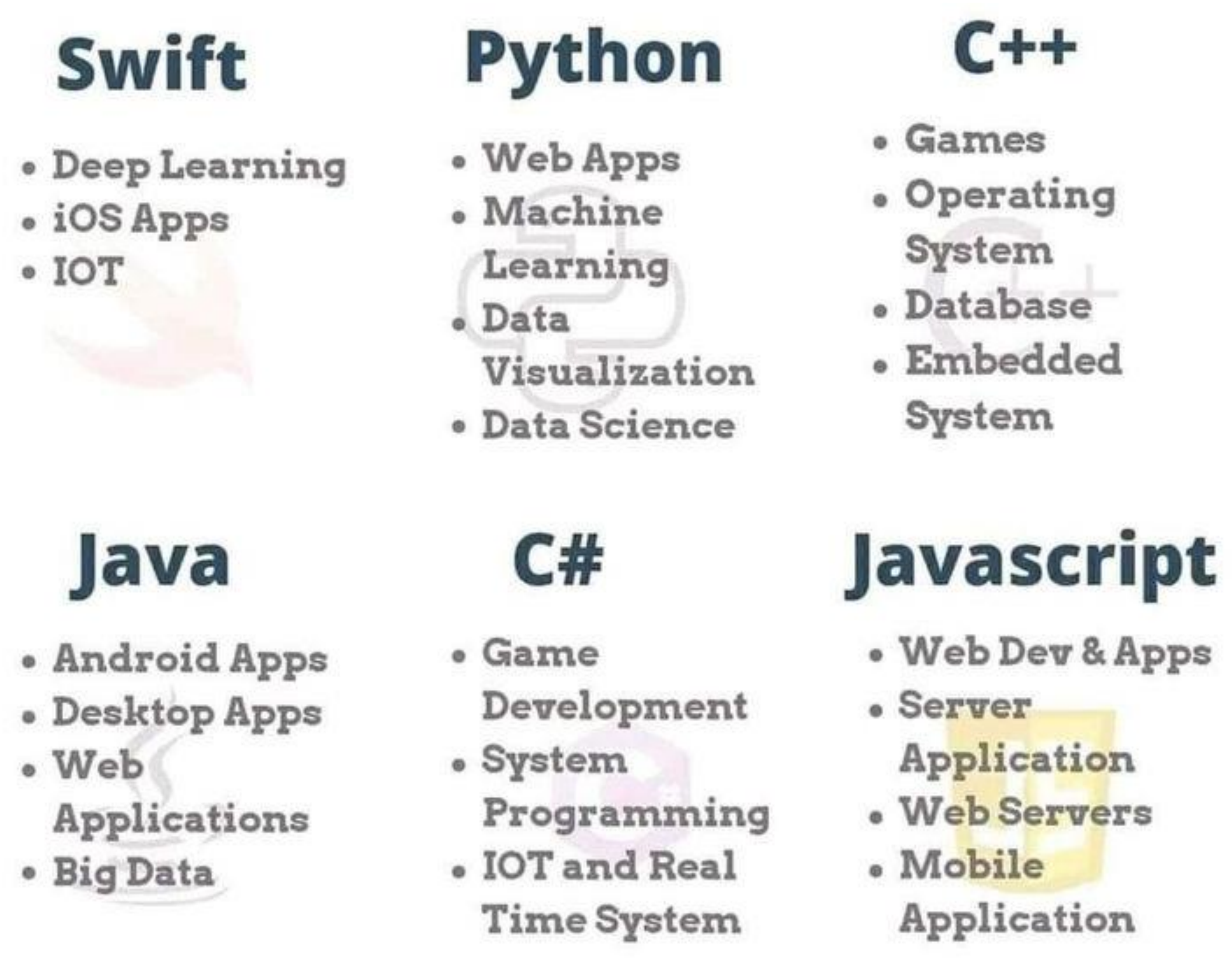
JavaScript

- Server and Client Side Language
- Used for both UI and core business logic

Java

- Server side language
- Primarily used for core business logic

Use Programming Languages



Swift

- Deep Learning
- iOS Apps
- IOT

Python

- Web Apps
- Machine Learning
- Data Visualization
- Data Science

C++

- Games
- Operating System
- Database
- Embedded System

Java

- Android Apps
- Desktop Apps
- Web Applications
- Big Data

C#

- Game Development
- System Programming
- IOT and Real Time System

Javascript

- Web Dev & Apps
- Server Application
- Web Servers
- Mobile Application

Feature *of* JavaScript

- **Scripting language and not Java:** In fact, JavaScript has nothing to do with Java. Then why is it called "Java" Script? When JavaScript was first released it was called Mocha, it was later renamed to LiveScript and then to JavaScript when Netscape (founded JavaScript) and Sun did a license agreement.

- **Object-based scripting language** which supports polymorphism, encapsulation and to some extent *inheritance* as well.

- **Interpreted language**: It doesn't have to be compiled like Java and C which require a compiler.

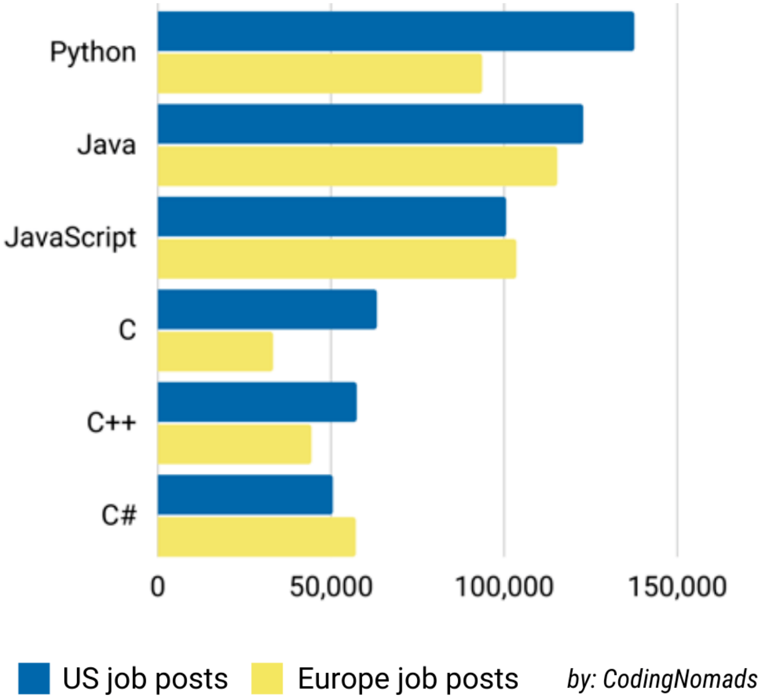
- **JavaScript runs in a browser**: We can run it on Google Chrome, Internet Explorer, Safari, etc. JavaScript can execute not only in the browser but also on the *server* and any device which has a JavaScript Engine.

2 – Why JavaScript ???

Intro. JavaScript

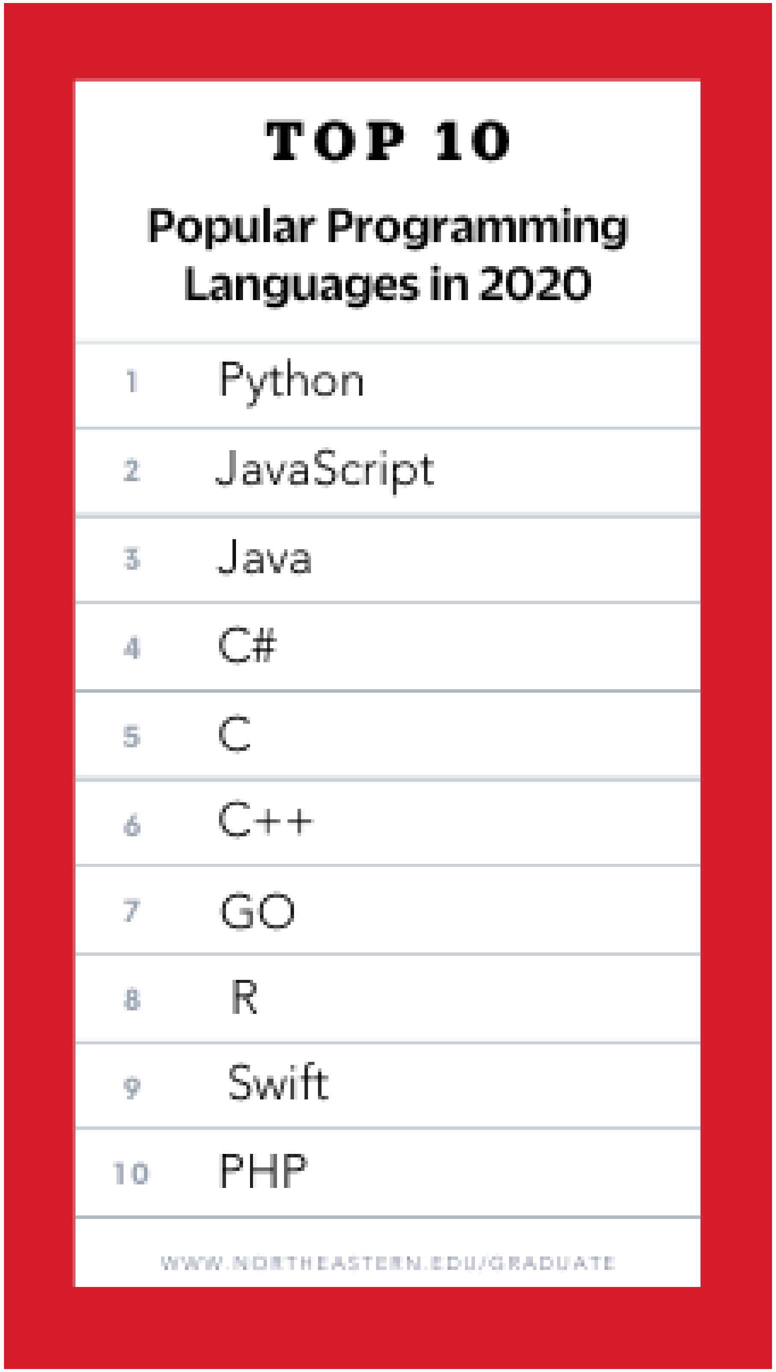
1) JavaScript is the world's most popular programming language.
2) JavaScript is the programming language of the Web.
3) JavaScript is easy to learn.

Most in-demand programming languages 2021-2022



Why Study JavaScript?
JavaScript is one of the 3 languages all web developers must learn:

1. HTML to define the content of web pages
2. CSS to specify the layout of web pages
3. JavaScript to program the behavior of web pages



TOP 10

Popular Programming Languages in 2020

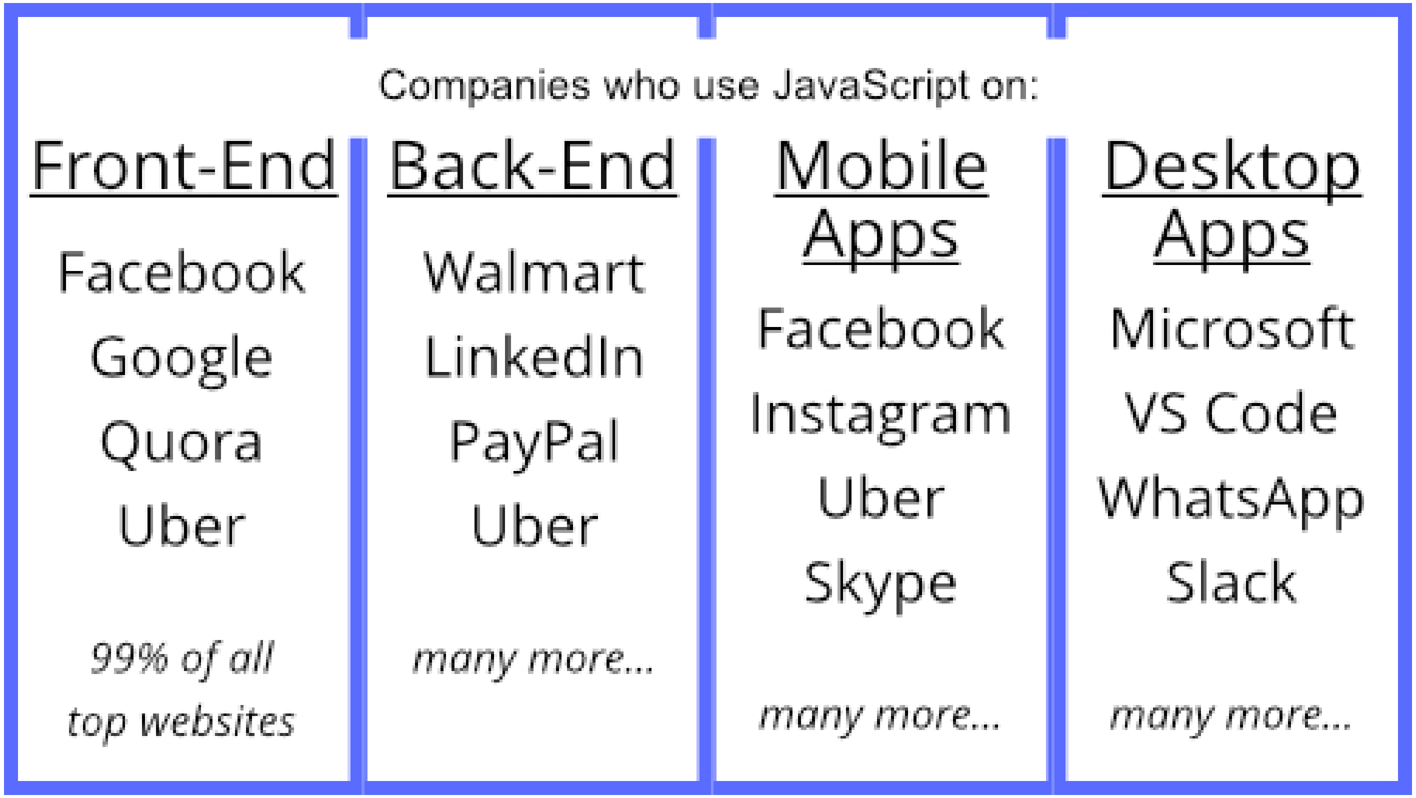
1	Python
2	JavaScript
3	Java
4	C#
5	C
6	C++
7	GO
8	R
9	Swift
10	PHP

WWW.NORTHEASTERN.EDU/GRADUATE

What *can* we do with Javascript?

Why Javascript?

