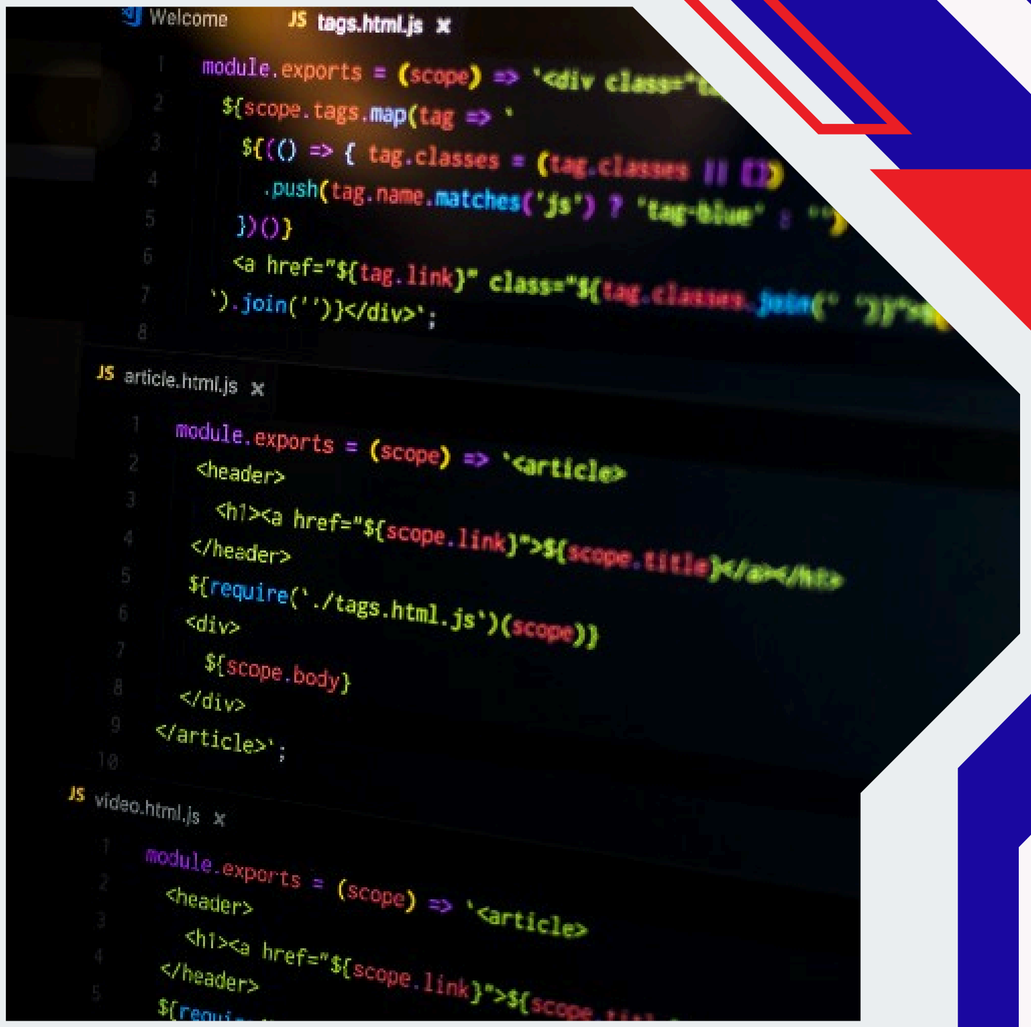
UNDERSTANDING HTML DOM ELEMENTS



WEB APPLICATION DEVELOPMENT | IT6315

JOHN PAUL M. MANUEL | ACLC COLLEGE OF TAYTAY

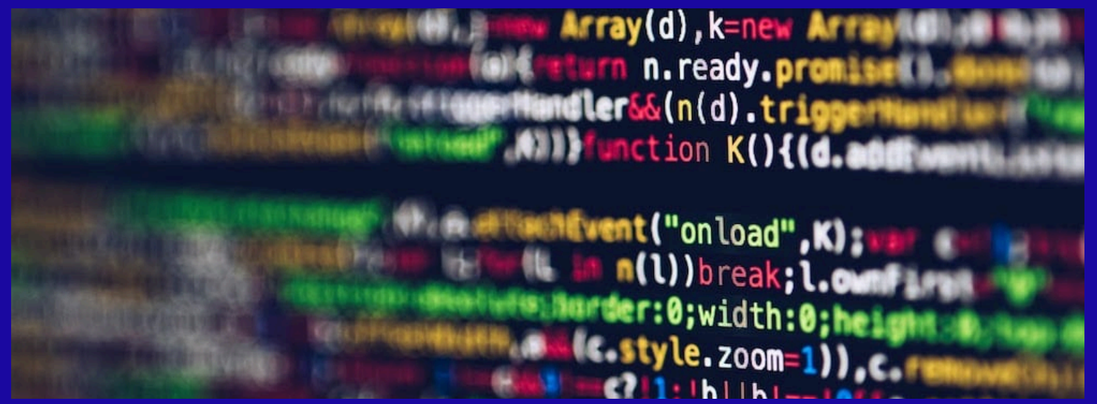
ACLC COLLEGE OF TAYTAY
(A Member of AMA Education System)

VERSION 01-250203-2425

WHAT IS THE DOM (DOCUMENT OBJECT MODEL) ?

