**MIT xPRO Data Engineering Certificate**

**Document Object Model (DOM) & HTML**

**What is the DOM?**

The Document Object Model (DOM) of a web page is a programming interface for web documents. It represents the page's structure as a tree-like data structure where each node in the tree represents an HTML element, attribute, or piece of text. It provides a way for programs to interact with and modify every aspect of a web document, making interactive and dynamic web experiences possible. It is the essential bridge between the static HTML structure of a webpage and dynamic scripting languages like JavaScript.

When a web browser loads an HTML file it parses the HTML and creates an internal representation of that document – the DOM. This model allows programs, and specifically JavaScript, to:

* Access and manipulate the structure of the document: Add, remove, and modify HTML elements.
* Access and manipulate the content of the document: Change the text inside elements.
* Access and manipulate the style of elements: Change colors, fonts, sizes, and other CSS properties.
* React to events: Respond to user interactions like clicks and mouse movements.

Key characteristics of the DOM:

* Tree Structure: The DOM organizes the HTML elements in a hierarchical tree structure. The <html> element is the root of the tree, and all other elements are its descendants. This parent-child relationship is crucial for navigating and manipulating the document.
* Object-Oriented: The DOM represents each part of the HTML document as an object with properties and methods. For example, an <img> element is represented by an object with properties like src (the image source) and methods that allow you to interact with it.
* Dynamic: The DOM is not static. It can be dynamically updated and modified by JavaScript. This is what allows for interactive web pages where content and styling can change in response to user actions or other events.
* Platform and Language Neutral: While most commonly used with JavaScript in web browsers, the DOM is a W3C (World Wide Web Consortium) standard and is designed to be language-neutral. Other programming languages can also interact with the DOM in different environments.

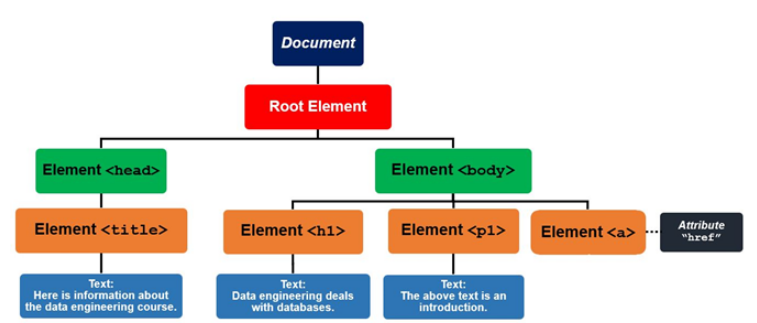
You can use JavaScript to act on the DOM to create an HTML page that defines a website. For example, you can change the color of the title on your webpage by simply changing your JavaScript code. These changes will be reflected on the DOM and, therefore, will be automatically updated on your website.

Using a DOM enables you to make any relevant changes to the web page without having to change the actual web code or the corresponding HTML page. The DOM serves as a bridge to communicate the changes made through JavaScript code to the website. For example, if you would like to change the top bar on a website, you would first need to make changes to your JavaScript code, then communicate that to the website via the DOM.

The DOM is also used when dealing with unstructured data in an HTML format. In this case, you can use the DOM to parse and read through the data to derive some information about it. In particular, you may want to understand which HTML *tags* enclose the type of data you are interested in so that you can access and visualize it.

**How is the DOM Structured?**

The DOM is a structural representation of your website. No matter what you want your web page to look like, the DOM always follows a tree-like structure. The figure below depicts an example of a DOM:



The above DOM is defined by the code below:

<html lang="en" dir="ltr">

<head>

<title>Here is information about the data engineering course.

</title>

</head>

<body>

<h1>Data engineering deals with databases.

</h1>

<p>The above text is an introduction.</p>

</body>

</html>

As you can see, the <title> *tag* is inside of the <head> *tag*. The text, “Here is information about the data engineering course.” will be the title of the web page in the header section. The <body> *tag* includes the main content of the web page.