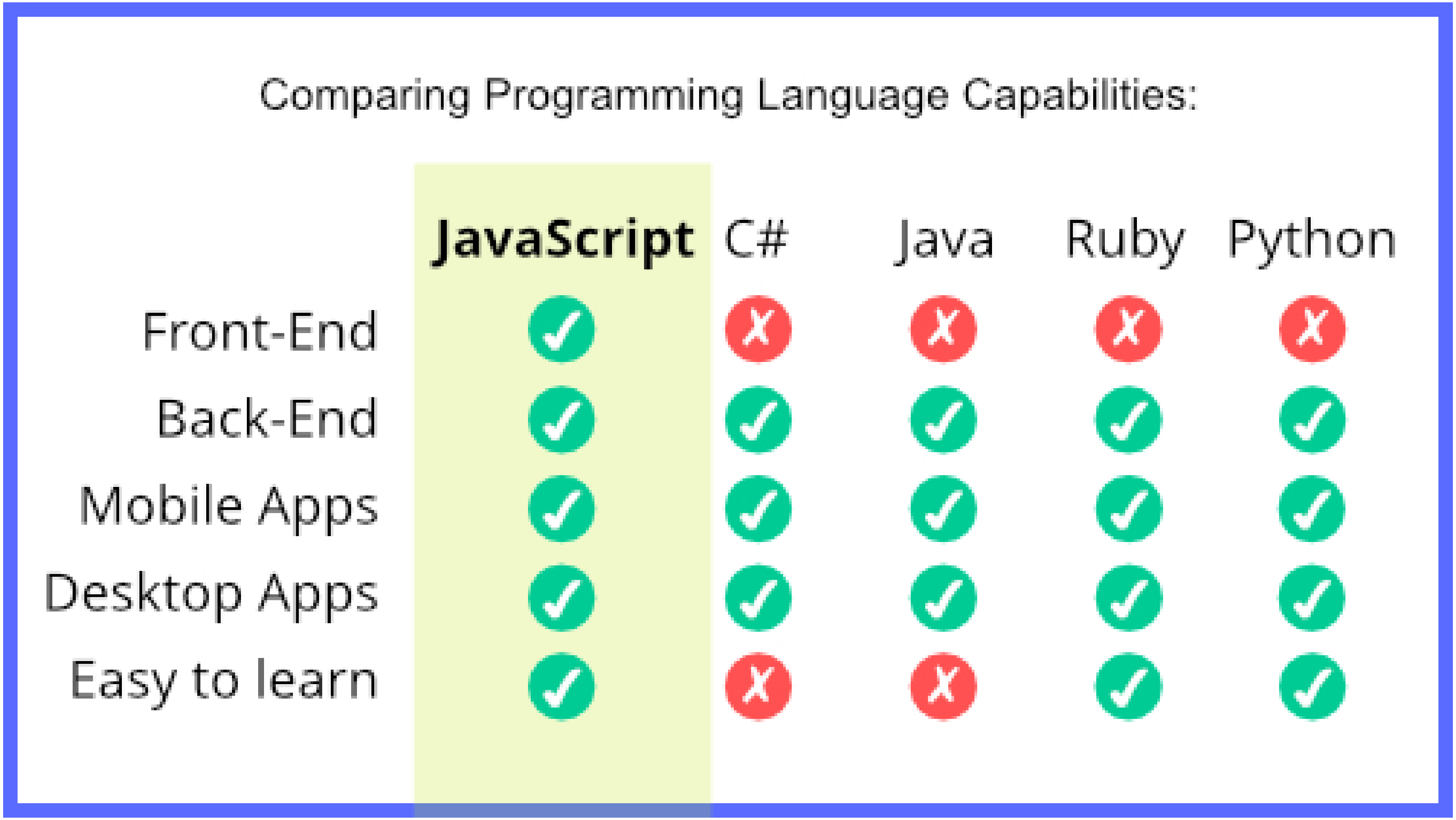
1) JavaScript Can Change HTML Content
2) JavaScript Can Change HTML Attribute Values
3) JavaScript Can Change HTML Styles (CSS)
4) JavaScript Can Hide HTML Elements
5) JavaScript Can Show HTML Elements



Companies who use JavaScript on:

Front-End	Back-End	Mobile Apps	Desktop Apps
Facebook	Walmart	Facebook	Microsoft
Google	LinkedIn	Instagram	VS Code
Quora	PayPal	Uber	WhatsApp
Uber	Uber	Skype	Slack
99% of all top websites	*many more...*	*many more...*	*many more...*

JavaScript Capabilities



Comparing Programming Language Capabilities:

	JavaScript	C#	Java	Ruby	Python
Front-End	✓	✗	✗	✗	✗
Back-End	✓	✓	✓	✓	✓
Mobile Apps	✓	✓	✓	✓	✓
Desktop Apps	✓	✓	✓	✓	✓
Easy to learn	✓	✗	✗	✓	✓

3 – JavaScript Fundamentals

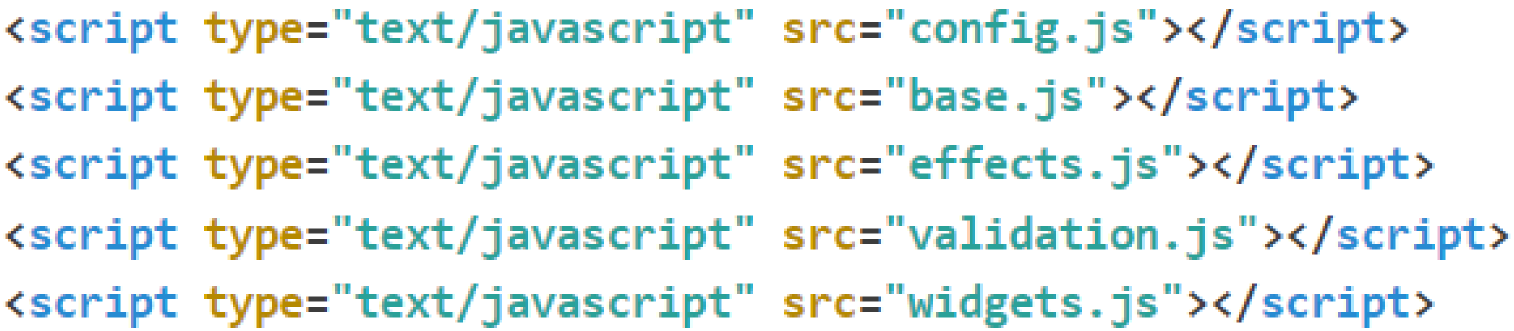
How *to* execute Javascript?

JavaScript is a text-based language that does not need any conversion before being executed. Other languages like Java and C++ need to be compiled to be executable but JavaScript is executed instantly by a type of program that interprets the code called a parser (*pretty much all web browsers contain a JavaScript parser*).

To execute JavaScript in a browser you have two options:

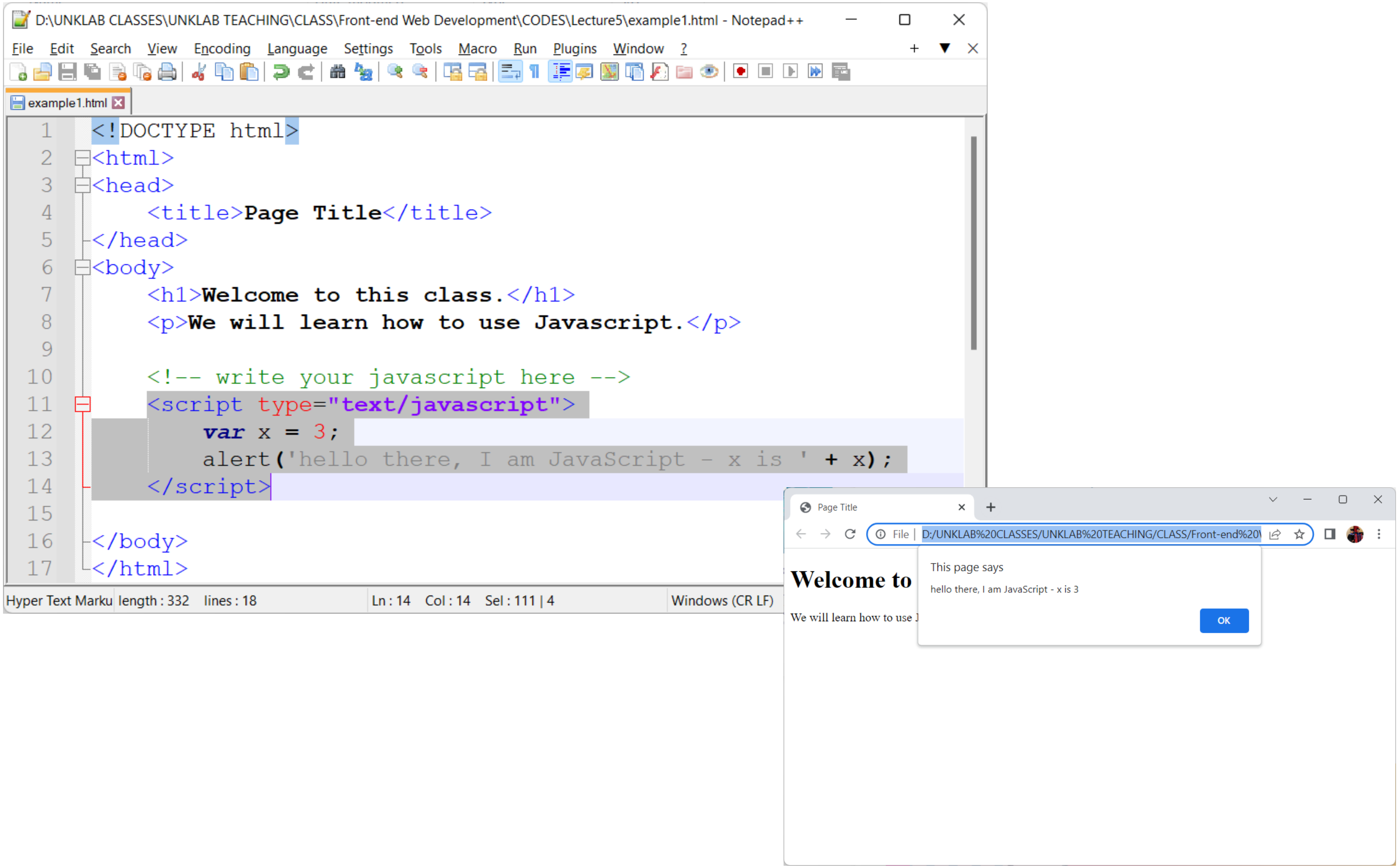
(1) put *it* inside a <script> element anywhere inside an HTML document, or

(2) put *it* inside an external JavaScript file (*with a .js extension*) and then reference that file inside the HTML document using an empty <script> element with a *src* attribute.



```
<script type="text/javascript" src="config.js"></script>
<script type="text/javascript" src="base.js"></script>
<script type="text/javascript" src="effects.js"></script>
<script type="text/javascript" src="validation.js"></script>
<script type="text/javascript" src="widgets.js"></script>
```

(1) Include JavaScript Inside HTML



D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\example1.html - Notepad++

```html
<!DOCTYPE html>
<html>
<head>
    <title>Page Title</title>
</head>
<body>
    <h1>Welcome to this class.</h1>
    <p>We will learn how to use Javascript.</p>

    <!-- write your javascript here -->
    <script type="text/javascript">
        var x = 3;
        alert('hello there, I am JavaScript - x is ' + x);
    </script>

</body>
</html>
```

Hyper Text Marku length : 332 lines : 18 Ln : 14 Col : 14 Sel : 111 | 4 Windows (CR LF)

Welcome to

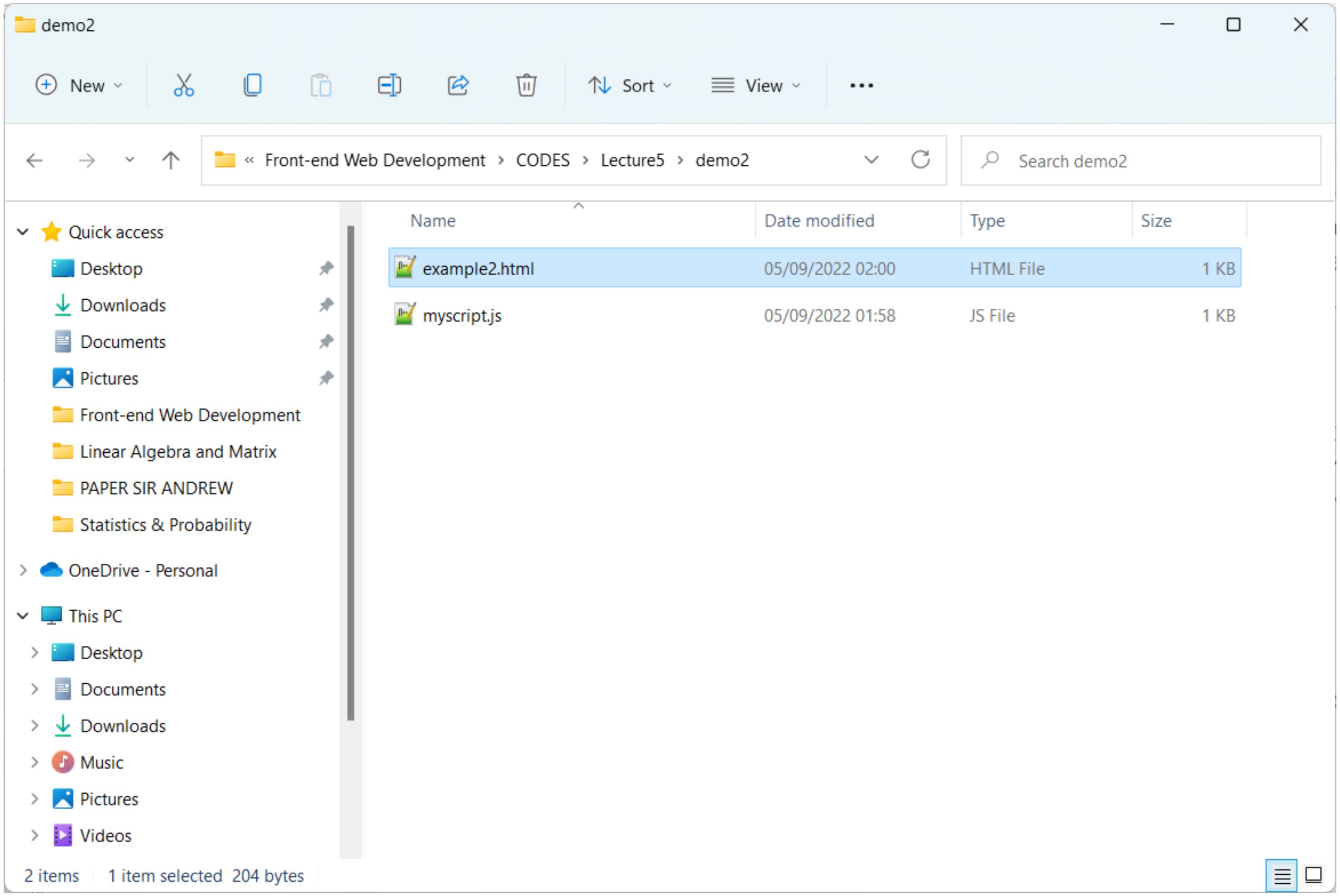
We will learn how to use J

This page says

hello there, I am JavaScript - x is 3

(2) External JavaScript File

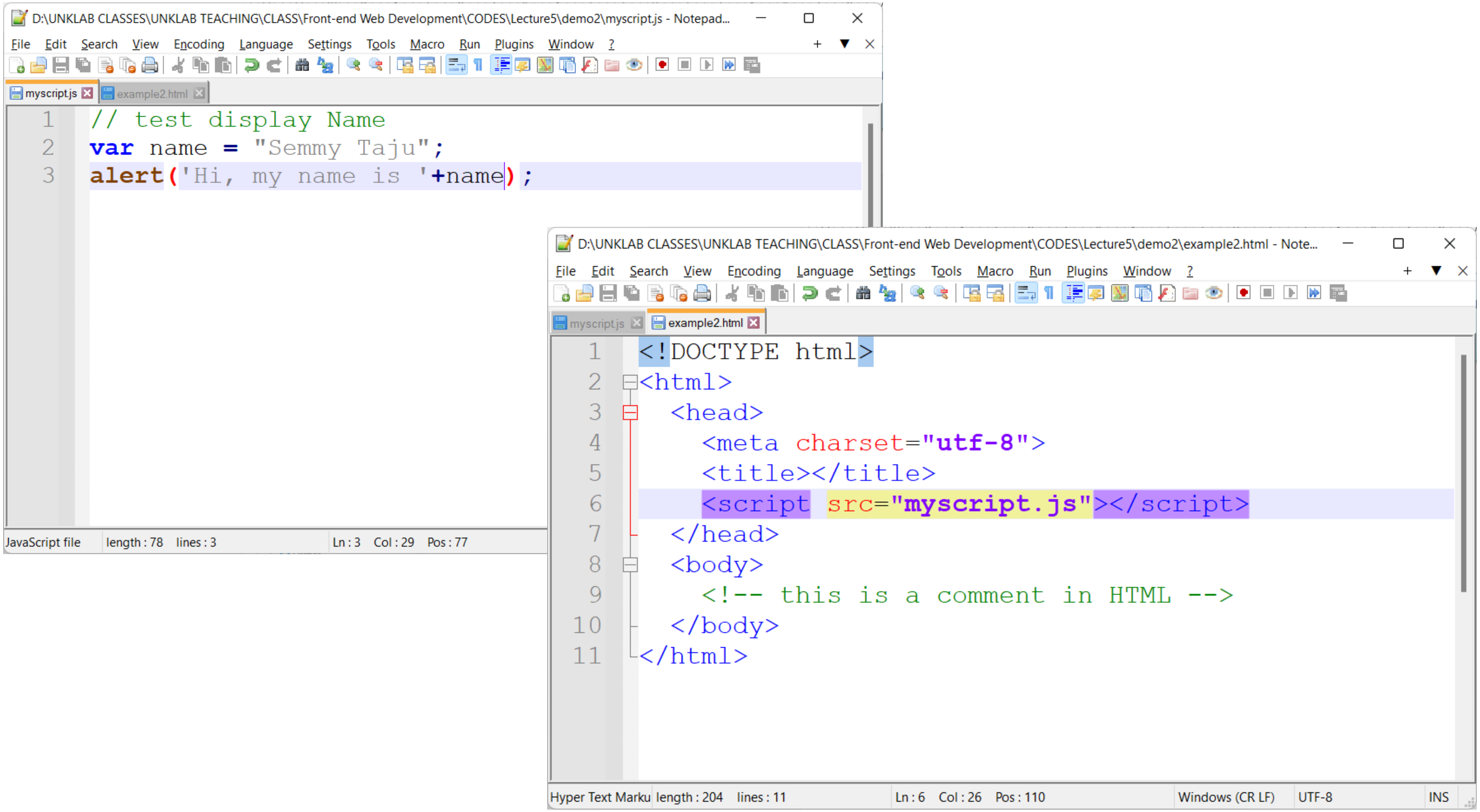
Create two files in your directory.