DOM (Document Object Model) is a programming interface for web documents.

- It represents the page so that programs can manipulate its structure, style, and content.



DOM ELEMENTS OVERVIEW

What are DOM Elements?

DOM elements correspond to HTML tags and their attributes.
These elements can be manipulated using JavaScript.

Example of DOM Element:

```
<div id="example">Hello World</div>
```

div is the DOM element, with an id attribute of "example" and text content "Hello World."

DOM ELEMENTS

ACCESSING DOM ELEMENTS

Using JavaScript:

- getElementById(id) – Access elements by their unique ID.
- getElementsByTagName(tag) – Access elements by their tag name.
- querySelector(selector) – Access the first matching element by CSS selector.
- querySelectorAll(selector) – Access all matching elements by CSS selector.

```
let element = document.getElementById('example');

document.querySelector(".example");
```

DOM ELEMENTS

MODIFYING DOM ELEMENTS

Changing Content:

Use .innerHTML or .textContent to modify text or HTML inside an element.

```
element.innerHTML = "New Content";
```

Changing Attributes:

Use .setAttribute() to modify attributes.

```
element.setAttribute('class', 'new-class');
```

Changing Styles:

Directly modify .style to change inline styles.

```
element.style.color = "blue";
```

DOM ELEMENTS

EVENT HANDLING WITH DOM ELEMENTS

Adding Event Listeners:

Attach events like clicks, mouse movements, keypresses to DOM elements.

```
element.addEventListener('click', function() {
  alert('Element clicked!');
});
```

Common Events:

click, mouseover, keydown, submit, etc.

DOM ELEMENTS

MANIPULATING THE DOM STRUCTURE

Creating New Elements:

Use document.createElement() to create new HTML elements dynamically.

```
let newElement = document.createElement('p');
newElement.textContent = 'This is a new paragraph!';
document.body.appendChild(newElement);
```

Removing Elements:

Use .removeChild() to remove child elements.

```
document.body.removeChild(newElement);
```

DOM ELEMENTS

GETTING DATA FROM DOM ELEMENTS

Accessing Data in Text Content

.textContent: Retrieves the raw text content inside an element, excluding any HTML tags.

```
let element = document.getElementById('example');
let textData = element.textContent;
console.log(textData); // Output: "Hello World"
```

.innerText: Similar to .textContent, but it reflects visible text (i.e., excludes hidden text or elements). It also takes into account CSS styles like text-transform.

```
let visibleText = element.innerText;
console.log(visibleText); // Output might be different depending on styles
```

DOM ELEMENTS

GETTING DATA FROM DOM ELEMENTS

Accessing Attributes

.getAttribute(): Retrieves the value of a specified attribute of the element.

```
let element = document.getElementById('example');
let classValue = element.getAttribute('class');
console.log(classValue); // Output: value of class attribute, e.g.
"highlighted"
```

.id, .className: Directly access common attributes like id and class.

```
let elementId = element.id; // Direct access to id attribute
let className = element.className; // Direct access to class
attribute
```

DOM ELEMENTS

GETTING DATA FROM DOM ELEMENTS

Getting Form Data

Input Elements: For form elements like input, select, textarea, you can access the values they contain.

For <input> elements:

```
let inputValue = document.getElementById('username').value;
```

For <select> elements:

```
let selectedOption = document.getElementById('dropdown').value;
```

For <textarea> elements:

```
let textAreaValue = document.getElementById('comments').value;
```

DOM ELEMENTS

GETTING DATA FROM DOM ELEMENTS

Getting Data from Data Attributes

Using data-* Attributes: HTML5 introduced custom data attributes, which you can use to store extra data on DOM elements.

```
<div id="product" data-price="25.99" data-category="electronics">
</div>
```

Accessing data attributes with dataset:

```
let productElement = document.getElementById('product');
let price = productElement.dataset.price;
let category = productElement.dataset.category;
console.log(price, category); // Output: "25.99" "electronics"
```

DOM ELEMENTS

GETTING DATA FROM DOM ELEMENTS

Accessing HTML Content

.innerHTML: Retrieves the HTML content inside an element, including all child tags and content.

```
let divContent = document.getElementById('example').innerHTML;
console.log(divContent); // Output: HTML markup inside the element
```