The <html> *node* acts as the parent *node* to the <head> and <body> *nodes*. The <head> and <body> *nodes* act as parent *nodes* to the <title> and <h1> and <p> *nodes*. You can add more *nodes* depending on how you want your web page to appear. An HTML *attribute* can be thought of as an additional value that configures any element within an HTML page.

**HTML**

HTML stands for HyperText Markup Language. It's the standard markup language for creating web pages, which provides the basic framework for all the content that you see on the internet. It uses tags to describe different elements and how they should be organized within a document.

Key aspects of HTML:

* HyperText: Refers to the links (hyperlinks) that connect web pages together. Clicking on a link takes you to another document or a different part of the same document. HTML allows you to embed these links within your content.
* Markup Language: Unlike programming languages that tell a computer to do something, a markup language uses tags to describe the structure and content of a document. These tags tell the web browser how to display the information.
* Uses Tags: HTML uses angle brackets (< >) to enclose elements, which are the building blocks of an HTML page. Most elements have an opening tag (e.g., <h1>) and a closing tag (e.g., </h1>). Some elements are self-closing (e.g., <br />).
* Describes Structure: HTML defines the different parts of a webpage, such as headings, paragraphs, lists, images, links, forms, and more. It provides the semantic meaning to the content. For example, using the <h1> tag tells the browser (and search engines) that this text is the main heading of the page.
* Forms the DOM: When a web browser reads an HTML document, it parses it and creates the Document Object Model (DOM). The DOM is a tree-like representation of the HTML structure, which allows JavaScript to interact with and manipulate the content and style of the page.
* Foundation of web pages: Every web page you see on the internet is built using HTML (often in combination with CSS for styling and JavaScript for interactivity).
* Plain Text: HTML files are plain text documents, meaning you can create and edit them using any basic text editor. However, web developers typically use specialized code editors that offer features like syntax highlighting and error checking.
* Browser Interpretation: Web browsers (like Chrome, Firefox, Safari, etc.) read the HTML code and render the content visually on your screen according to the tags and their attributes.

**Structure of an HTML document**

These three elements form the basic structure of almost every HTML document on the web:

* The root element

<html>: The container for the entire HTML document. It tells the browser that this is an HTML document. This is the root element of every HTML page. It encloses all other HTML content.

* The head

<head>: Contains meta-information about the HTML document and links to external resources; not visible on the page itself. This information is generally not displayed directly on the web page itself. Instead, it provides instructions and context for the browser and other systems (like search engines). There can be only one <head> element in an HTML document. Common elements found within the <head> include:

<title>: Defines the title that appears in the browser tab or window title bar. This is also important for SEO (Search Engine Optimization).

<meta>: Provides various metadata about the HTML document, such as:

charset: Specifies the character encoding for the document (usually utf-8).

description: A brief summary of the page's content, often used by search engines.

keywords: Relevant keywords for search engines (less important now than in the past).

viewport: Configures how the page should be displayed on different devices (crucial for responsive design).

<link>: Establishes relationships with external resources, most commonly used to link to CSS stylesheets (<link rel="stylesheet" href="style.css">).

<style>: Contains inline CSS rules that apply to the document (generally best practice to use external stylesheets for larger projects).

<script>: Can contain or link to JavaScript code. Scripts placed in the <head> are typically executed before the page content is rendered.

* The body

<body>: Contains all the visible content of the HTML document. This is what users see and interact with on the webpage. All the elements that make up the structure and content of your page go inside the <body>, such as:

Headings: <h1> to <h6>

Paragraphs: <p>

Images: <img>

Links: <a> - typically, this element will have an href attribute that specifies the URL link to the linked source in the hyperlink.

Lists: <ul>, <ol>, <li>

Divisions and Spans: <div>, <span>

Forms: <form>, <input>, <button>

Tables: <table>, <tr>, <td>

Semantic elements: <article>, <nav>, <aside>, <footer>, etc.

Line break <br>