Name	Date modified	Type	Size
example2.html	05/09/2022 02:00	HTML File	1 KB
myscript.js	05/09/2022 01:58	JS File	1 KB

(2) External JavaScript File

Write down these Javascript and HTML codes. The classic best practice for placing scripts was in the *head* of the document:

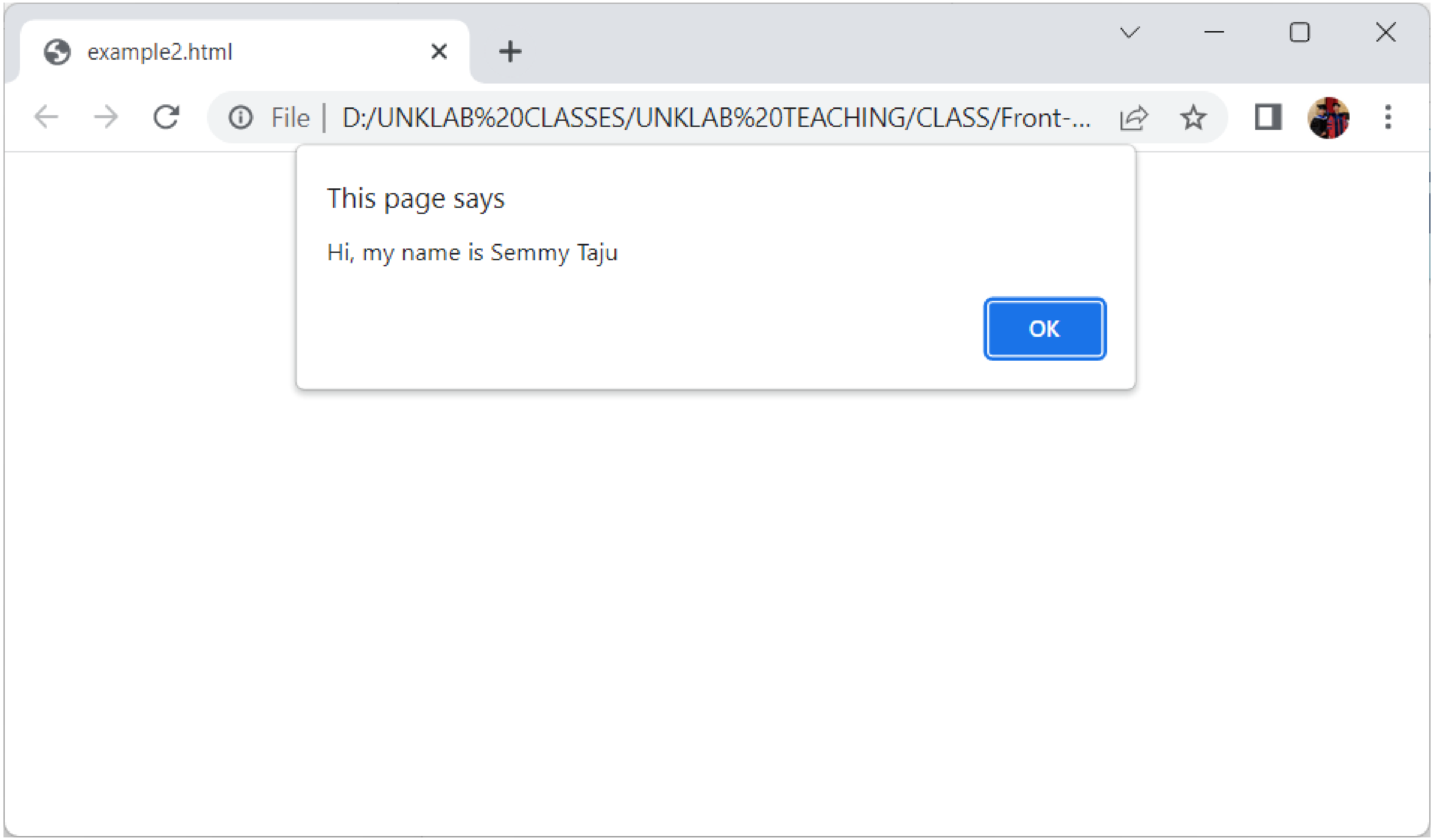

D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo2\myscript.js - Notepad...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscript.js
example2.html
JavaScript file
length : 78
lines : 3
Ln : 3
Col : 29
Pos : 77
D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo2\example2.html - Note...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscript.js
example2.html
Hyper Text Marku
length : 204
lines : 11
Ln : 6
Col : 26
Pos : 110
Windows (CR LF)
UTF-8
INS

```javascript
// test display Name
var name = "Semmy Taju";
alert('Hi, my name is '+name);
```

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title></title>
    <script src="myscript.js"></script>
  </head>
  <body>
    <!-- this is a comment in HTML -->
  </body>
</html>
```

(2) External JavaScript File

Output program should display your name using alert message dialog.



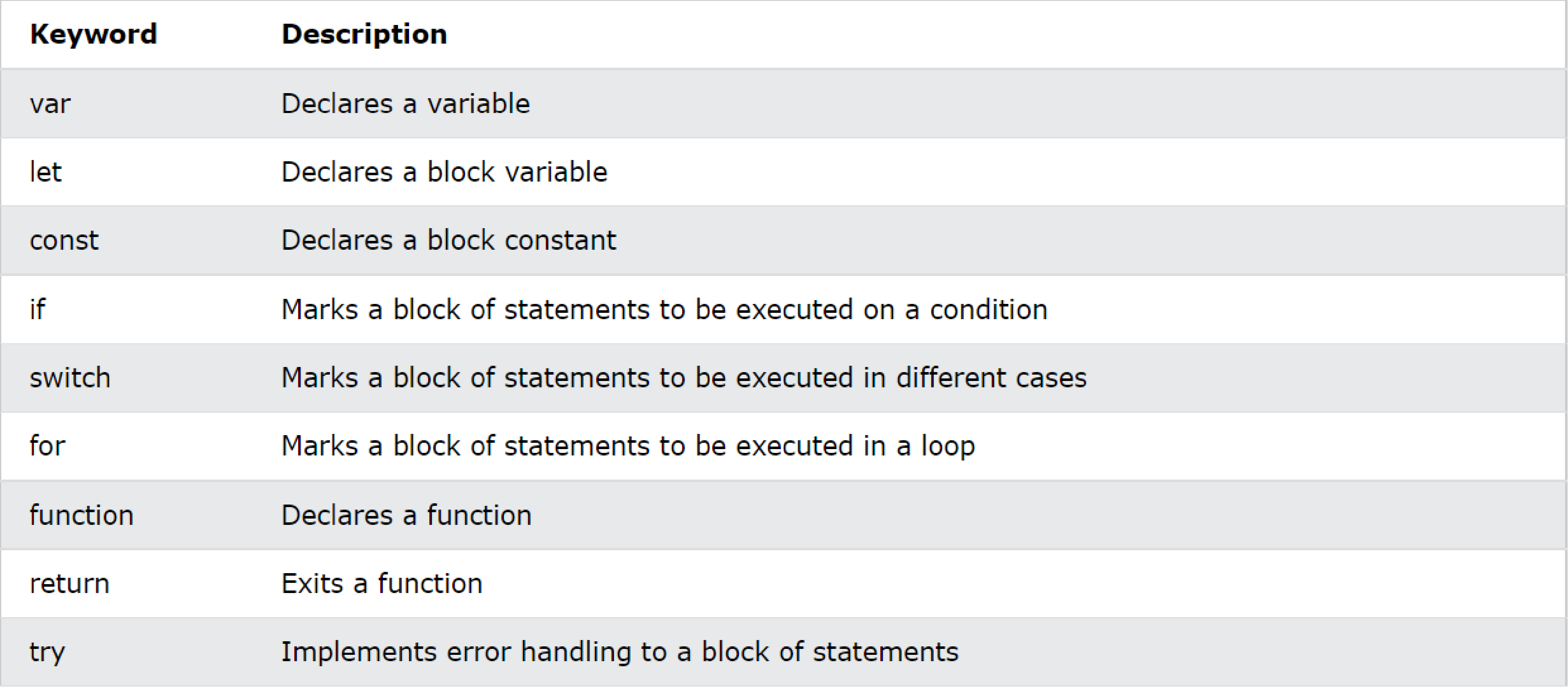
example2.html
File | D:/UNKLAB%20CLASSES/UNKLAB%20TEACHING/CLASS/Front-...

This page says

Hi, my name is Semmy Taju

4 – JavaScript Statements

JavaScript Keywords



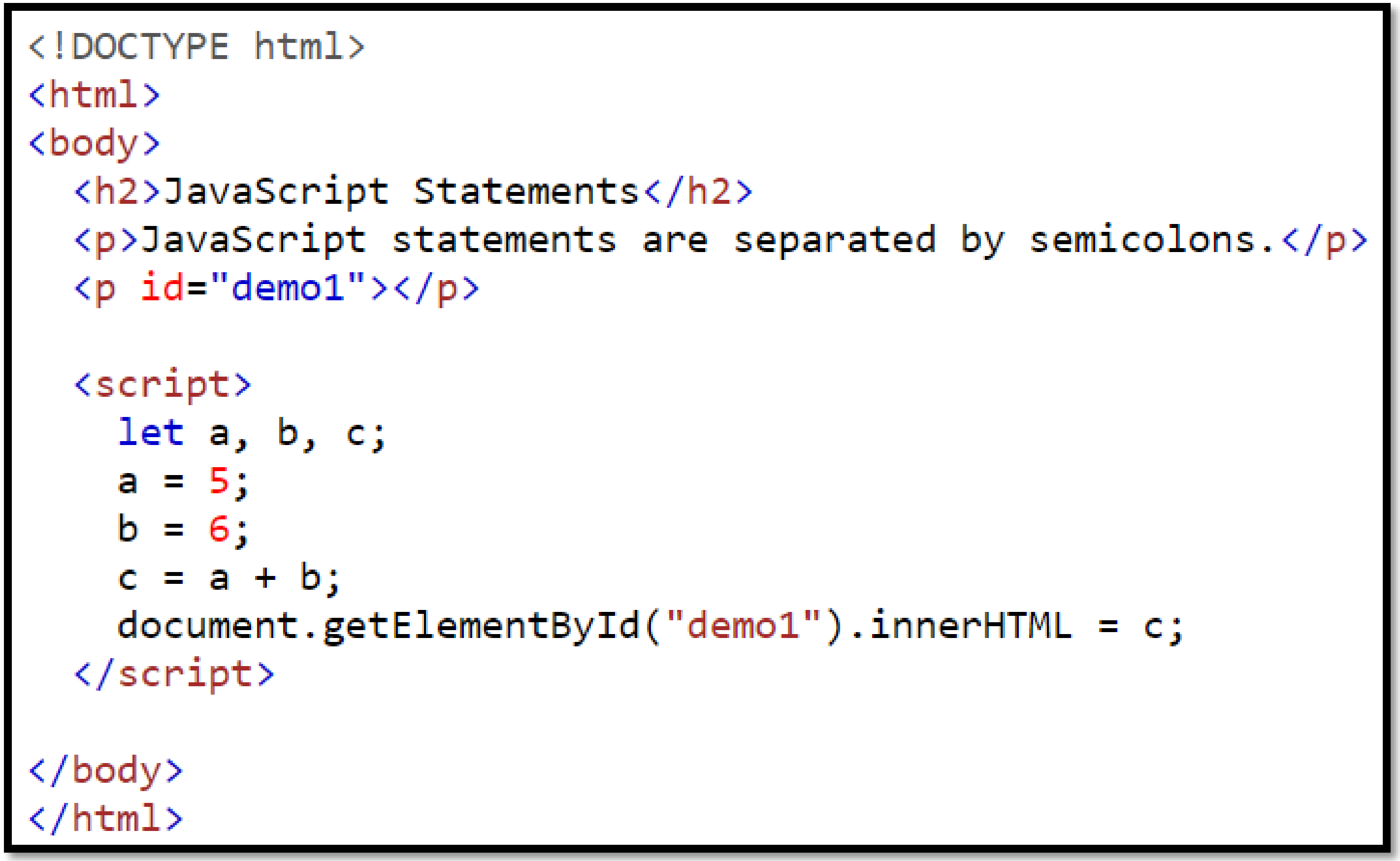
Keyword	Description
var	Declares a variable
let	Declares a block variable
const	Declares a block constant
if	Marks a block of statements to be executed on a condition
switch	Marks a block of statements to be executed in different cases
for	Marks a block of statements to be executed in a loop
function	Declares a function
return	Exits a function
try	Implements error handling to a block of statements

- ❖ JavaScript statements often start with a keyword to identify the JavaScript action to be performed.

- ❖ Let dan Const menganut sistem *block scope*, yang mana cakupan variabelnya hanya bisa diakses di dalam blocknya saja. Var menganut sistem functional scope, yang mana variabelnya dapat diakses dari dalam maupun dari luar block kecuali di luar function.

Semicolons in JavaScript Statements

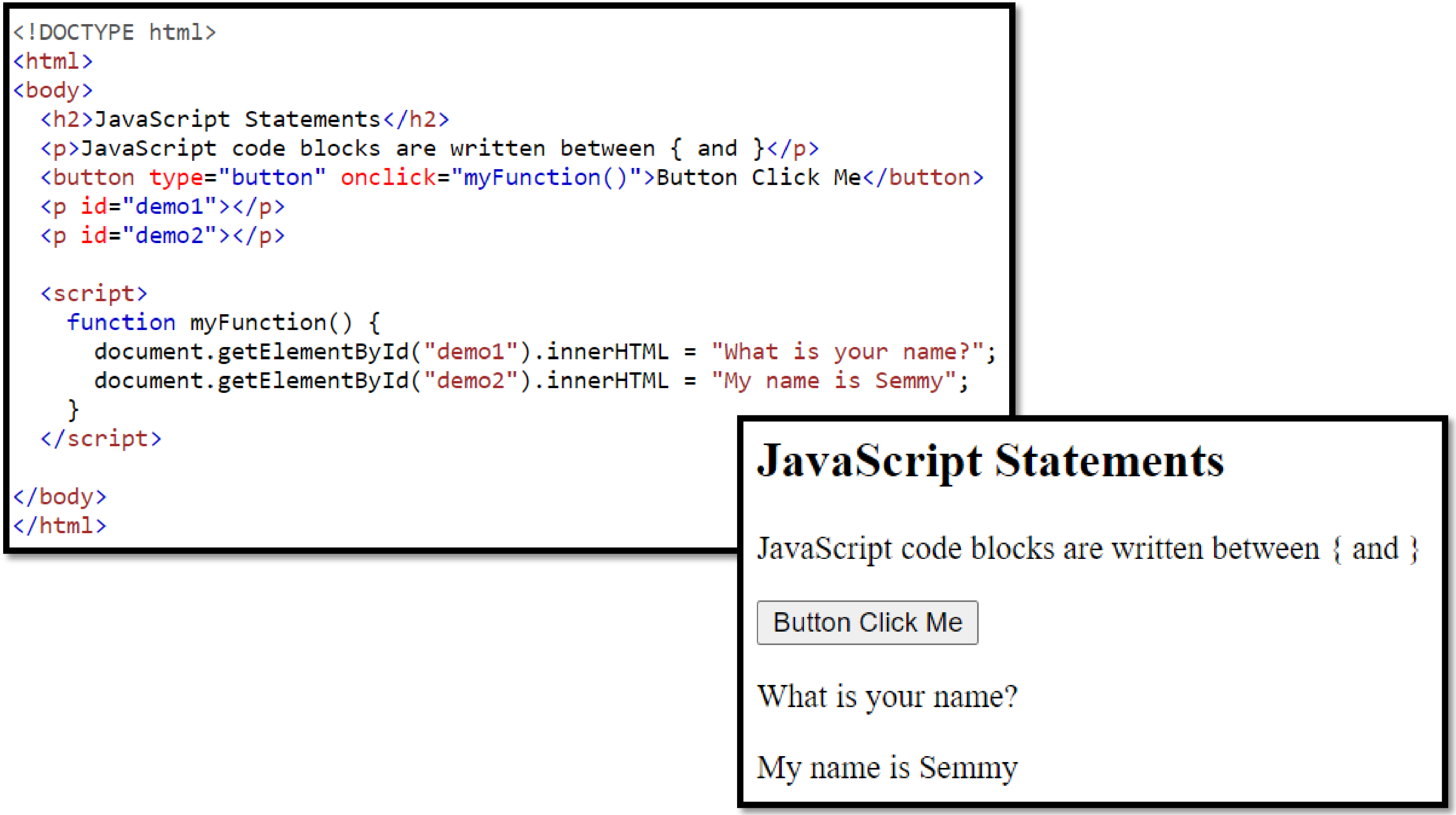
Semicolons separate JavaScript statements. Add a semicolon at the end of each executable statement. When separated by semicolons, multiple statements on one line are allowed.



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Statements</h2>
  <p>JavaScript statements are separated by semicolons.</p>
  <p id="demo1"></p>

  <script>
    let a, b, c;
    a = 5;
    b = 6;
    c = a + b;
    document.getElementById("demo1").innerHTML = c;
  </script>

</body>
</html>
```

JavaScript Code Blocks

JavaScript statements can be grouped together in code blocks, inside curly brackets {...}. The purpose of code blocks is to define statements to be executed together.



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Statements</h2>
  <p>JavaScript code blocks are written between { and }</p>
  <button type="button" onclick="myFunction()">Button Click Me</button>
  <p id="demo1"></p>
  <p id="demo2"></p>

  <script>
    function myFunction() {
      document.getElementById("demo1").innerHTML = "What is your name?";
      document.getElementById("demo2").innerHTML = "My name is Semmy";
    }
  </script>

</body>
</html>
```

JavaScript Statements

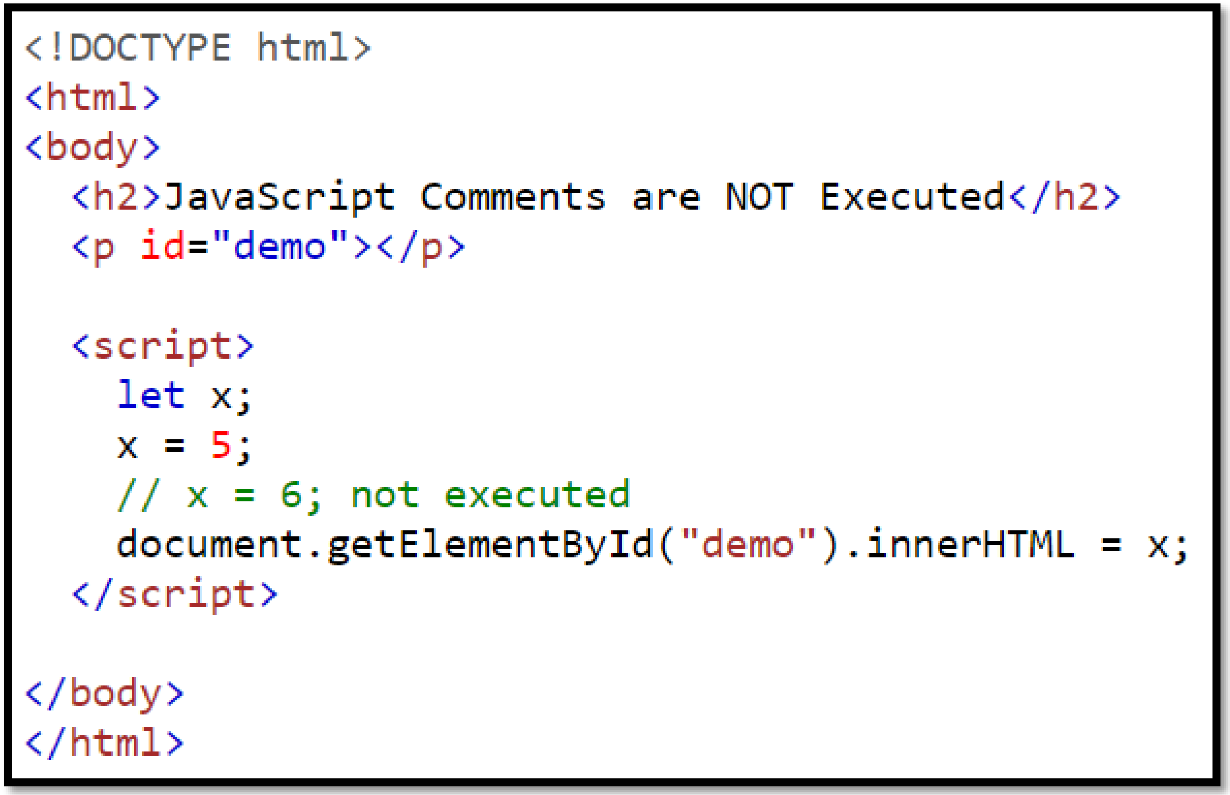
JavaScript code blocks are written between { and }

What is your name?

My name is Semmy

JavaScript Comments

- Not all JavaScript statements are "executed".
- Code after double slashes // or between /* and */ is treated as a comment.



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Comments are NOT Executed</h2>
  <p id="demo"></p>

  <script>
    let x;
    x = 5;
    // x = 6; not executed
    document.getElementById("demo").innerHTML = x;
  </script>

</body>
</html>
```

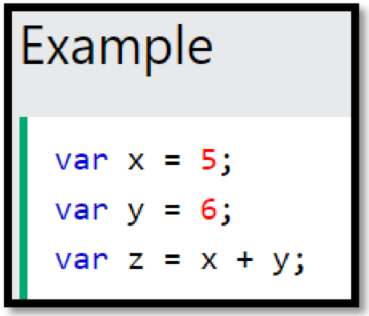
5 – JavaScript Variables

JavaScript Variables

Variables are containers for storing data (*storing data values*). In this example, x, y, and z, are variables, declared with the *var* keyword.

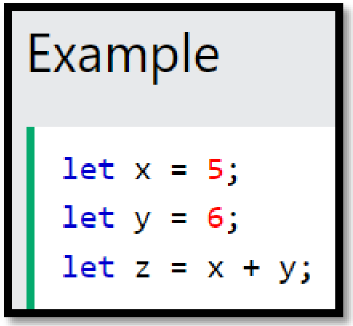
4 Ways to Declare a JavaScript Variable:

1) Using *var*
2) Using *let*
3) Using *const*
4) Using nothing



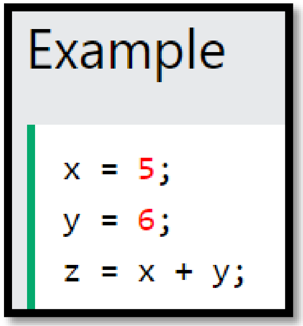
Example

```
var x = 5;
var y = 6;
var z = x + y;
```



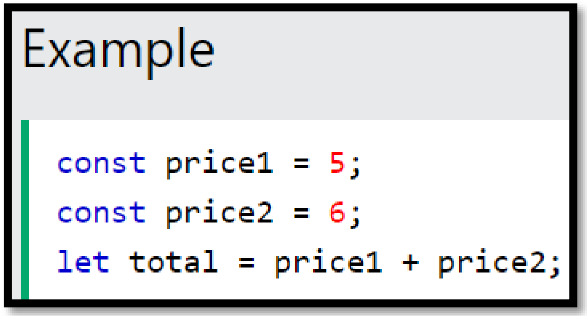
Example

```
let x = 5;
let y = 6;
let z = x + y;
```



Example

```
x = 5;
y = 6;
z = x + y;
```

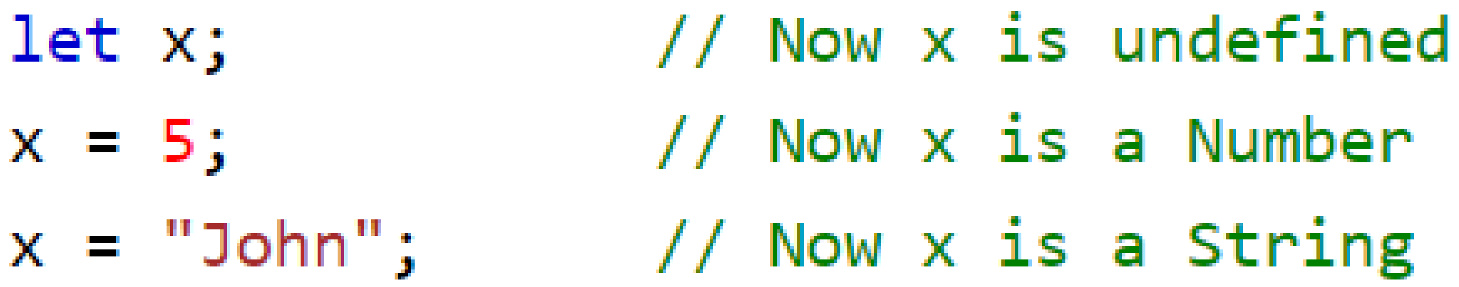


Example

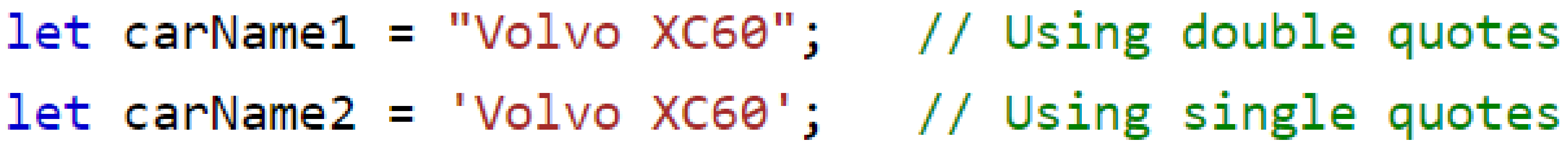
```
const price1 = 5;
const price2 = 6;
let total = price1 + price2;
```

JavaScript Types *are* Dynamic

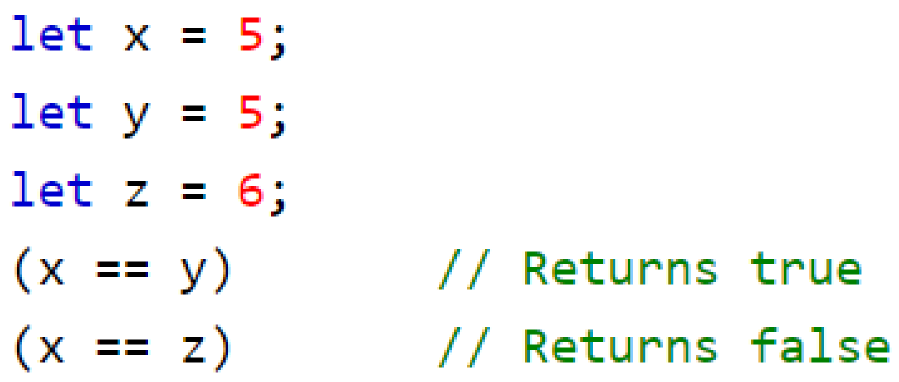
JavaScript has dynamic types. This means that the same variable can be used to hold different data types.



```
let x;           // Now x is undefined
x = 5;           // Now x is a Number
x = "John";      // Now x is a String
```



```
let carName1 = "Volvo XC60";    // Using double quotes
let carName2 = 'Volvo XC60';    // Using single quotes
```



```
let x = 5;
let y = 5;
let z = 6;
(x == y)       // Returns true
(x == z)       // Returns false
```

JavaScript Arrays

Why Use Arrays?

If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
let car1 = "Saab";
let car2 = "Volvo";
let car3 = "BMW";
```

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?

The solution is an array!

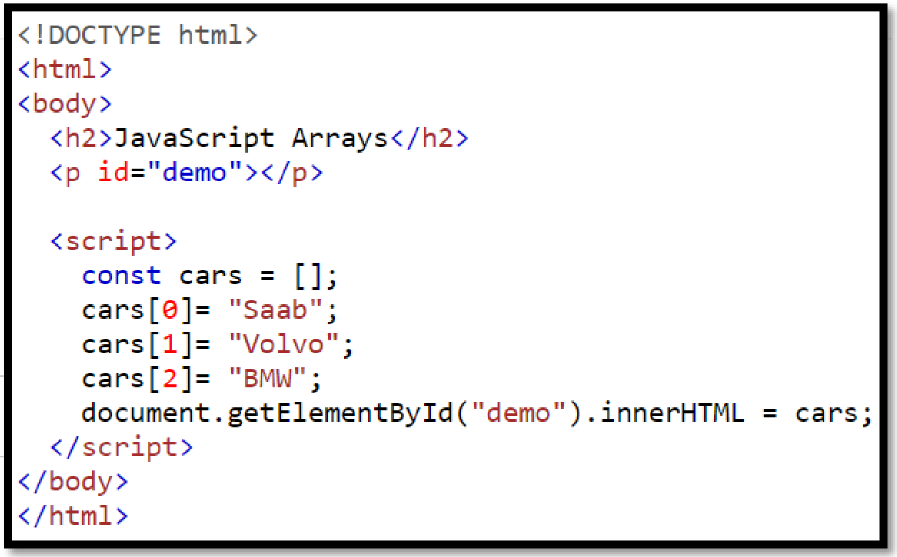
An array can hold many values under a single name, and you can access the values by referring to an index number.

Creating an Array

Using an array literal is the easiest way to create a JavaScript

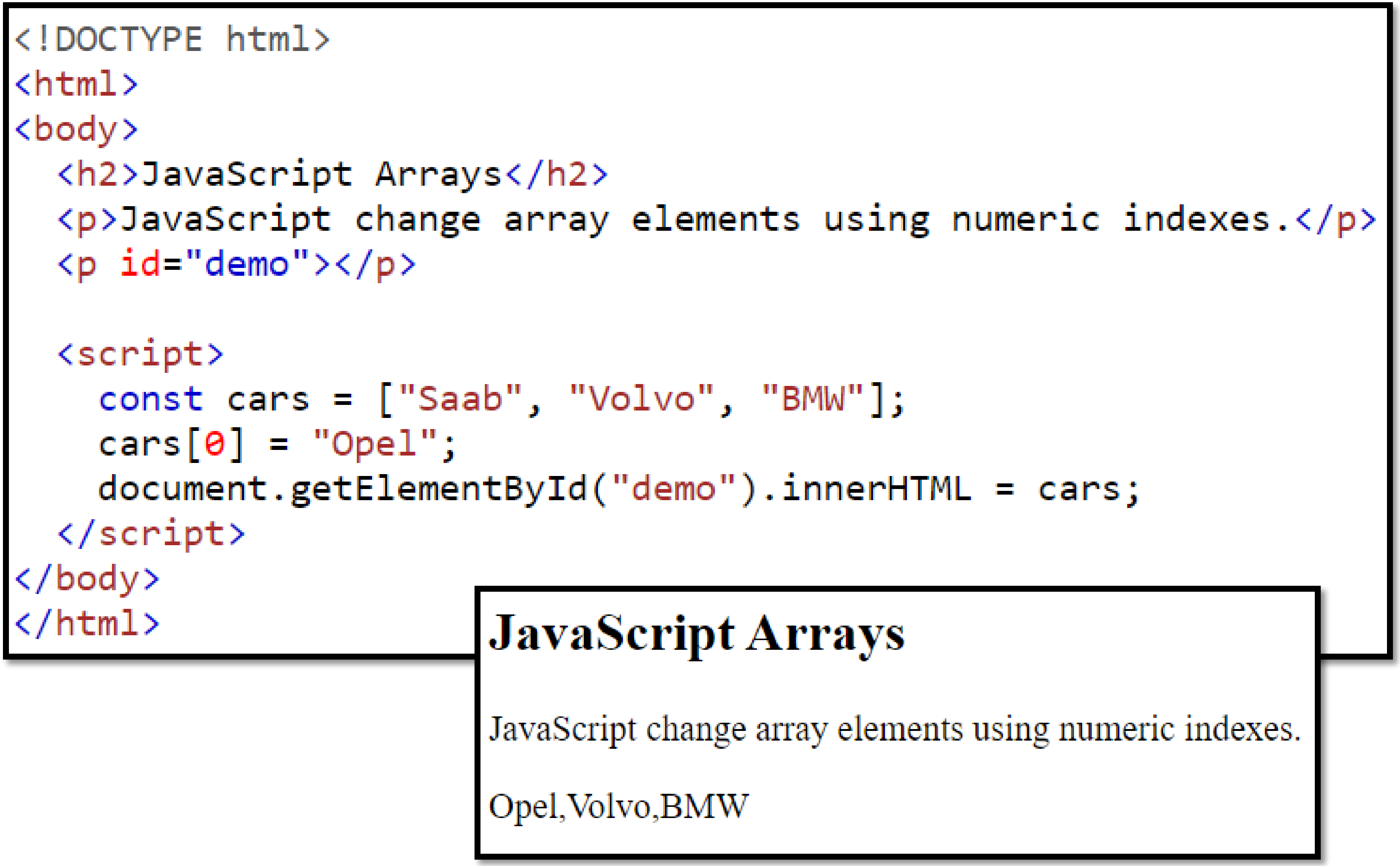
Syntax:

```
const array_name = [item1, item2, ...];
```



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Arrays</h2>
  <p id="demo"></p>

  <script>
    const cars = [];
    cars[0]= "Saab";
    cars[1]= "Volvo";
    cars[2]= "BMW";
    document.getElementById("demo").innerHTML = cars;
  </script>
</body>
</html>
```

Changing *an* Array Element



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript Arrays</h2>
  <p>JavaScript change array elements using numeric indexes.</p>
  <p id="demo"></p>

  <script>
    const cars = ["Saab", "Volvo", "BMW"];
    cars[0] = "Opel";
    document.getElementById("demo").innerHTML = cars;
  </script>
</body>
</html>
```

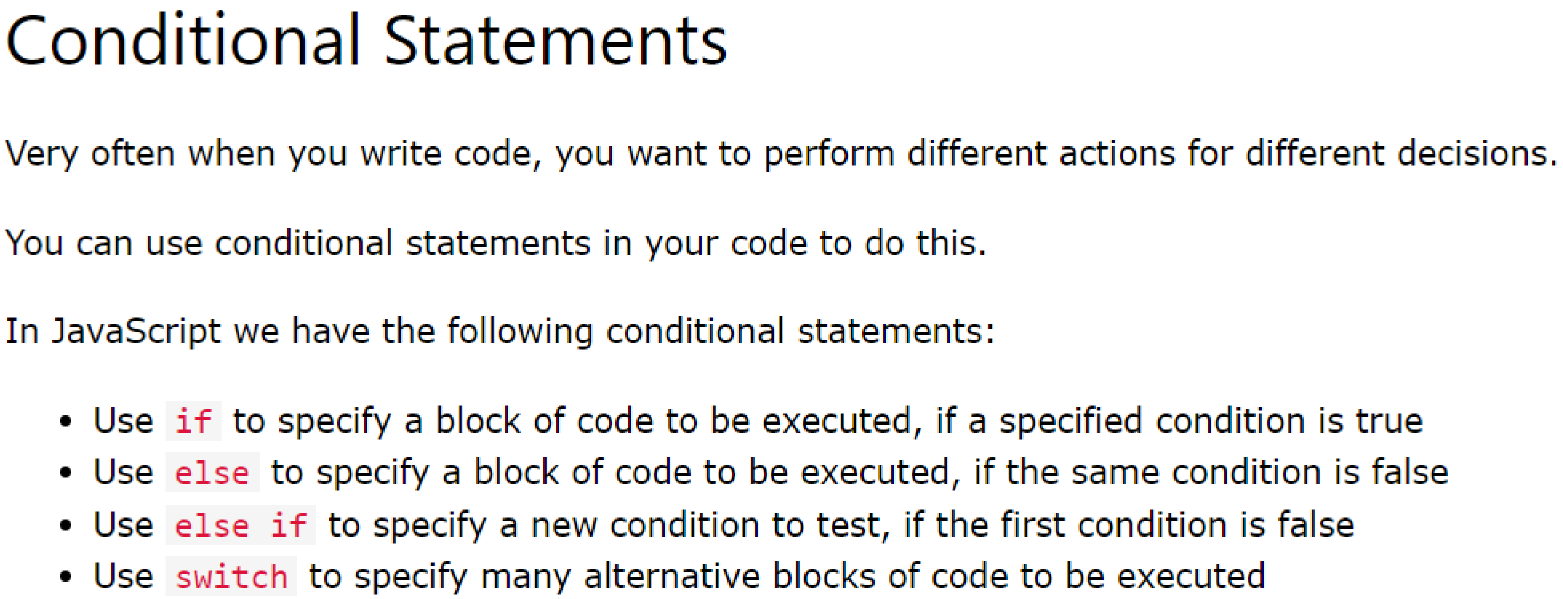
JavaScript Arrays

JavaScript change array elements using numeric indexes.

Opel,Volvo,BMW

JS Conditional Statements

JavaScript if, else, and else if



Conditional Statements

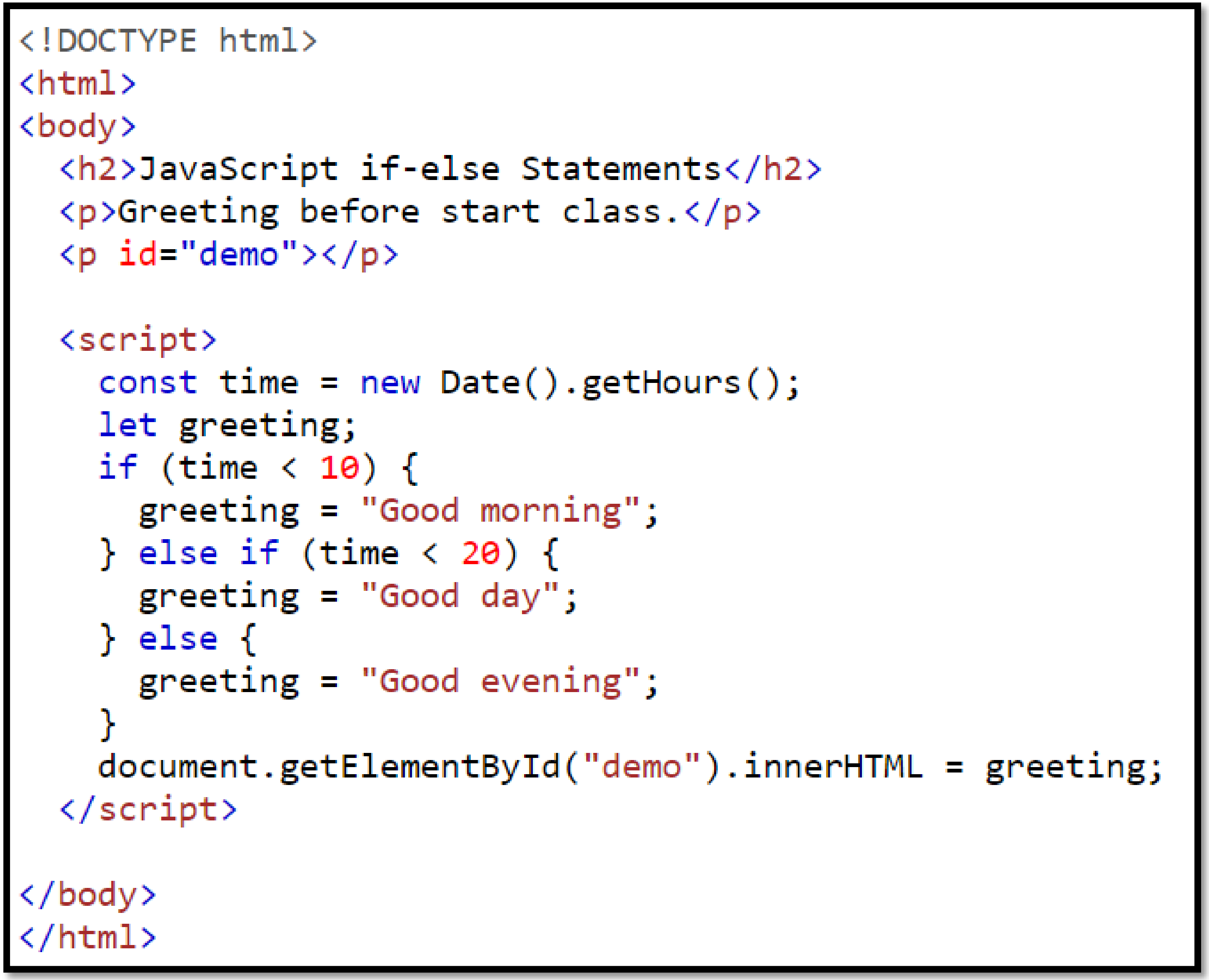
Very often when you write code, you want to perform different actions for different decisions.

You can use conditional statements in your code to do this.

In JavaScript we have the following conditional statements:

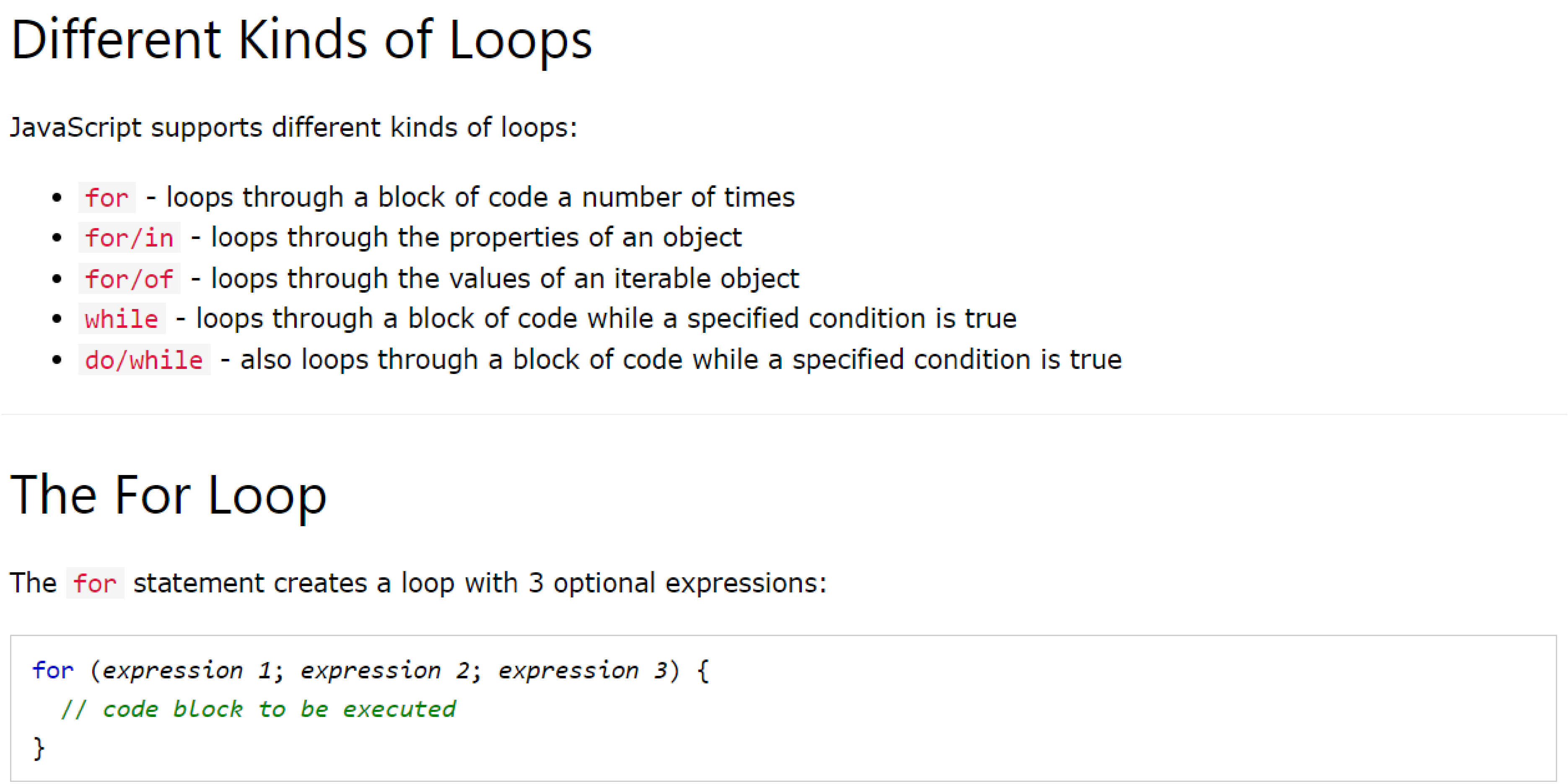
- Use `if` to specify a block of code to be executed, if a specified condition is true
- Use `else` to specify a block of code to be executed, if the same condition is false
- Use `else if` to specify a new condition to test, if the first condition is false
- Use `switch` to specify many alternative blocks of code to be executed

The else if Statement



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript if-else Statements</h2>
  <p>Greeting before start class.</p>
  <p id="demo"></p>

  <script>
    const time = new Date().getHours();
    let greeting;
    if (time < 10) {
      greeting = "Good morning";
    } else if (time < 20) {
      greeting = "Good day";
    } else {
      greeting = "Good evening";
    }
    document.getElementById("demo").innerHTML = greeting;
  </script>

</body>
</html>
```

JavaScript For Loop



Different Kinds of Loops

JavaScript supports different kinds of loops:

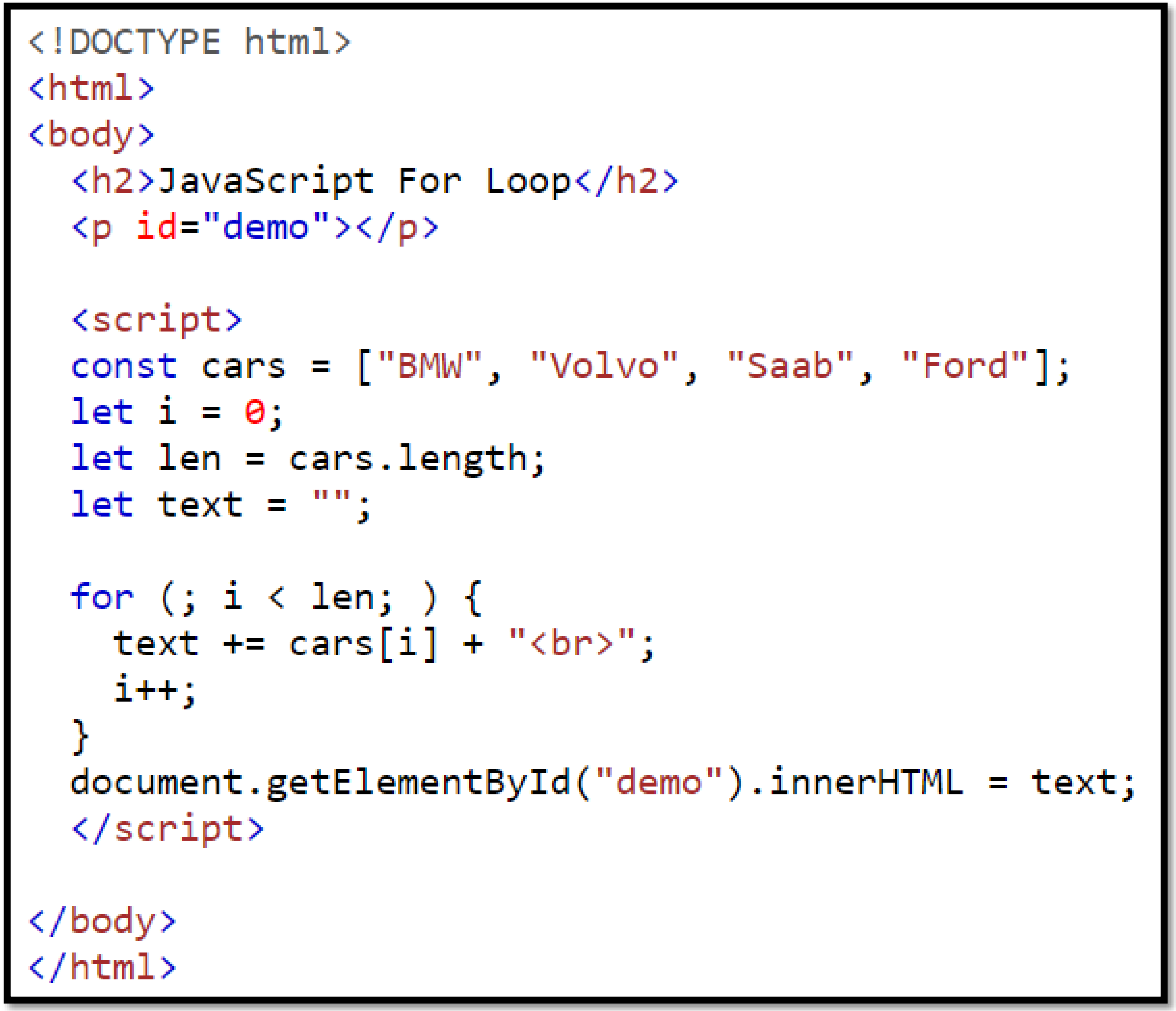
- `for` - loops through a block of code a number of times
- `for/in` - loops through the properties of an object
- `for/of` - loops through the values of an iterable object
- `while` - loops through a block of code while a specified condition is true
- `do/while` - also loops through a block of code while a specified condition is true

The For Loop

The `for` statement creates a loop with 3 optional expressions:

```
for (expression 1; expression 2; expression 3) {
  // code block to be executed
}
```

JavaScript For Loop



```
<!DOCTYPE html>
<html>
<body>
  <h2>JavaScript For Loop</h2>
  <p id="demo"></p>

  <script>
  const cars = ["BMW", "Volvo", "Saab", "Ford"];
  let i = 0;
  let len = cars.length;
  let text = "";

  for (; i < len; ) {
    text += cars[i] + "<br>";
    i++;
  }
  document.getElementById("demo").innerHTML = text;
  </script>

</body>
</html>
```

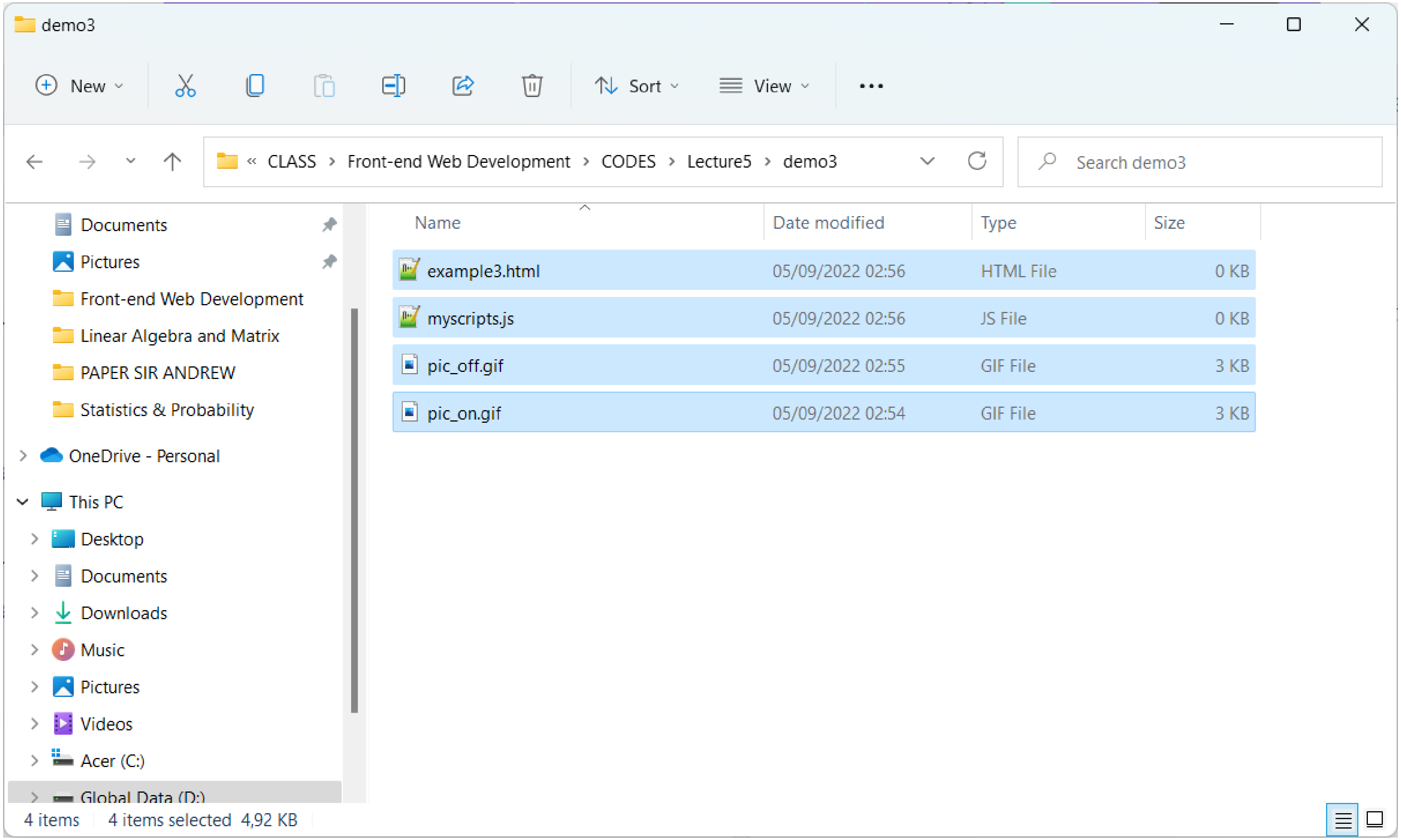
Exercise *for* Students

Exercise #1

JavaScript changes HTML attribute values to turn ON/OFF the light.

Create New Folder

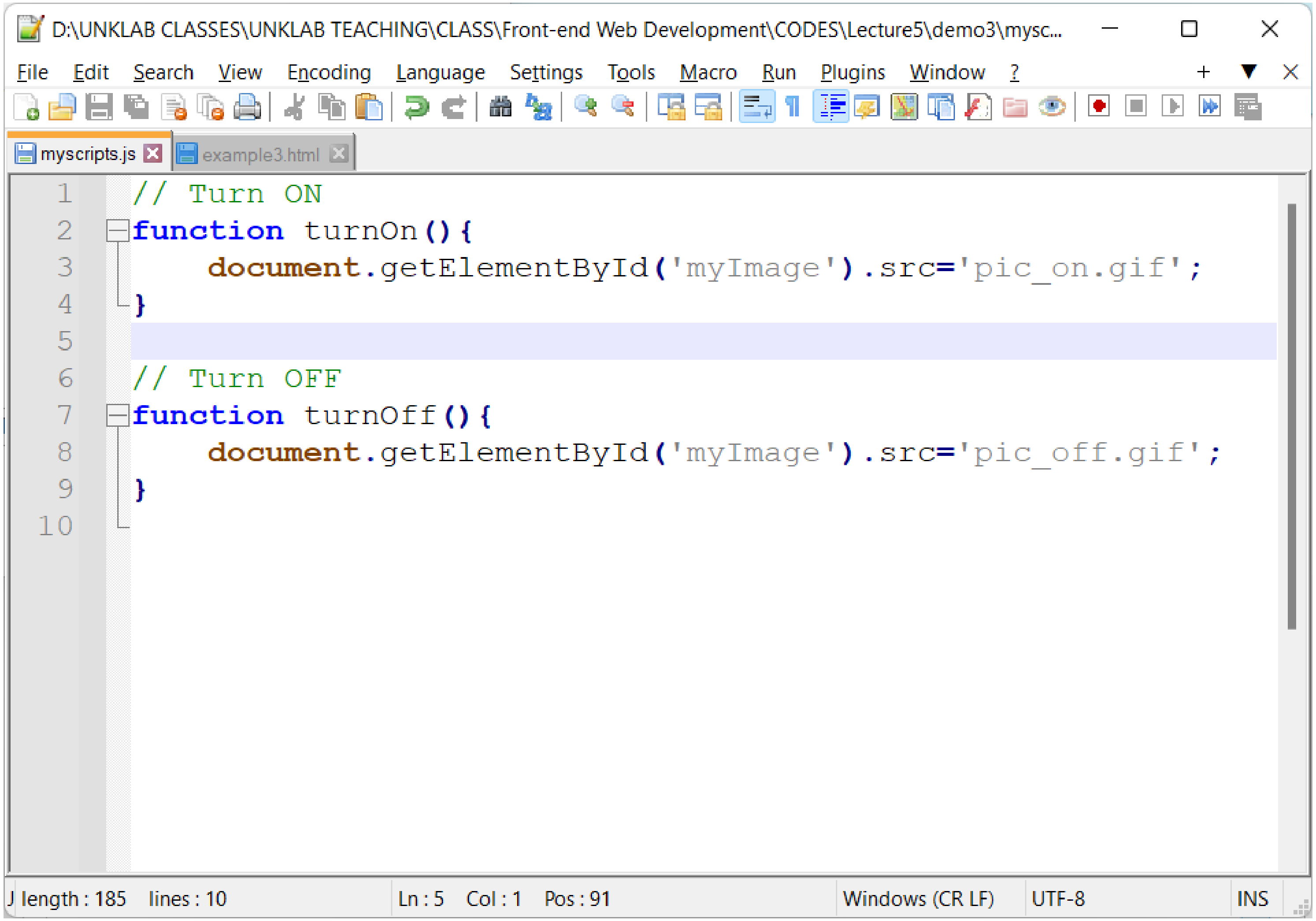
Buat folder baru dengan nama “demo3”.



Name	Date modified	Type	Size
example3.html	05/09/2022 02:56	HTML File	0 KB
myscripts.js	05/09/2022 02:56	JS File	0 KB
pic_off.gif	05/09/2022 02:55	GIF File	3 KB
pic_on.gif	05/09/2022 02:54	GIF File	3 KB

4 items 4 items selected 4,92 KB

Write Javascript Code

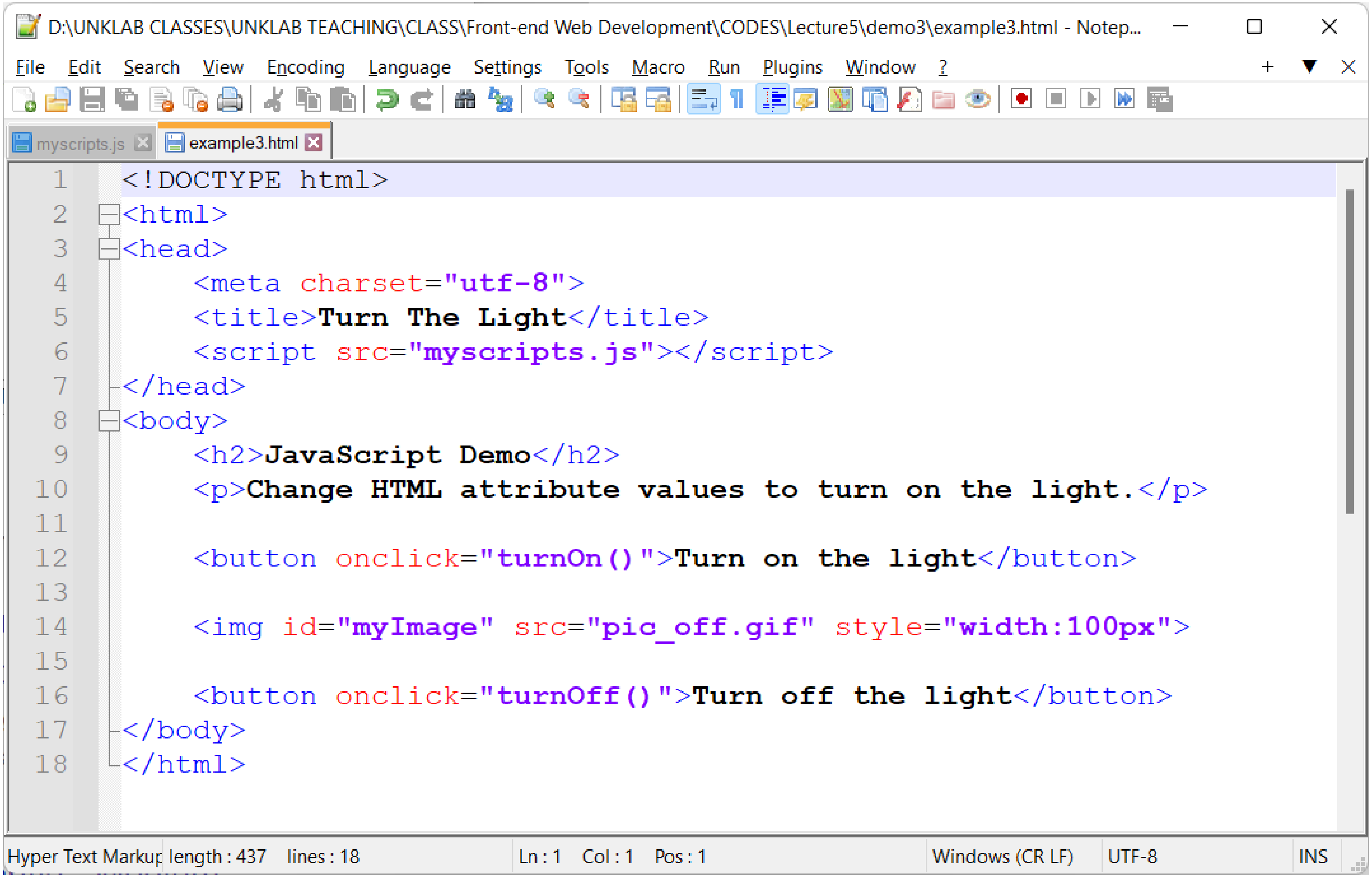


D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo3\mysc...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscripts.js
example3.html

```
// Turn ON
function turnOn(){
    document.getElementById('myImage').src='pic_on.gif';
}

// Turn OFF
function turnOff(){
    document.getElementById('myImage').src='pic_off.gif';
}

```

length : 185   lines : 10   Ln : 5   Col : 1   Pos : 91   Windows (CR LF)   UTF-8   INS

Write HTML Tags

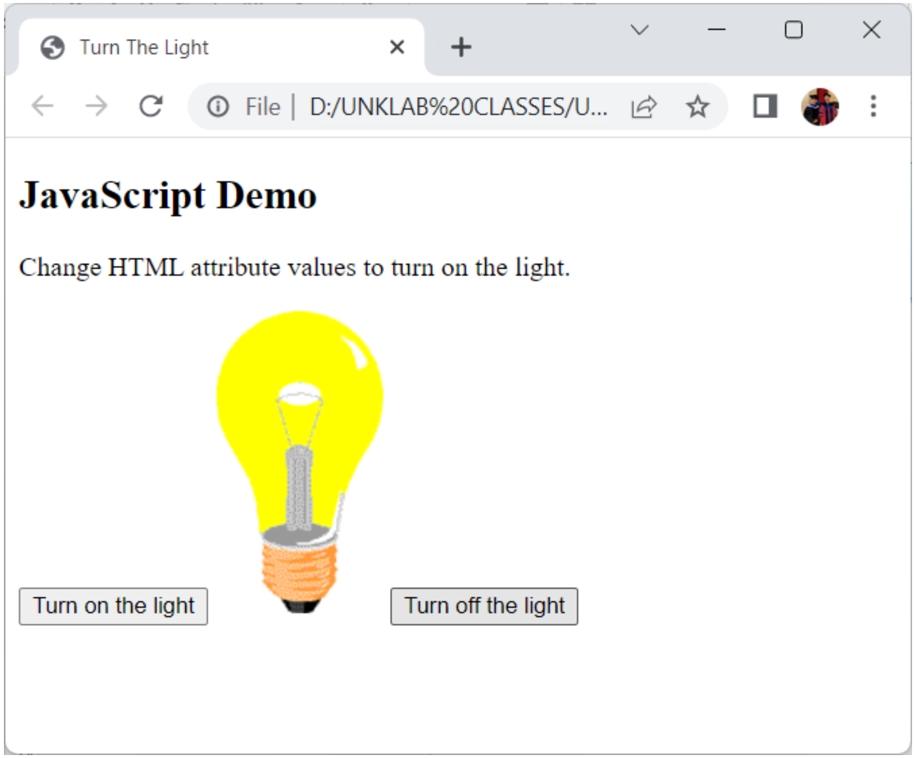


D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo3\example3.html - Notep...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
myscripts.js
example3.html

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Turn The Light</title>
    <script src="myscripts.js"></script>
</head>
<body>
    <h2>JavaScript Demo</h2>
    <p>Change HTML attribute values to turn on the light.</p>

    <button onclick="turnOn()">Turn on the light</button>

    <img id="myImage" src="pic_off.gif" style="width:100px">

    <button onclick="turnOff()">Turn off the light</button>
</body>
</html>
```

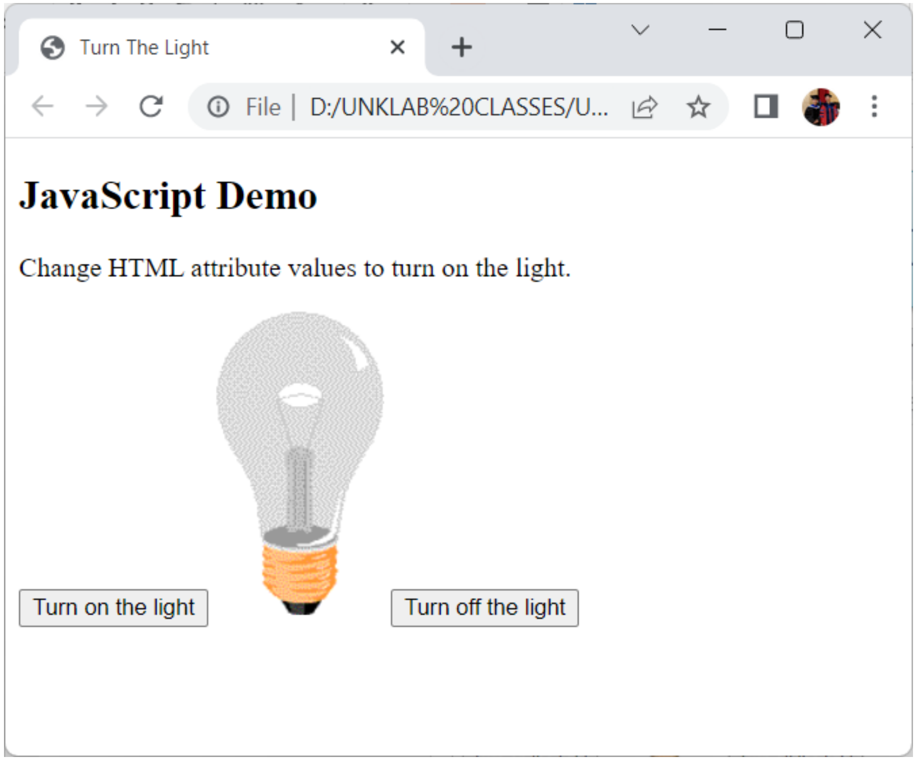
Hyper Text Markup length : 437 lines : 18
Ln : 1 Col : 1 Pos : 1
Windows (CR LF)
UTF-8
INS

Expected Output



JavaScript Demo

Change HTML attribute values to turn on the light.



JavaScript Demo

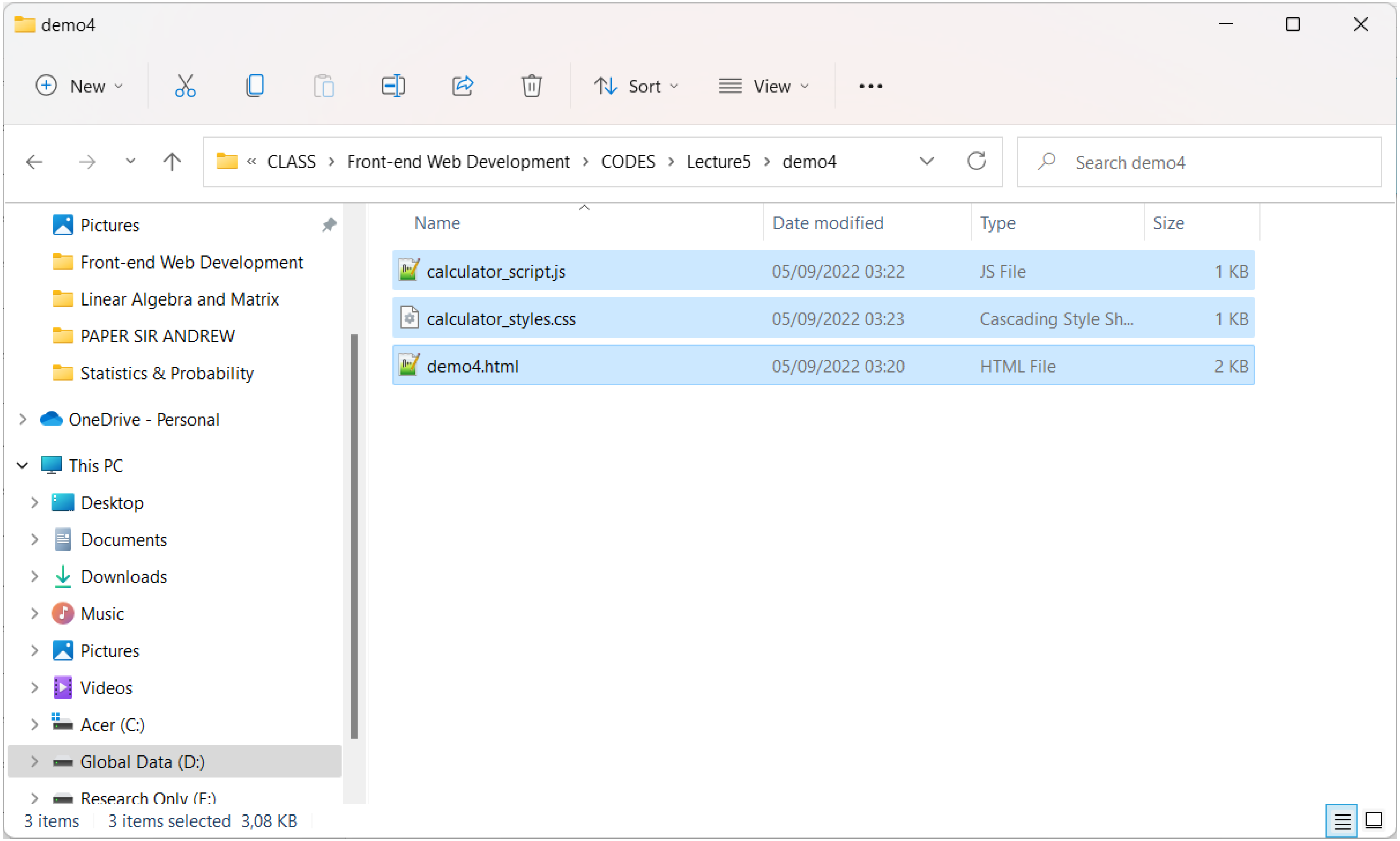
Change HTML attribute values to turn on the light.

Exercise #2

HTML, JavaScript and CSS to create simple calculator.

Create New Folder & Files

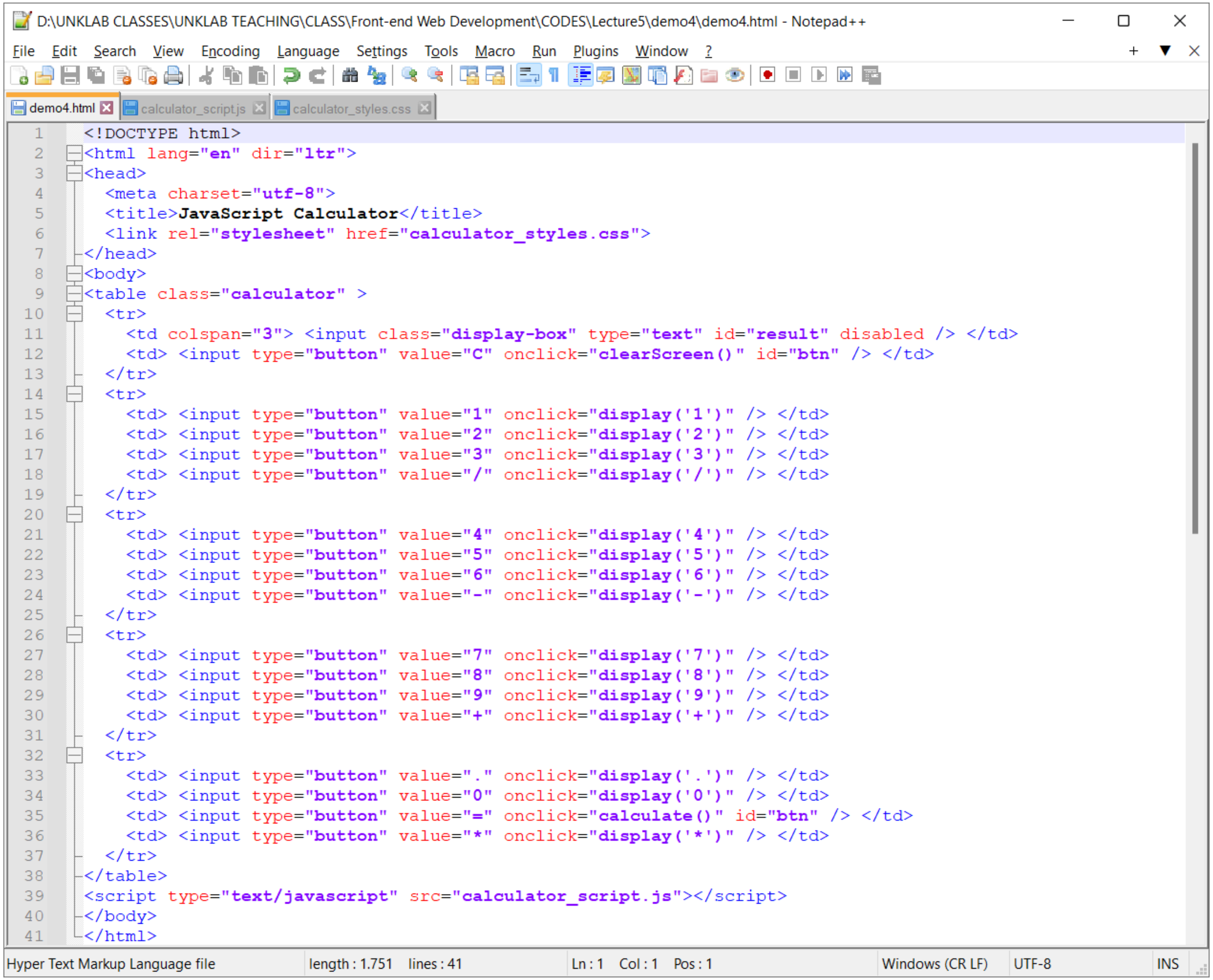
Buat folder baru dengan nama “demo4”.



Name	Date modified	Type	Size
calculator_script.js	05/09/2022 03:22	JS File	1 KB
calculator_styles.css	05/09/2022 03:23	Cascading Style Sh...	1 KB
demo4.html	05/09/2022 03:20	HTML File	2 KB

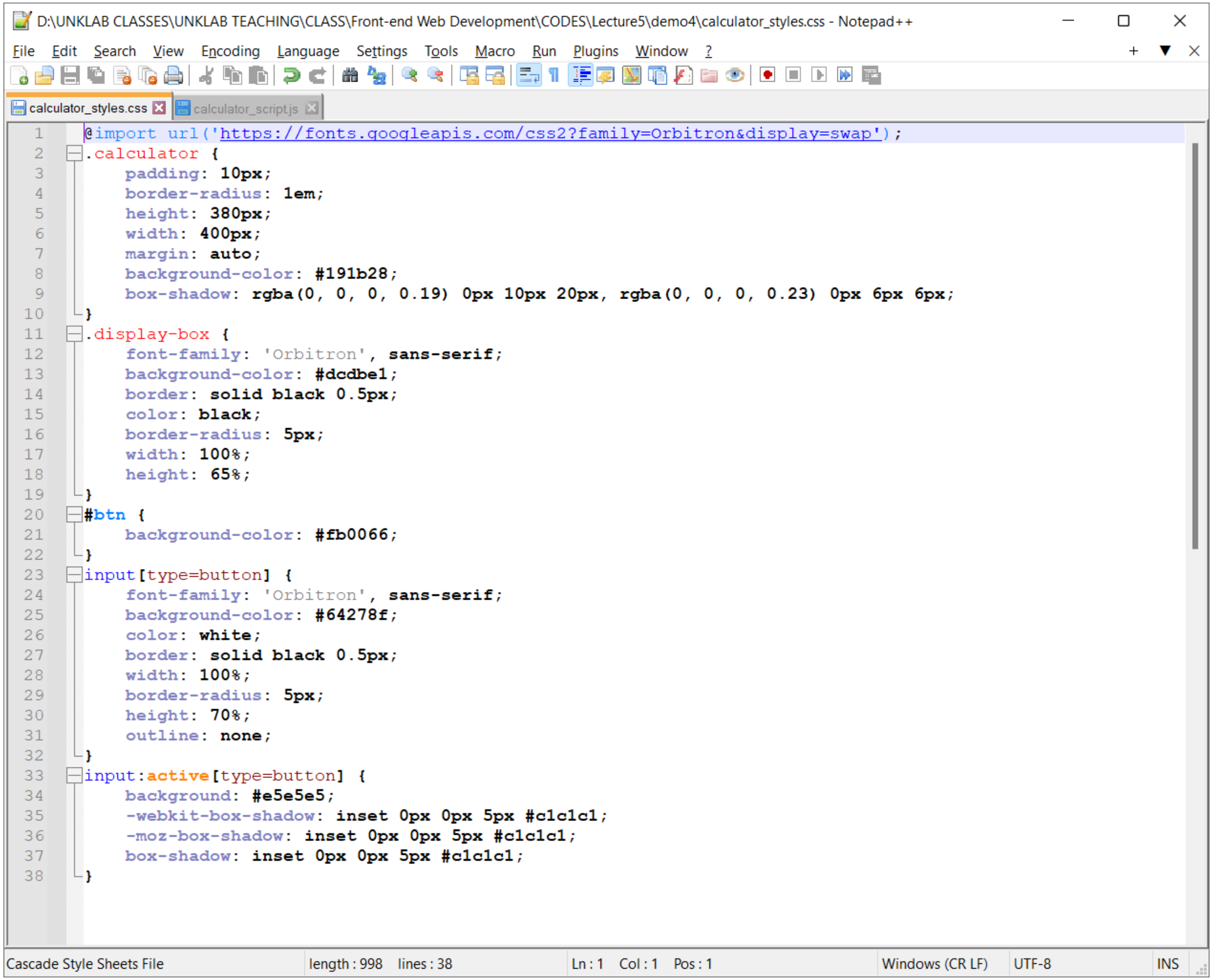
3 items 3 items selected 3,08 KB

Write HTML Tags


D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo4\demo4.html - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
demo4.html
calculator_script.js
calculator_styles.css
Hyper Text Markup Language file
length : 1.751 lines : 41
Ln : 1 Col : 1 Pos : 1
Windows (CR LF)
UTF-8
INS

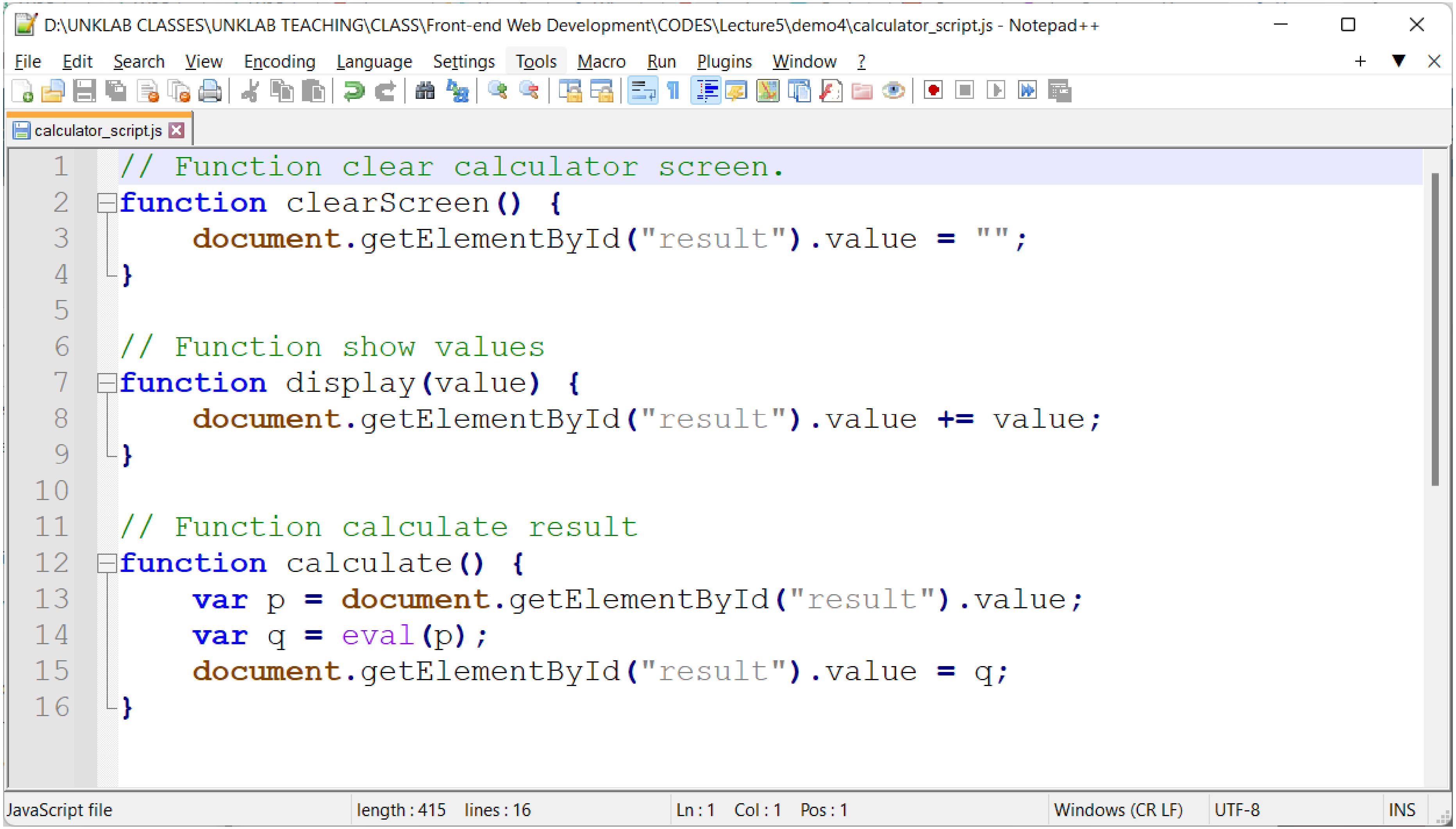
```html
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="utf-8">
  <title>JavaScript Calculator</title>
  <link rel="stylesheet" href="calculator_styles.css">
</head>
<body>
<table class="calculator" >
  <tr>
    <td colspan="3"> <input class="display-box" type="text" id="result" disabled /> </td>
    <td> <input type="button" value="C" onclick="clearScreen()" id="btn" /> </td>
  </tr>
  <tr>
    <td> <input type="button" value="1" onclick="display('1')" /> </td>
    <td> <input type="button" value="2" onclick="display('2')" /> </td>
    <td> <input type="button" value="3" onclick="display('3')" /> </td>
    <td> <input type="button" value="/" onclick="display('/')" /> </td>
  </tr>
  <tr>
    <td> <input type="button" value="4" onclick="display('4')" /> </td>
    <td> <input type="button" value="5" onclick="display('5')" /> </td>
    <td> <input type="button" value="6" onclick="display('6')" /> </td>
    <td> <input type="button" value="-" onclick="display('-')" /> </td>
  </tr>
  <tr>
    <td> <input type="button" value="7" onclick="display('7')" /> </td>
    <td> <input type="button" value="8" onclick="display('8')" /> </td>
    <td> <input type="button" value="9" onclick="display('9')" /> </td>
    <td> <input type="button" value="+" onclick="display('+')" /> </td>
  </tr>
  <tr>
    <td> <input type="button" value="." onclick="display('.')" /> </td>
    <td> <input type="button" value="0" onclick="display('0')" /> </td>
    <td> <input type="button" value="=" onclick="calculate()" id="btn" /> </td>
    <td> <input type="button" value="*" onclick="display('*')" /> </td>
  </tr>
</table>
<script type="text/javascript" src="calculator_script.js"></script>
</body>
</html>
```

Write CSS Code *for* Styling



```css
@import url('https://fonts.googleapis.com/css2?family=Orbitron&display=swap');
.calculator {
    padding: 10px;
    border-radius: 1em;
    height: 380px;
    width: 400px;
    margin: auto;
    background-color: #191b28;
    box-shadow: rgba(0, 0, 0, 0.19) 0px 10px 20px, rgba(0, 0, 0, 0.23) 0px 6px 6px;
}
.display-box {
    font-family: 'Orbitron', sans-serif;
    background-color: #dcdbe1;
    border: solid black 0.5px;
    color: black;
    border-radius: 5px;
    width: 100%;
    height: 65%;
}
#btn {
    background-color: #fb0066;
}
input[type=button] {
    font-family: 'Orbitron', sans-serif;
    background-color: #64278f;
    color: white;
    border: solid black 0.5px;
    width: 100%;
    border-radius: 5px;
    height: 70%;
    outline: none;
}
input:active[type=button] {
    background: #e5e5e5;
    -webkit-box-shadow: inset 0px 0px 5px #c1c1c1;
    -moz-box-shadow: inset 0px 0px 5px #c1c1c1;
    box-shadow: inset 0px 0px 5px #c1c1c1;
}
```

Write JavaScript Code

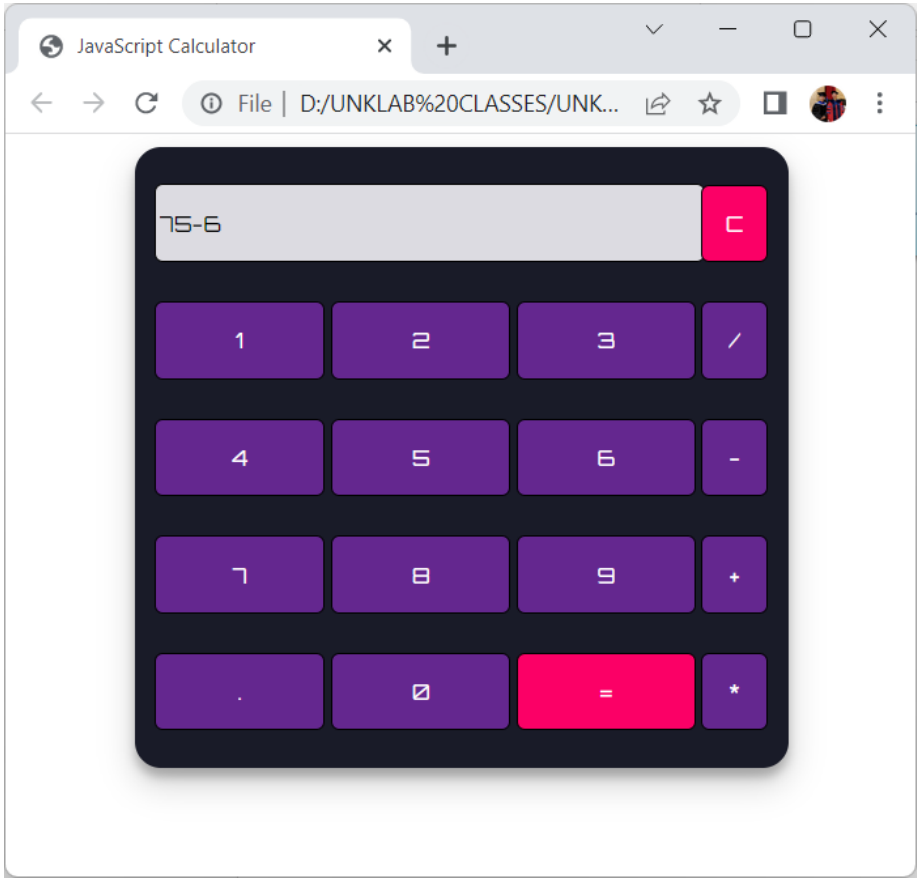


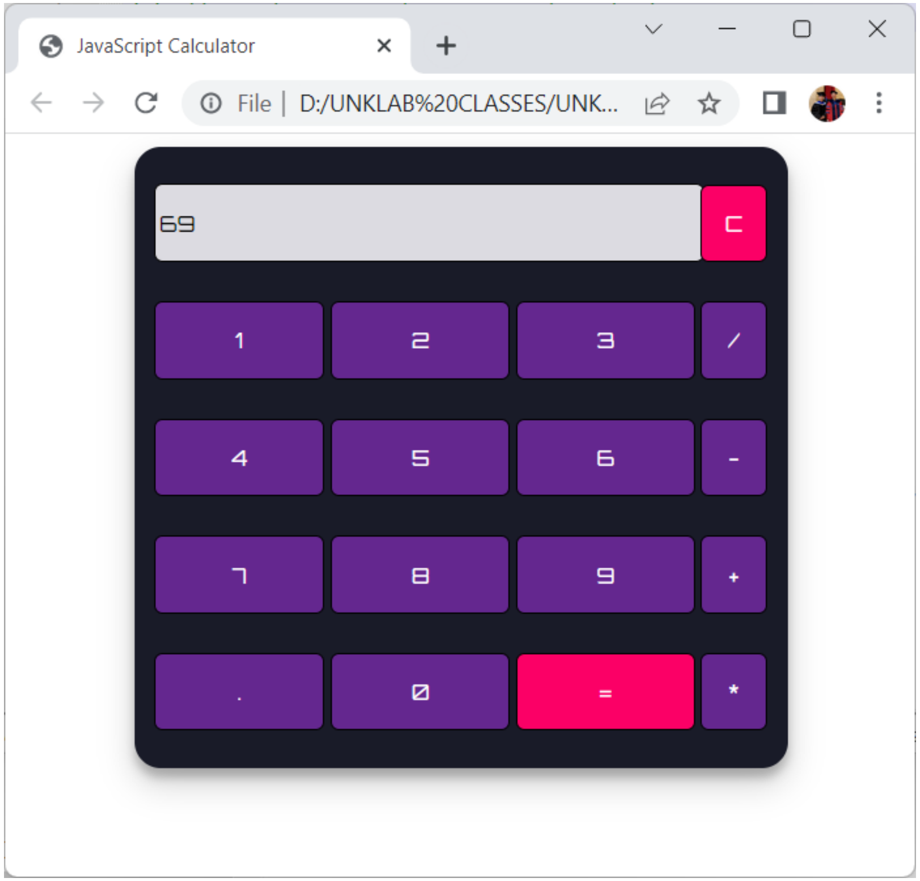
D:\UNKLAB CLASSES\UNKLAB TEACHING\CLASS\Front-end Web Development\CODES\Lecture5\demo4\calculator_script.js - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
calculator_script.js

```javascript
// Function clear calculator screen.
function clearScreen() {
    document.getElementById("result").value = "";
}

// Function show values
function display(value) {
    document.getElementById("result").value += value;
}

// Function calculate result
function calculate() {
    var p = document.getElementById("result").value;
    var q = eval(p);
    document.getElementById("result").value = q;
}
```

JavaScript file
length : 415 lines : 16
Ln : 1 Col : 1 Pos : 1
Windows (CR LF)
UTF-8
INS

Expected Output





END PRESENTATION

Thank you for your attention

Instructor: S – W – T